Functional Data Analysis, Regression, Classification and Clustering using fda.usc package

Manuel Oviedo de la Fuente

May 11, 2021

Índice general

Introduction

This vignette describes the usage of fda.usc in R. fda.usc package carries out exploratory and descriptive analysis of functional data exploring its most important features such as:

- Functional Data Representation
- Functional Regression
- Functional Classification
- Functional ANOVA

The co-author of fda.usc is *Manuel Febrero-Bande*, the contributors are **Pedro Galeano**, **Alicia Nieto** and **Eduardo Garcia-Portugues**.

CRAN Task View

R task view devoted to FDA: https://cran.r-project.org/web/views/FunctionalData.html

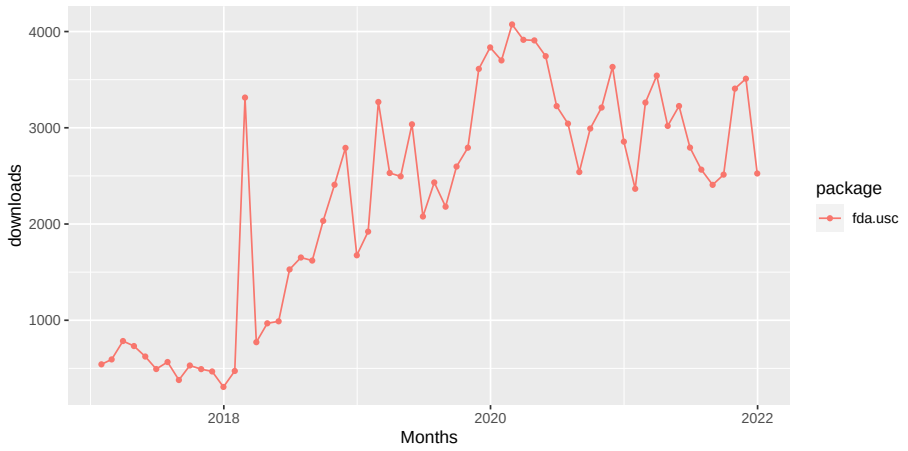
*__fda.usc__ is included in the set of the 7 recommended core packages.

Dependencies:

- Imports, dependent: MASS, fda, mgcv, nlme, rpart (depecrated)
- Reverse depends: GPFDA, ILS, MFHD, NITPicker, qcr
- Reverse imports: classiFunc, PCRedux, mlr
- Versions (by year): 5 (2012), 6 (2013), 2 (2014), 1 (2015), 2 (2016), 1 (2018), 1 (2019)

fda.usc package download by month:

```
## Warning: package 'ggplot2' was built under R version 4.1.1
```



Functional Data Analysis in R

1. fda It is a basic reference to work in R with functional data, see (Ramsay and Silverman, 2005a), http://ego.psych.mcgill.ca/misc/fda/

2. Ferray and Vieu, (2006) processed FD from a nonparametric point of view (normed or semi–normed functional spaces). These authors are part of the French group STAPH maintaining the page http://www.lsp.ups-tlse.fr/staph/

3. Core packages:

- FDboost: Boosting Functional Regression Models
- fds: Functional Data Sets
- ftsa: Functional Time Series Analysis
- fdasrvf: Elastic Functional Data Analysis
- refund: Regression with Functional Data
- fdapace: Functional Data Analysis and Empirical Dynamics

4.Interactive tools: + StatFda: exploratory analysis and functional regression models,http://www.statfda.com forand refund.shiny package for interactive plotting. + refund.shiny: Interactive Plotting for Functional Data Analyses + tidyfun: makes data wrangling and exploratory analysis of functional data easier, https://fabian-s.github.io/tidyfun

5. Other packages:

- GPFDA: Use Functional regression as the mean structure and Gaussian Process as the covariance structure.
- MFHD: Multivariate Functional Halfspace Depth
- rainbow: Bagplots, Boxplots and Rainbow Plots for Functional Data
- NITPicker: Finds the Best Subset of Points to Sample
- fdatest: Interval Testing Procedure for Functional Data
- fdakma: Functional Data Analysis: K-Mean Alignment
- fdaMixed: Functional data analysis in a mixed model framework
- geofd: Spatial Prediction for Function Value Data
- goffda: Goodness-of-Fit Tests for Functional Data
- mlr: Machine Learning in R

Installation

Like many other R packages, the simplest way to obtain `fda.usc` is to install it directly from CRAN. Type the following command in R console:

```
install.packages("fda.usc", repos = "http://cran.us.r-project.org")
```

Users may change the `repos` options depending on their locations and preferences. Other options such as the directories where to install the packages can be altered in the command. For more details, see `help(install.packages)`.

Here the R package has been downloaded and installed to the default directories.

Alternatively, users can download the package source at http://cran.r-project.org/web/packages/fda.usc/index.html and type Unix commands to install it to the desired location.

Quick Start

The purpose of this section is to give users a general sense of the package, including the components, what they do and some basic usage. We will briefly go over the main functions, see the basic operations and have a look at the outputs. Users may have a better idea after this section what functions are available, which one to choose, or at least where to seek help. More details are given in later sections.

First, we load the `fda.usc` package:

```
library(fda.usc)
```

Capítulo 1

Functional Data: Definition, Representation and Manipulation

1.1 Some definitions of Functional Data

Functional data analysis is a branch of statistics that analyzes data providing information about curves, surfaces or anything else varying over a continuum. The continuum is often time, but may also be spatial location, wavelength, probability, etc.

Functional data analysis is a branch of statistics concerned with analysing data in the form of functions.

- Definition 1: A random variable $\mathcal{X}$ is called a functional variable if it takes values in a functional space $\mathcal{E}$ –complete normed (or seminormed) space–, (Ferraty and Vieu, 2006)
- Definition 2: A functional dataset $\{\mathcal{X}_1, \dots, \mathcal{X}_n\}$ is the observation of n functional variables $\mathcal{X}_1, \dots, \mathcal{X}_n$ identically distributed as $\mathcal{X}$, (Ferraty and Vieu, 2006).

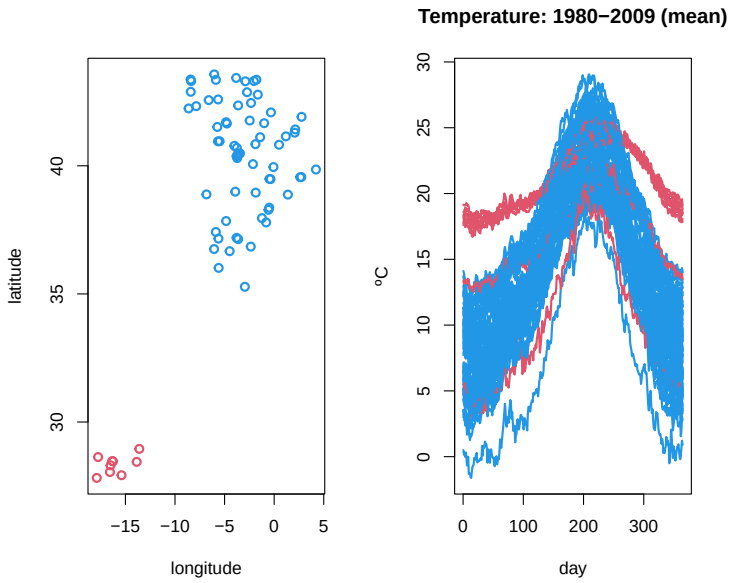
A research stream focuses on the computational treatment that is used to working with functional data as an extension of multivariate data. Thus, the following definition of functional data proposed by (Müller et al., 2008) is also common in practice.

- Definition 3. Functional data is multivariate data with an ordering on the dimensions, so that $a = t_1 < t_2, \dots, t_{m-1} < b = t_m$.

1.2 In fda.usc: "The data are curves''

$X_i(t)$ represents the mean temperature (averaged over 1980-2009 years) at the ith weather station in Spain, and at time time t during the year.

Temperature curves (right) located in Spanish airports (left),



But what code has been used to obtain the graph?

```
data(aemet)
par(mfrow=c(1,2))
CanaryIslands <- ifelse(aemet$df$latitude < 31, 2, 4)
plot(aemet$df[,c("longitude","latitude")], col = CanaryIslands, lwd=2)
plot(temp, col = CanaryIslands, lwd=2)
```

1.2.1 Definition of –fdata– class in R

Definition of fdata class object: An object called `fdata` as a list of the following components:

- `data`: typically a matrix of *(n x m)* dimension which contains a set of n curves discretized in m points or `argvals`.
- `argvals`: locations of the discretization points, by default: $t_1 = 1, \ldots, t_m = m\}$.

- `rangeval`: rangeval of discretization points.
- `names`: (optional) list with three components: `main`, an overall title, `xlab`, a title for the x axis and `ylab`, a title for the y axis.

```
lapply(aemet,class)
names(aemet$df)
lapply(aemet$temp,class)
temp <- aemet$temp
names(temp)
dim(temp)
length(argvals(temp))
rangeval(temp)
temp$names
```

1.2.2 Some utilities of fda.usc package

1. Basic operations for fdata class objects:

- Group Math: `abs`, `sqrt`, `floor`, `ceiling`,semimetric.basis(),trunc,round,signif,exp,log,cos,sin,tan‘.
- Other operations: `[]`, `is.fdata()`, `c()`, `dim()`, `ncol()`, `nrow()`.

```
is.fdata(aemet$temp)
is.fdata(aemet$df)
dim(aemet$df)
dim(aemet$temp)
```

2. Convert the class

- The class **fdata** only uses the evaluations at the discretization points.
- The `fdata2fd()` converts **fdata** object to **fd** object (using the basis representation).
- Inversely, the `fdata()` converts object of class: **fd**, **fds**, **fts**, **sfts**, **vector**, **matrix**, **data.frame** to an object of class **fdata**.

```
temp.fd=fdata2fd(temp,type.basis="fourier",nbasis=15)
temp.fdata=fdata(temp.fd) #back to fdata
class(temp.fd)
```

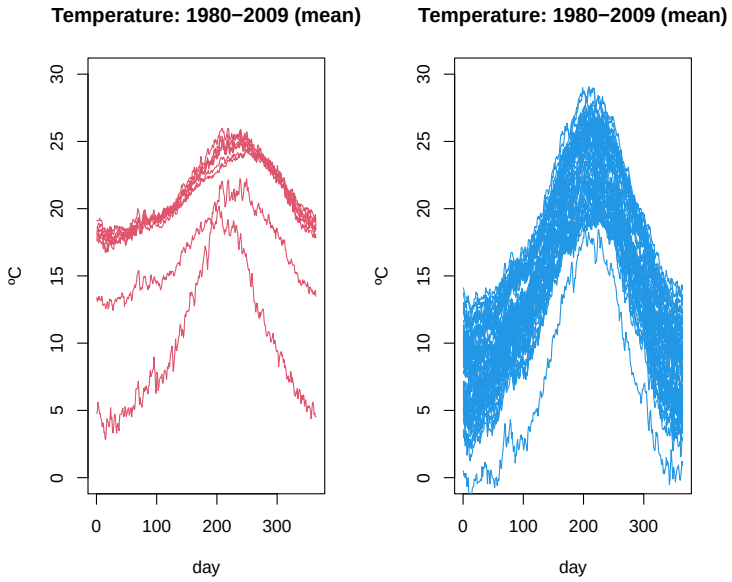
```
## [1] "fd"
```

```
class(temp.fdata)
```

```
## [1] "fdata"
```

- `split.fdata()` and `unlist()`: A wrapper for the split and unlist function for fdata object

```
# Canary Islands in red vs Iberian Peninsula in blue
l1 <- fda.usc:::split.fdata(temp,CanaryIslands)
par(mfrow=c(1,2))
plot(l1[[1]],col=2,ylim=c(0,30))
plot(l1[[2]],col=4,ylim=c(0,30))
```




```
dim(l1[[1]]);dim(l1[[2]])
```

```
## [1]   9 365
```

```
## [1]  64 365
```

- Group Ops: + -, *, /, ^, %%, %/%, &, |, !, ==, !=, <, <=, >=, >.

```
l1[[1]]==l1[[2]]
```

```
## [1] FALSE
```

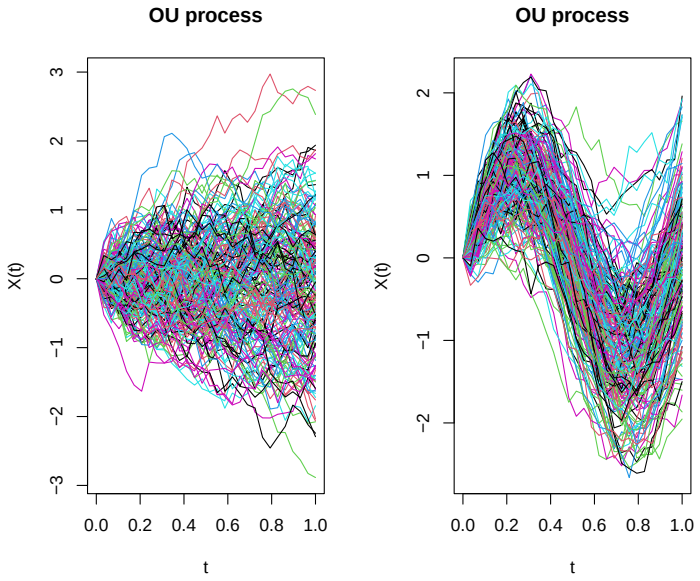
- `order.fdata()` A wrapper for the order function. The funcional data is ordered w.r.t the sample order of the values of vector.

```
temp2 <- order.fdata(order(CanaryIslands),temp)
```

- Generate random process of fdata class.

`rproc2fdata()` function generates Functional data from: Ornstein Uhlenbeck process, Brownian process, Gaussian process or Exponential variogram process.

```
par(mfrow=c(1,2))
lent <- 30
tt <- seq(0,1,len=lent)
xgen1 <- rproc2fdata(200,t=tt,sigma="OU",par.list=list("scale"=1))
plot(xgen1)
mu <- fdata(sin(2*pi*tt),tt)
xgen2 <- rproc2fdata(200,mu=mu,sigma="OU",par.list=list("scale"=1))
plot(xgen2)
```

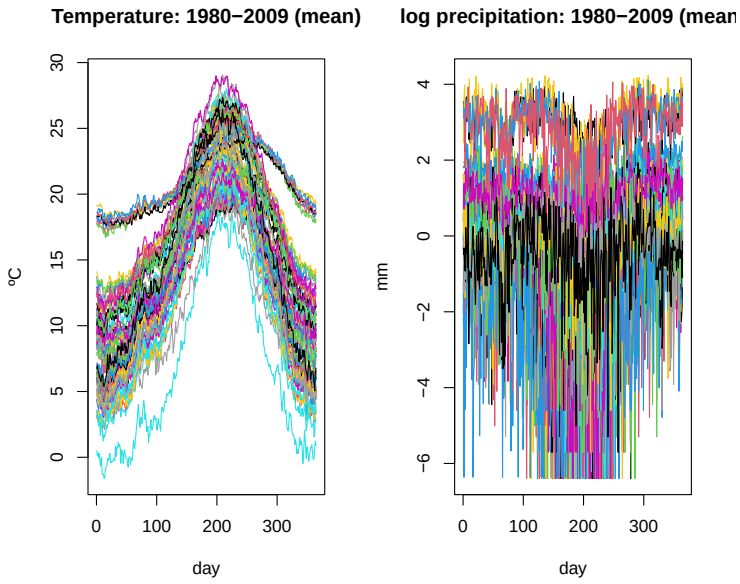


1.2.3 Definition of –ldata– class in R

ldata is a list with two type of objects:

- `df` is a data frame with the multivariate data with n rows.
- `...` objects of class `fdata` with n rows.

```
ldat <- ldata(df=aemet$df,temp=aemet$temp, logprec=aemet$logprec )
plot(ldat)
```



1.3 Resume by smoothing

If we supposed that our functional data $Y(t)$ is observed through the model $Y(t_i) = X(t_i) + \varepsilon(t_i)$ where the residuals $\varepsilon(t)$ are independent with $X(t)$.

We can to get back the original signal $X(t)$ using a linear smoother:

$$\hat{X}(t_i) = \sum_{i=1}^{n} s_i(t_j) Y(t_i) \Rightarrow \hat{\mathbf{X}} = \mathbf{SY}$$

where $s_i(t_j)$ is the weight that the point t_j gives to the point t_i.

We use two methods to estimate the smoothing matrix S.

1. Finite representation in a fixed basis (Ramsay and Silverman, 2005a):

Let $X(t) \in \mathcal{L}_2$,

$$X(t) = \sum_{k \in \mathbb{N}} c_k \phi_k(t) \approx \sum_{k=1}^{K} c_k \phi_k(t) = c^\top \Phi$$

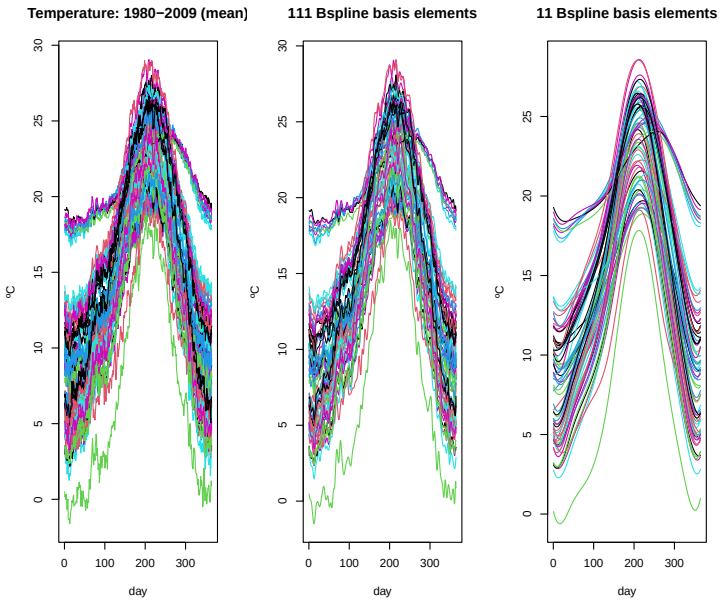
The smoothing matrix is given by: $\mathbf{S} = \Phi(\Phi^\top W \Phi + \lambda R)^{-1} \Phi^\top W$ where λ is the penalty parameter.

Type of basis:

- Fourier: Design to represent periodic functions. Orthonormal
- BSplines: Set of polynomials (of order m) defined in subintervals constructed in such a way that in the border of the subintervals the polynomials coincide (up to $m-2$ derivative). Banded
- Wavelets: Powerful especially when the grid is a power of 2. Orthonormal
- Polynomials: Adjust a polynomial to the whole curve.

Example: raw (left) and smoothed temperature curves with 111 (center) and 11 (right) elements of a B-spline base:

```
bsp11 <- create.bspline.basis(temp$rangeval,nbasis=11)
bsp111 <- create.bspline.basis(temp$rangeval,nbasis=111)
S.bsp11 <-  S.basis(temp$argvals, bsp11)
S.bsp111 <- S.basis(temp$argvals, bsp111)
temp.bsp11 <- temp.bsp111<-temp
temp.bsp11$data <- temp$data%*%S.bsp11
temp.bsp111$data <- temp$data%*%S.bsp111
par(mfrow=c(1,3))
plot(temp)
plot(temp.bsp111, main="111 Bspline basis elements")
plot(temp.bsp11, main="11 Bspline basis elements")
```




```
# Another way
out1<-optim.basis(temp, type.CV=CV.S)
names(out1)
```

```
##  [1] "gcv"           "numbasis"     "lambda"       "fdataobj"     "fdata.est"
##  [6] "gcv.opt"       "numbasis.opt" "lambda.opt"   "S.opt"        "base.opt"
```

```
out1$numbasis.opt
```

```
## [1] 76
```

2. Kernel Smoothing (Ferraty and Vieu, 2006)

The problem is to estimate the smoothing parameter or bandwidth $\nu = h$ that better represents the functional data using kernel smoothing. Now,the nonparametric smoothing of functional data is given by the smoothing matrix S:

$$s_{ij} = \frac{1}{h} K\left(\frac{t_i - t_j}{h}\right)$$

$$S(h) = (s_j(t_i)) = \frac{K\left(\frac{t_i - t_j}{h}\right)}{\sum_{k=1}^{\top} K\left(\frac{t_k - t_j}{h}\right)}$$

where h is the bandwidth and $K()$ the Kernel function.

Different types of kernels $K()$ are defined in the package, see `Kernel` function.

Example: three examples of non-parametric smoothing with different values of the bandwidth parameter ($h = 1$, left; $h = 10$, center and h selected by the GCV criteria, right).

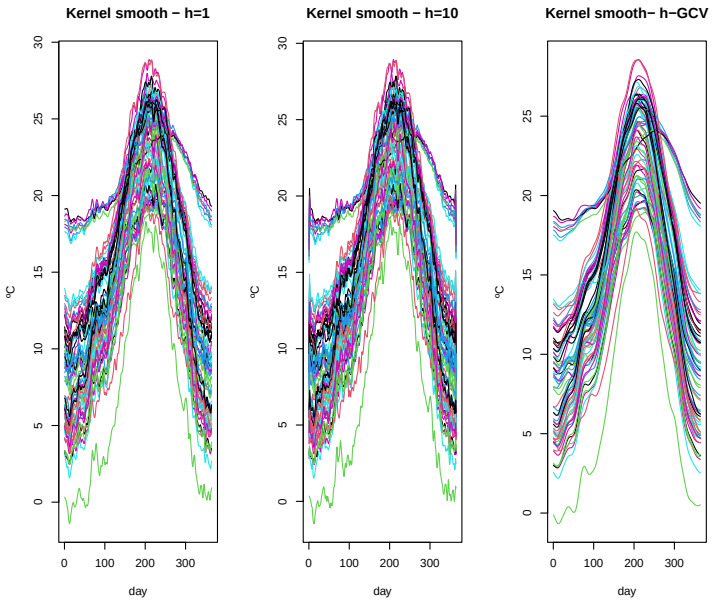
```
S <- S.NW(temp$argvals,h=1)
temp <- aemet$temp
temp.hat <- temp
temp.hat$data <- temp$data%*%S
args(optim.np)
```

```
## function (fdataobj, h = NULL, W = NULL, Ker = Ker.norm, type.CV = GCV.S,
##     type.S = S.NW, par.CV = list(trim = 0, draw = FALSE), par.S = list(),
##     correl = TRUE, verbose = FALSE, ...)
## NULL
```

```
# Another way
temp.est <- optim.np(temp,h=1)$fdata.est
par(mfrow=c(1,3))
plot(temp.est,main="Kernel smooth - h=1")
temp.est == temp.hat
```

```
## [1] FALSE
```

```
plot(temp.hat, main="Kernel smooth - h=10")
plot(optim.np(temp)$fdata.est, main="Kernel smooth- h-GCV")
```



3. Finite representation in a Data-driven basis (Cardot et al., 1999)

- Functional Principal Components (FPC) Functional Principal Components Analysis (FPCA) and the Functional PLS (FPLS) allow display the functional in a few components.

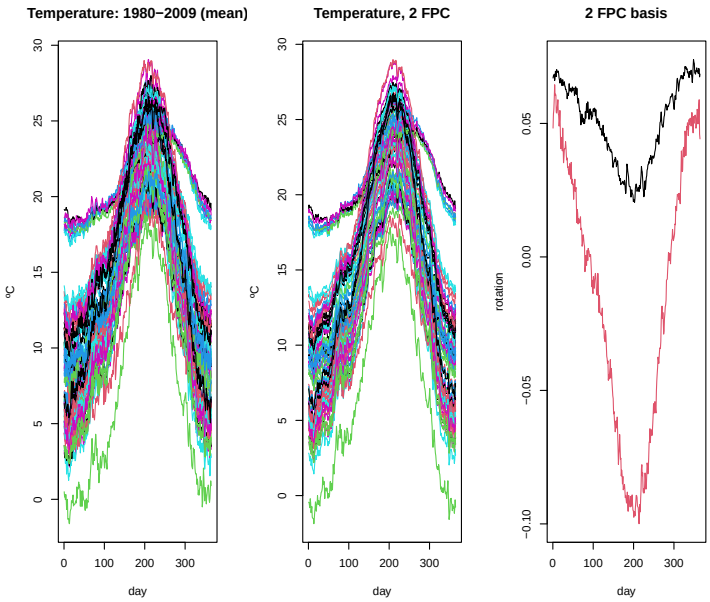
The functional data can be rewritten as a decomposition in an orthonormal PC basis: maximizing the variance of $X(t)$:

$$\hat{X}_i(t) = \sum_{i=1}^{K} f_{ik}\xi_k(t)$$

where $f_{i,k}$ is the score of the principal component PC, ξ_k.

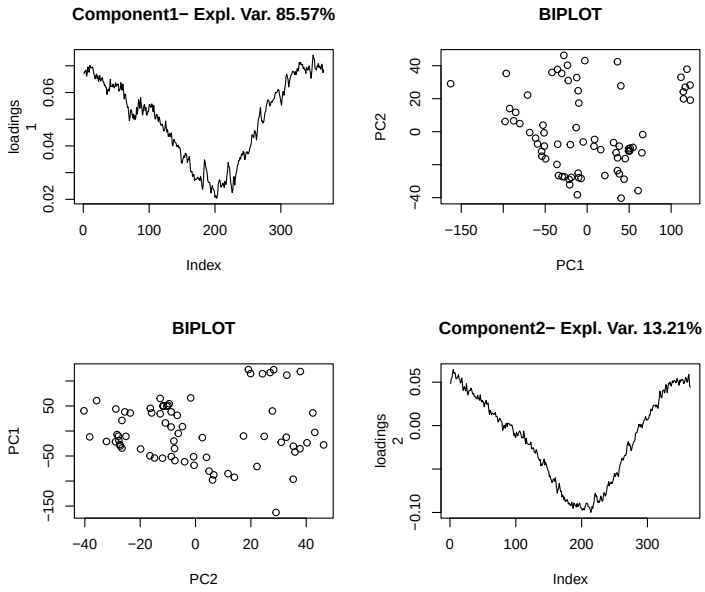
Example: Functional principal component analysis (FPCA)

```
y <- rowSums(aemet$logprec$data)
par(mfrow=c(1,3))
plot(aemet$temp)
plot((pc <- create.pc.basis(aemet$temp,l=1:2))$fdata.est,main="Temperature, 2 FPC")
pc2 <- fdata2pc(aemet$temp)
plot(pc2$rotation, main="2 FPC basis", xlab="day")
```




```
summary(pc2)
```

```
## [1] TRUE
##
##      - SUMMARY:   fdata2pc  object   -
##
## -With 2  components are explained  98.78 %
##  of the variability of explicative variables.
##
## -Variability for each component (%):
## [1] 85.57 13.21
```



- Functional Partial Least Squares (FPLS) (Krämer and Sugiyama, 2011) The PLS components seek to maximize the covariance of $X(t)$ and y.

```
args(create.pls.basis)
args(fdata2pls)
```

An integrated version of these functions is coming.

Example: Exploratory analysis for spectrometric curves

Tecator dataset: 215 spectrometric curves of meat samples also with Fat, Water and Protein contents obtained by analytic procedures.

Tecator Goal: Explain the fat content through spectrometric curves.

```
data(tecator)
names(tecator)
```

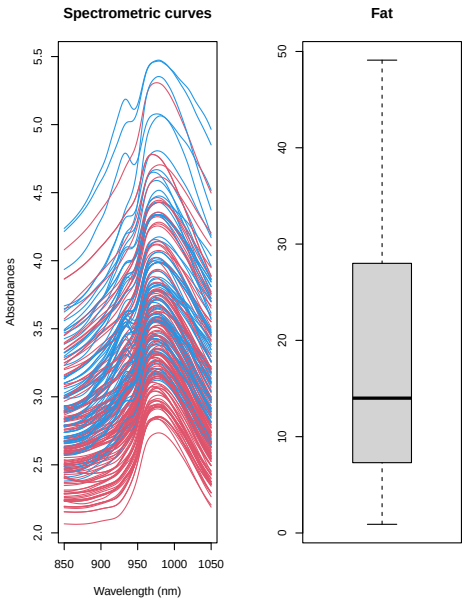
```
## [1] "absorp.fdata" "y"
```

```
names(tecator$y)
```

```
## [1] "Fat"     "Water"   "Protein"
```

As shown in tecator Figure, the plot of $X(t)$ against t is not necessarily the most informative and other semi-metric can allow to extract much information from functional variables.

The information about fat seems to be contained in the shape of the curves, so, the semimetric of derivatives could be preferred to $\mathcal{L}_2$. It is difficult to find the best plot given a particular functional dataset because the shape of graphics depends strongly on the proximity measure.

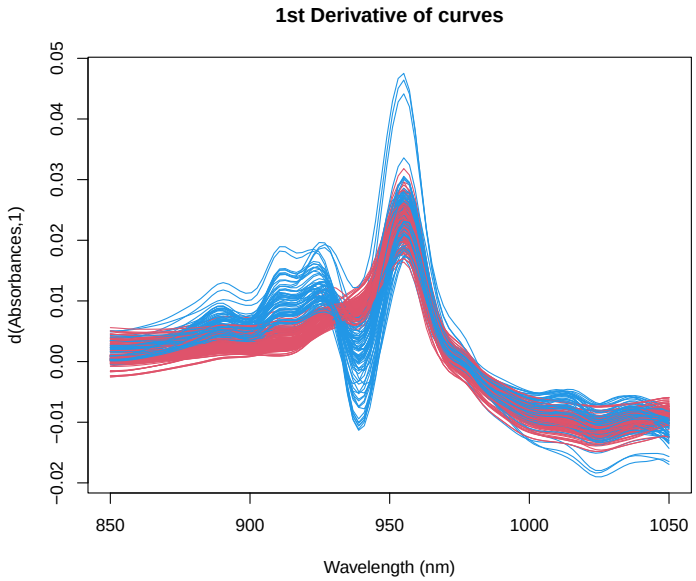


1.3.1 Derivatives

To compute derivatives there are several options (see `fdata.deriv` function):

- Raw Diferentiation: Only interesting when the number of discretization points are dense.
- Basis representation: Dierentiate a finite representation in a basis.
- Spline Interpolation: Perform a spline interpolation of given data points and use this interpolation for computing derivatives.
- Nonparametric Estimation: Use a Local Polynomial Estimator of the functional data.

Examle:



1.3.2 Computing distances, norms and inner products

Utilities for computing distances, norm and inner products are included in the package. Below are described those but of course, many others can be built with the only restriction that the first two arguments correspond to class –fdata–. In addition, the procedures of the package **fda.usc** that contains the argument –metric– allow the use of metric or semi-metric functions as shown in the following sections.

- Distance between functional elements, `metric.lp`: $d(X(t), Y(t)) = \|X(t) - Y(t)\|$ with a norm $\|\cdot\|$
- Norm `norm.fdata`: $\|X(t)\| = \sqrt{\langle X(t), X(t)\rangle}$
- Inner product `inprod.fdata`: $\langle x, y\rangle = (1/4)(\|x + y\|^2 - \|x - y\|^2)$.

fda.usc package collects several metric and semi-metric functions which allow to extract as much information possible from the functional variable.

Option 1: If the data is in a Hilbert space, represent your data in a basis, and compute all the things accordingly. From a practical point of way, the representation of a functional datum must be approximated by a finite number of terms.

- `semimetric.basis()`

Option 2: Approximates all quantities using numerical approximations (Valid for Hilbert and non-Hilbert spaces). This usually involves numerical approximations of integrals, derivatives and so on. A collection of semi-metrics proposed by (Ferraty and Vieu, 2006) are also included in the package.

- `semimetric.pca()`, based on the Principal Components.
- `semimetric.mplsr()`, based on the Partial Least Squares.

- `semimetric.deriv()`, based on B-spline representation.
- `semimetric.hshift()`, measure the horizontal shift effect.
- `semimetric.fourier()`, based on ther Fourier representation

Example of computing norms

Consider $X(t) = t^2; t \in [0, 1]$. In this case $||X_1|| = \sqrt{1/5} = 0.4472136$ and $||X_2|| = 1/3$

```
t = seq(0, 1, len = 101)
x2 = t^2
x2 = fdata(x2, t)
(c(norm.fdata(x2), norm.fdata(x2, lp = 1))) # L2, L1 norm
```

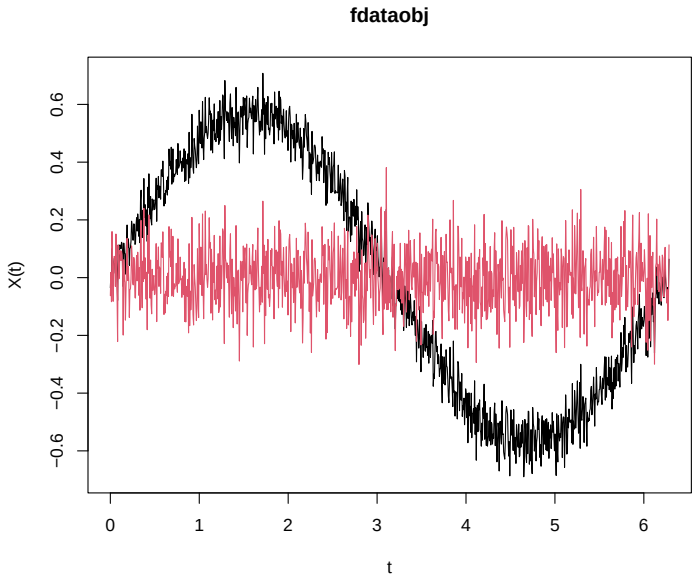
```
## [1] 0.4472509 0.3333500
```

```
f1 = fdata(rep(1, length(t)), t)
f2 = fdata(sin(2 * pi * t), t) * sqrt(2)
f3 = fdata(cos(2 * pi * t), t) * sqrt(2)
f4 = fdata(sin(4 * pi * t), t) * sqrt(2)
f5 = fdata(cos(4 * pi * t), t) * sqrt(2)
fou5 = c(f1, f2, f3, f4, f5) # Fourier basis is Orthonormal
round(inprod.fdata(fou5), 5)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
## [5,]    0    0    0    0    1
```

Example of computing distances

```
set.seed(1:4)
p<-1001
r<-rnorm(p,sd=.1)
x<-seq(0,2*pi,length=p)
fx<-fdata((sin(x)+r)/sqrt(pi),x)
fx0<-fdata(r,x)
plot(c(fx,fx0))
```




```
metric.lp(fx,fx0)[1,1]
```

```
## [1] 1.003144
```

```
semimetric.basis(fx,fx0,nbasis1=5,nbasis2=5)[1,1]
```

```
## [1] 0.9927386
```

```
semimetric.basis(fx,fx0,nbasis1=11,nbasis2=111)[1,1]
```

```
## [1] 0.9992571
```

```
semimetric.fourier(fx,fx0,nbasis=5)[1,1]
```

```
## [1] 0.9972634
```

```
semimetric.fourier(fx,fx0,nbasis=11)[1,1]
```

```
## [1] 0.9992104
```

```
semimetric.deriv(fx,fx0,nderiv=0,nknot=5)[1,1]
```

```
## [1] 0.9934345
```

```
semimetric.deriv(fx,fx0,nderiv=0,nknot=11)[1,1]
```

```
## [1] 0.9984524
```

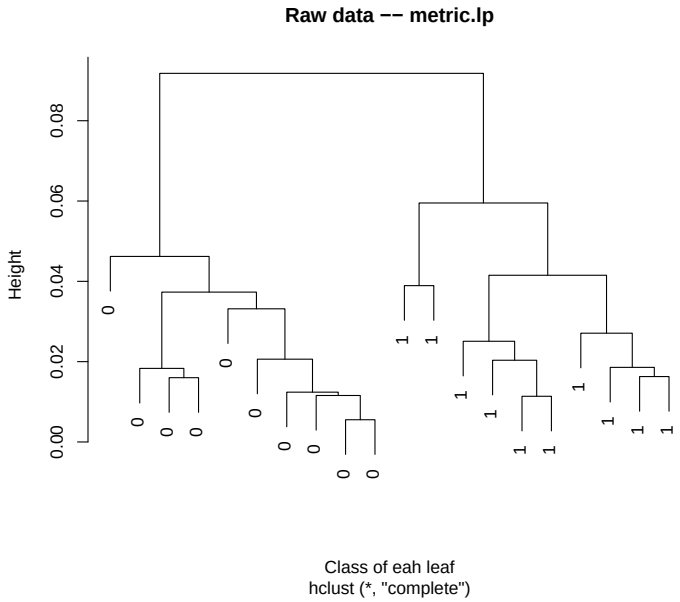
```
integrate(function(x){(sin(x)/sqrt(pi))^2},0,2*pi)
```

```
## 1 with absolute error < 7.3e-10
```

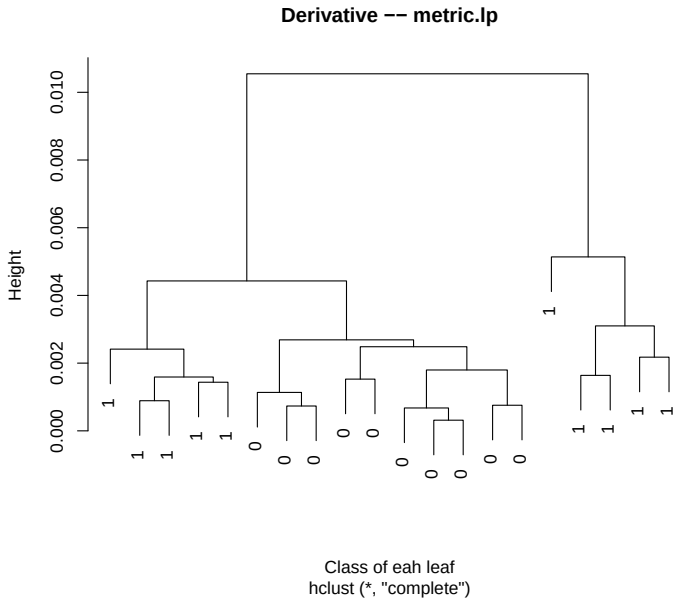
Example of semi-metric as classification rule

```
data(tecator)
set.seed(4:1)
#ind<-sample(215,30)
#xx<-tecator$absorp[ind]

yy<-ifelse(tecator$y$Fat<20,0,1)
ind<-c(sample(which(yy==0),10),sample(which(yy==1),10))
xx<-fdata.deriv(tecator$absorp[ind])
yy<-yy[ind]
```

```
par(mfrow=c(1,1))
d1<-as.dist(metric.lp(xx),T,T)
c1<-hclust(d1)
c1$labels=yy
plot(c1,main="Raw data -- metric.lp",xlab="Class of eah leaf")
```



Try with the derivative of the curves:



Extension: Combining the information of two components using a p-dimensional metric such as, for example, the Euclidean:

$$m\left(\left(x_0^1,\ldots,x_0^p\right),\left(x_i^1,\ldots,x_i^p\right)\right) := \sqrt{m_1\left(x_0^1,x_i^1\right)^2+\cdots+m_p\left(x_0^p,x_i^p\right)^2}$$

where m_i denotes the metric in the i-component of the product space, see `fda.usc:::metric.ldata` code.

It is important here to ensure that the different metrics of the spaces have similar scales to avoid one single component dominating the overall distance.

```
range(d1)
```

```
## [1] 0.005531143 0.091797359
```

```
range(d2)
```

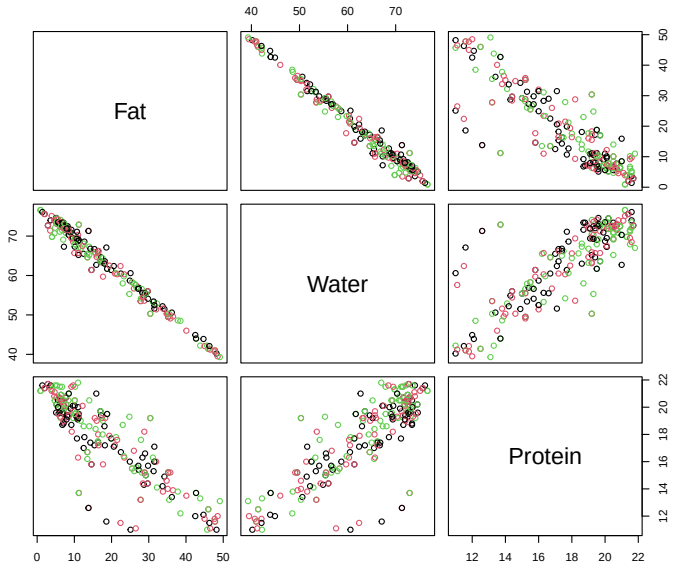
```
## [1] 0.0003143458 0.0105466510
```

NOTE: Dependent data in tecator dataset?

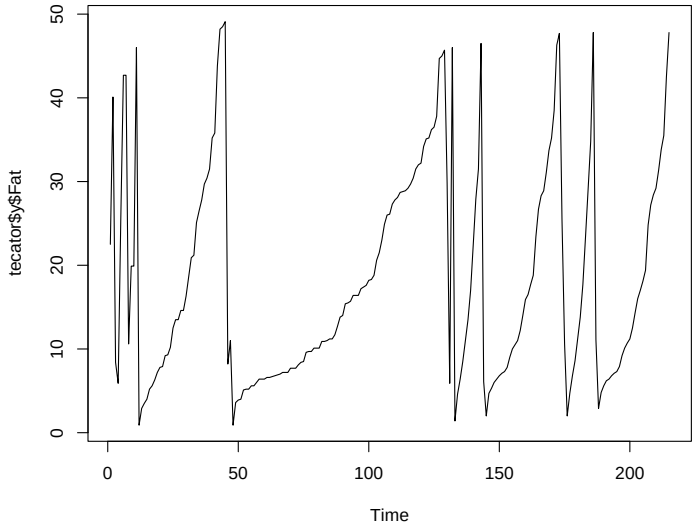
```
ar(tecator$y$Fat)
```

```
##
## Call:
## ar(x = tecator$y$Fat)
##
## Coefficients:
##      1       2
## 0.6552  0.1161
##
## Order selected 2  sigma^2 estimated as  72.82
```

```
plot(tecator$y,col=1:3)
```




```
par(mfrow=c(1,1))
ts.plot(tecator$y$Fat)
```



```
#barplot(t(tecator$y))
```

1.4 Correlation Distances

The correlation Distances characterizes independence between vectors of arbitrary finite dimensions. Recently, in (Székely and Rizzo, 2013), a bias-corrected version is considered and a test of independence developed.

```
data(tecator)
xx<-tecator$absorp
xx.d2<-fdata.deriv(xx,nderiv=2)
dcor.xy(xx,xx.d2)
```

```
##
##  dcor t-test of independence
##
## data:  D1 and D2
## T = 35.933, df = 22789, p-value < 2.2e-16
## sample estimates:
## Bias corrected dcor
##           0.2315581
```

```
d1<-metric.lp(xx)
d2<-metric.lp(xx.d2)
bcdcor.dist(d1,d2)
```

```
## [1] 0.2315581
```

```
Fat <-tecator$y[,"Fat",drop=FALSE]
d3<-metric.dist(Fat)
bcdcor.dist(d1,d3)
```

```
## [1] 0.1963254
```

```
bcdcor.dist(d2,d3)
```

```
## [1] 0.9147076
```

```
# Functional / Factor
Fat.cut<-as.matrix(cut(Fat[,1],labels=1:4,breaks=4))
dcor.xy(Fat.cut,xx.d2)
```

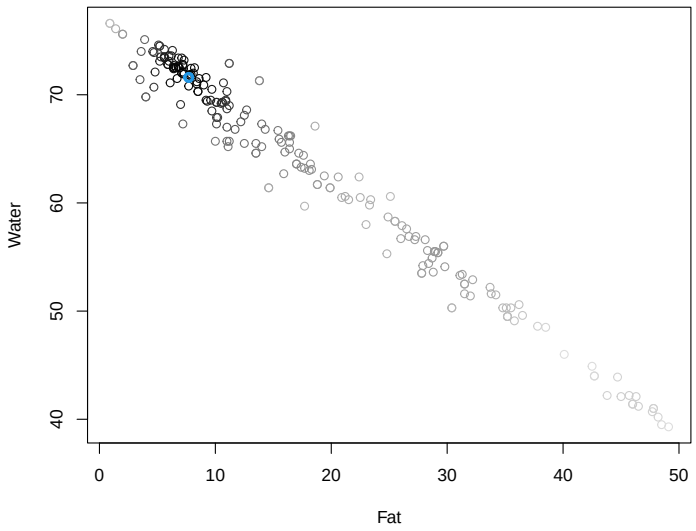
```
##
##  dcor t-test of independence
##
## data:  D1 and D2
## T = 252.57, df = 22789, p-value < 2.2e-16
## sample estimates:
## Bias corrected dcor
##           0.8583661
```

1.4.1 Depth for functional data

Depth (in univariate or multivariate context) is a statistical tool that provides a center-outward ordering of data points. This ordering can be employed to define location measures (and by oposition outliers). Also, some measures can be defined as maximizers of a particular depth.

- Mahalanobis Depth (Mean) `mdepth.MhD`
- Halfspace Depth, also known as Tukey Depth (Median) `mdepth.HS`
- Convex Hull Peeling Depth (Mode)
- Oja Depth
- Simplicial Depth `mdepth.SD`
- Likelihood Depth (Mode) `mdepth.LD`

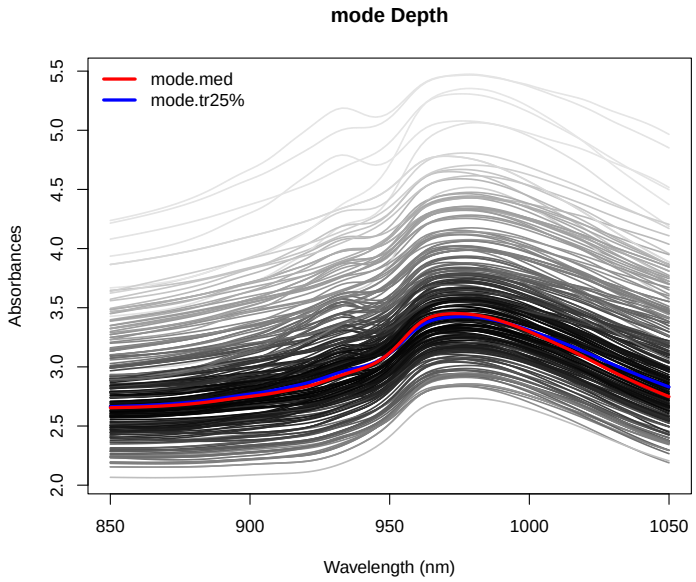
```
y12<-tecator$y[,1:2]
dep<-mdepth.LD(x=y12,scale=T)$dep
plot(y12,col=grey(1-dep))
points(y12[which.max(dep),],col=4,lwd=3)
```



Many of the preceding depth functions defined in a multivariate contex cannot be applied to functional data. In any case, given a depth for functional data.

- The deepest point is a location measure and based on the depth function can have a dierent interpretation: Mean, Median, Mode …
- Computing the depths of the whole sample and ordering them (in decreasing order) gives a rank from the most central point ($x_{[1]}$) to the most outlying one ($x_{[n]}$). So, those points with larger rank are the candidates to be outliers w.r.t. the data cloud (in the sense of the depth function).
- Summarizing the $(1-\alpha)$ deepest points could lead also to a robust location measure.
- Fraiman-Muniz Depth (Fraiman and Muniz, 2001) `depth.FM`
- Modal Depth (Cuevas et al., 2007) `depth.mode`
- Random Projection Depth (Cuevas et al., 2007) `depth.RP`
- Random Tukey Depth (Cuesta-Albertos and Nieto-Reyes, 2008) `depth.RT`
- Band Depth (López-Pintado and Romo, 2009) `depth.MBD` (private function)

```
x<-tecator$absorp
depth.mode(x,draw=TRUE)
```

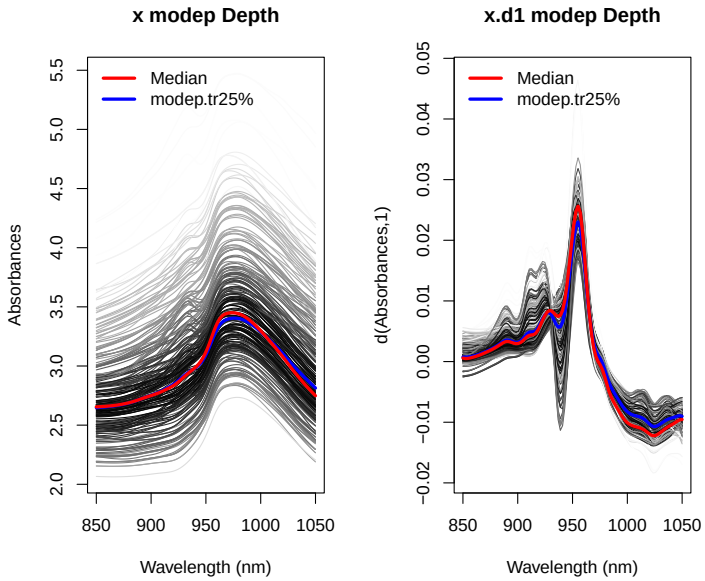


1.4.2 Depth (and distances) for multivariate functional data (Cuesta-Albertos et al., 2017)

Modify the procedure to incorporate the extended information.

- Fraiman–Muniz: Compute a multivariate depth marginally. `depth.FMp`
- Modal depth: Use a new distance between data (for derivatives, for example, the Sobolev metric). `depth.FMp`
- Random Projection: Consider a multivariate depth to be applied to the dierent projections (also the random projection method could be applied twice). `depth.RPp`

```
x.d1<-fdata.deriv(x)
depth.modep(list(x=x,x.d1=x.d1),draw=T)
```

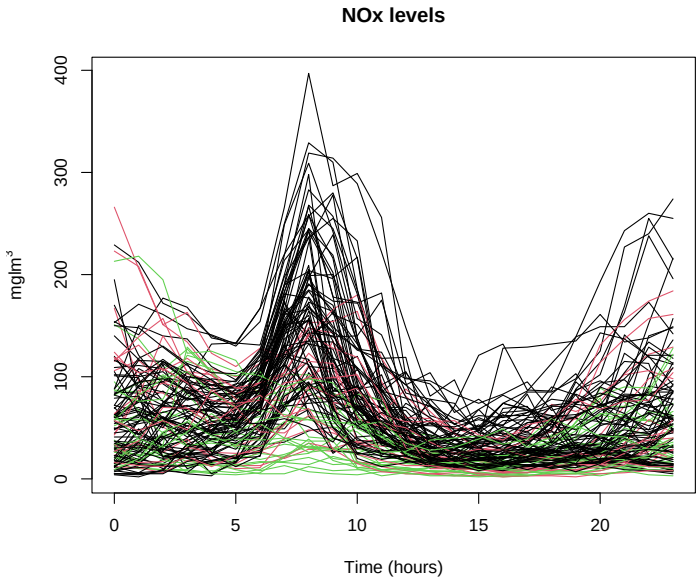


Example poblenou dataset

Hourly levels of nitrogen oxides in Poblenou (Barcelona). This dataset has 127 daily records (2005/01/06-2005/06/26).

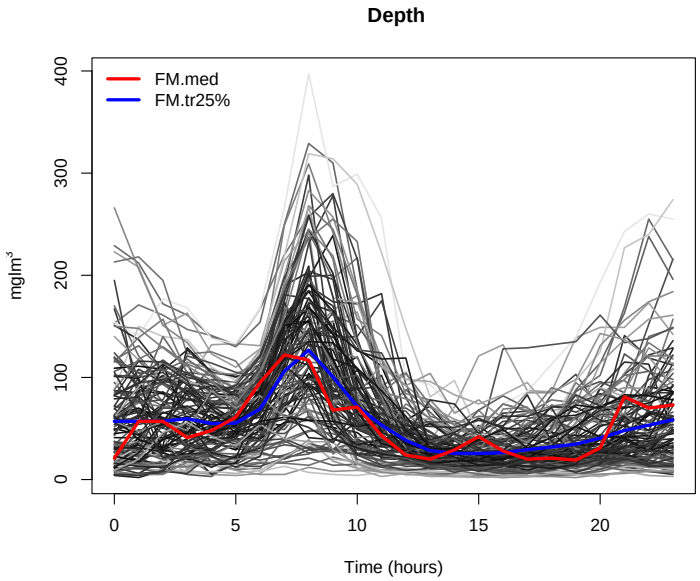
Objective: Explain the diferences in NOx levels as a function of day.

```
data(poblenou)
dayw = ifelse(poblenou$df$day.week == 7 | poblenou$df$day.festive ==1, 3, ifelse(poblenou$df$day.
plot(poblenou$nox,col=dayw)
```



The $\mathcal{L}_2$ space (distance is area between curves) seems appropriate although other possibilities could be take into account ($\mathcal{L}_1$, for example).

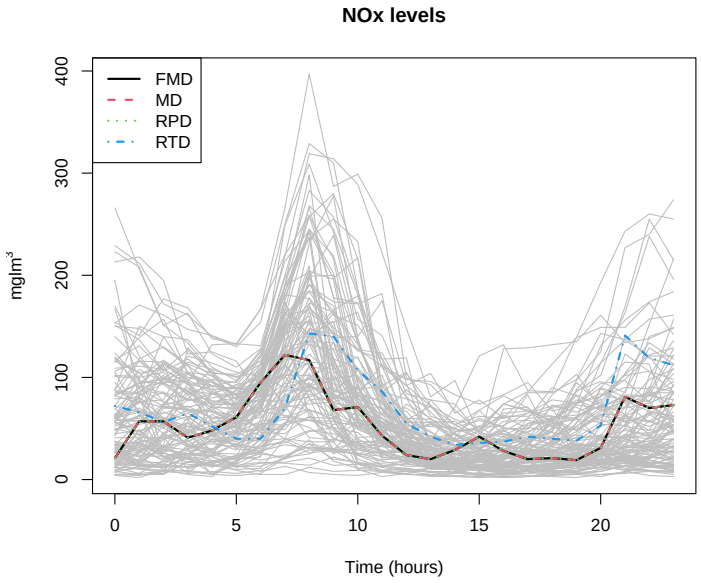
```
fmd = depth.FM(poblenou$nox,draw=T)
```



```
md =  depth.mode(poblenou$nox)
rpd = depth.RP(poblenou$nox, nproj = 50)
rtd = depth.RT(poblenou$nox)
print(cur <- c(fmd$lmed, md$lmed, rpd$lmed, rtd$lmed))
```

```
## 2005-05-06 2005-05-06 2005-05-06 2005-03-04
##         63         63         63         10
```

```
plot(poblenou$nox,col="grey")
lines(poblenou$nox[cur], lwd = 2, lty = 1:4, col = 1:4)
legend("topleft", c("FMD", "MD", "RPD", "RTD"), lwd = 2, lty = 1:4,col = 1:4)
```



1.4.3 Outliers detection

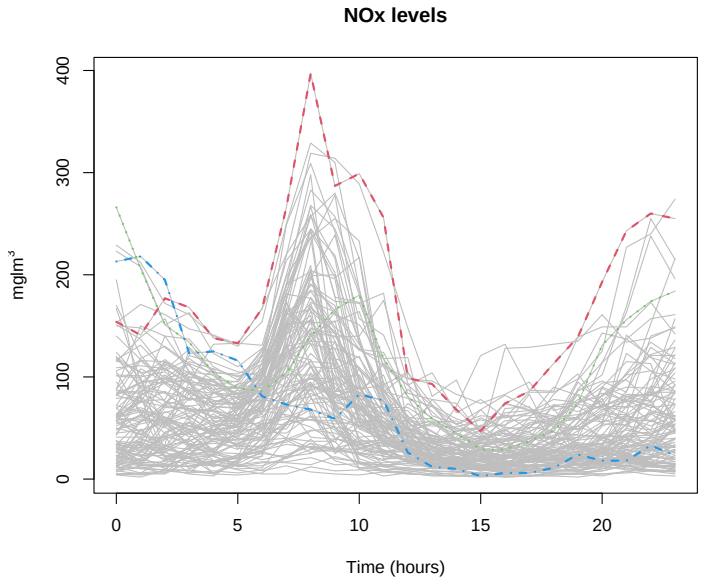
There is no general accepted definition of outliers in Functional data so, we define outlier as a datum generated from a dierent process than the rest of the sample with the following characteristics Its number in the sample is unknown but probably low.

An outlier will have low depth and it will be an outlier in the sense of the depth used.

```
md1 = depth.mode(lab <- poblenou$nox[dayw == 1]) #Labour days
md2 = depth.mode(sat <- poblenou$nox[dayw == 2]) #Saturdays
md3 = depth.mode(sun <- poblenou$nox[dayw == 3]) #Sundays/Festive
rbind(poblenou$df[dayw == 1, ][which.min(md1$dep), ], poblenou$df[dayw ==
2, ][which.min(md2$dep), ], poblenou$df[dayw == 3, ][which.min(md3$dep),])
```

```
##          date day.week day.festive
## 22 2005-03-18        5           0
## 57 2005-04-30        6           0
## 58 2005-05-01        7           0
```

```
plot(poblenou$nox,col="grey")
lines(poblenou$nox[c(22,57,58),], lwd = 2, lty = 2:4, col = 2:4)
```




```
# Method for detecting outliers, see Febrero-Bande, 2008
#out1<-outliers.depth.trim(poblenou$nox[dayw == 1],nb=100)$outliers
```

Two procedures for detecting outliers are implemented in the package. (Febrero-Bande et al., 2008)

- Detecting outliers based on trimming: `outliers.depth.trim()`
- Detecting outliers based on weighting: `outliers.depth.pond()`

```
# Method for detecting outliers, see Febrero-Bande, 2008
#out1<-outliers.depth.trim(poblenou$nox[dayw == 1],nb=100)$outliers
#out2<-outliers.depth.pond(poblenou$nox[dayw == 1],nb=100)$outliers
```

1.5 References

Capítulo 2

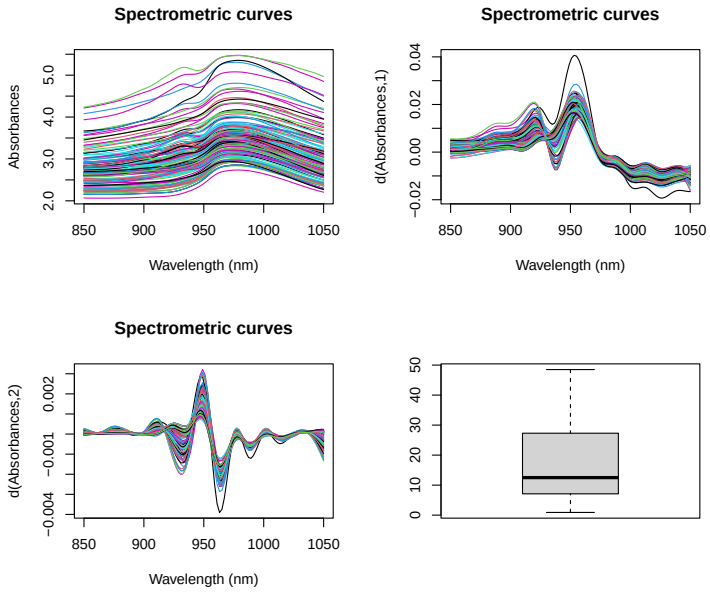
Functional Regression Model

Regression models are those techniques for modeling and analyzing the relationship between a dependent variable and one or more independent variables. When one of the variables have a functional nature, we have functional regression models.

This section is devoted to all the functional regression models where the response variable is scalar and at least, there is one functional covariate.

For illustration, we will use the Tecator dataset to predict the fat contents from The explanatory variables to introduce in the models are:p The curves of absorbance $X(t)$ as functional data or one of its two first derivatives $(X.d1, X.d2)$ and/or Water content as real variable.

```
library(fda.usc)
data(tecator)
absorp<-tecator$absorp
ind<-sample(215,129) #ind = 1:129
tt = absorp[["argvals"]]
y = tecator[["y"]]$Fat[ind]
X = absorp[ind, ]
X.d1 = fdata.deriv(X, nbasis = 19, nderiv = 1)
X.d2 = fdata.deriv(X, nbasis = 19, nderiv = 2)
par(mfrow=c(2,2))
plot(X)
plot(X.d1)
plot(X.d2)
boxplot(y)
```



In the following sections, regression methods implemented –fda.usc– pacakge in the package are presented one by one and illustrated with examples for estimating the **Fat** content of the Tecator dataset.

2.1 Functional linear model (FLR) with basis representation

Supose that $\mathcal{X} \in \mathcal{L}_2(T)$ and $y \in \mathbb{R}$. Assume also that $\mathbb{E}[\mathcal{X}(t)] = 0, \forall t \in [0, T]$ and $\mathbb{E}[y] = 0$.

The FLM states that

$$y = \langle \mathcal{X}, \beta \rangle + \varepsilon = \int_T X(t)\beta(t)dt + \varepsilon$$

where $\beta \in \mathcal{L}_2(T)$ and ε is the errror term.

One way of estimating β, it is representing the parametmer (and $\mathcal{X}$) in a $\mathcal{L}_2$-basis in the following way:

$$\beta(t) = \sum_k \beta_k \theta_k(t), \mathbf{X}(t) = \sum_k c_i \psi_k(t)$$

- `fregre.basis()` fucntion uses fixed basis: B–spline, Fourier, etc. Ramsay and Silverman (2005b), Cardot et al. (1999))

The next code illustrates how to estimate the fat contents using a sample of absorbances curves.

```
rangett <- X$rangeval
basis1 = create.bspline.basis(rangeval = rangett, nbasis = 17)
basis2 = create.bspline.basis(rangeval = rangett, nbasis = 7)
res.basis0 = fregre.basis(X, y, basis.x = basis1, basis.b = basis2)
res.basis1 = fregre.basis(X.d1, y, basis.x = basis1, basis.b = basis2)
res.basis2 = fregre.basis(X.d2, y, basis.x = basis1, basis.b = basis2)
res.basis0$r2;res.basis1$r2;res.basis2$r2
```

```
## [1] 0.9385496
```

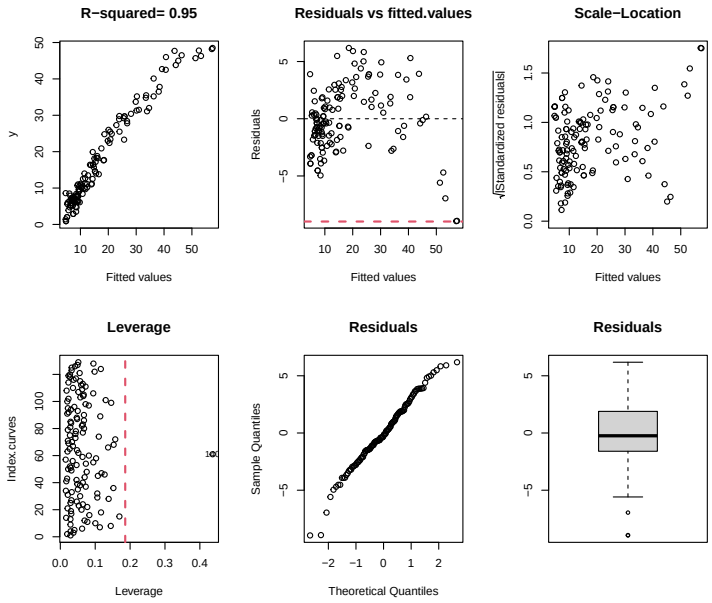
```
## [1] 0.9360606
```

```
## [1] 0.9518397
```

```
summary(res.basis2)
```

```
##  *** Summary Functional Data Regression with representation in Basis ***
##
## Call:
## fregre.basis(fdataobj = X.d2, y = y, basis.x = basis1, basis.b = basis2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.9498 -1.5962 -0.2428  1.8891  6.1841
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.759e+01  2.636e-01  66.723  < 2e-16 ***
## X.d2.bspl4.1 -1.294e+04  3.849e+03  -3.361 0.001040 **
## X.d2.bspl4.2  9.261e+03  2.901e+03   3.192 0.001801 **
## X.d2.bspl4.3 -1.215e+03  1.426e+03  -0.852 0.395973
## X.d2.bspl4.4  9.804e+02  1.092e+03   0.897 0.371275
## X.d2.bspl4.5 -1.599e+03  1.126e+03  -1.420 0.158232
## X.d2.bspl4.6  6.896e+03  1.802e+03   3.826 0.000207 ***
## X.d2.bspl4.7 -7.985e+03  1.438e+03  -5.554 1.69e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.994 on 121 degrees of freedom
## Multiple R-squared:  0.9518, Adjusted R-squared:  0.9491
## F-statistic: 341.6 on 7 and 121 DF,  p-value: < 2.2e-16
```

```
##
## -Names of possible atypical curves: No atypical curves
## -Names of possible influence curves: 140
```



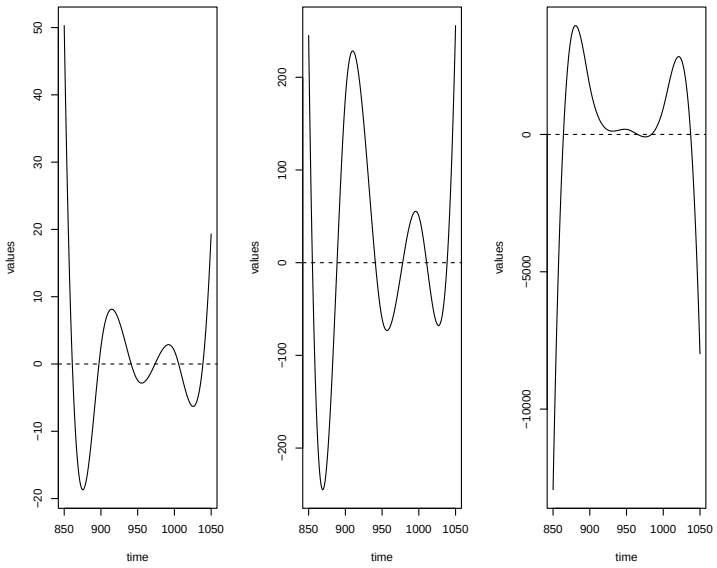

```
par(mfrow=c(1,3))
plot(res.basis0$beta.est)
```

```
## [1] "done"
```

```
plot(res.basis1$beta.est)
```

```
## [1] "done"
```

```
plot(res.basis2$beta.est)
```



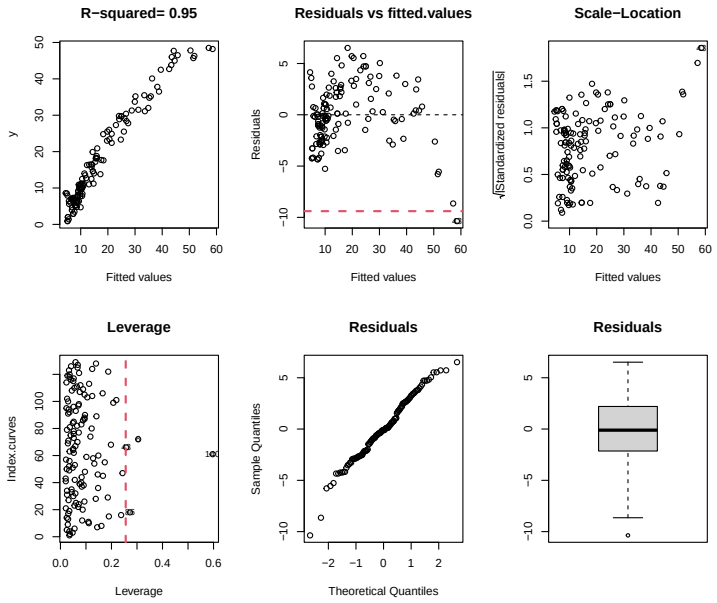

```
## [1] "done"
```

The choice of the appropiate basis (and the number of basis elements) becomes now in a crucial step:

```
res.basis.cv = fregre.basis(X, y)
summary(res.basis.cv)
```

```
##  *** Summary Functional Data Regression with representation in Basis ***
##
## Call:
## fregre.basis(fdataobj = X, y = y)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.3657  -2.1421  -0.1094   2.1998   6.5177
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   17.5884     0.2757  63.790   <2e-16 ***
## X.bspl4.1   -200.0776    93.7095  -2.135   0.0348 *
## X.bspl4.2    242.9587   112.0151   2.169   0.0321 *
## X.bspl4.3   -133.6675    72.6212  -1.841   0.0682 .
## X.bspl4.4     23.7801    33.6901   0.706   0.4817
```

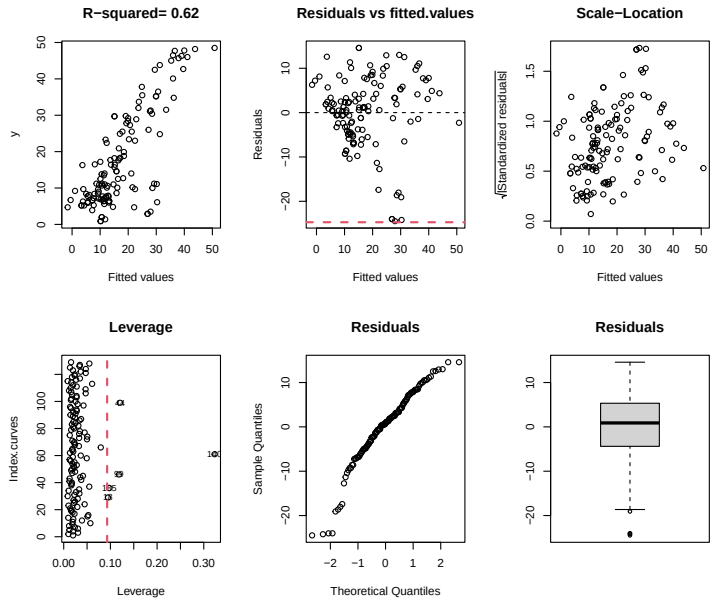
```
## X.bspl4.5     14.2783    19.4473   0.734   0.4643
## X.bspl4.6    -23.9989    16.8771  -1.422   0.1577
## X.bspl4.7     46.5994    27.9568   1.667   0.0982 .
## X.bspl4.8   -104.1480    65.9443  -1.579   0.1169
## X.bspl4.9    154.0717   108.4606   1.421   0.1581
## X.bspl4.10  -123.4221    94.1397  -1.311   0.1924
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.132 on 118 degrees of freedom
## Multiple R-squared:  0.9486, Adjusted R-squared:  0.9443
## F-statistic: 217.8 on 10 and 118 DF,  p-value: < 2.2e-16
##
## -Names of possible atypical curves: 43
## -Names of possible influence curves: 86 140 43 6
```



- Functional Principal Components (FPC).(Cardot et al., 1999), `fregre.pc()`

```
x<-X
basis.pc0 = create.pc.basis(X,1:3)
res.pc1 = fregre.pc(X, y, basis.x = basis.pc)
summary(res.pc1)
```

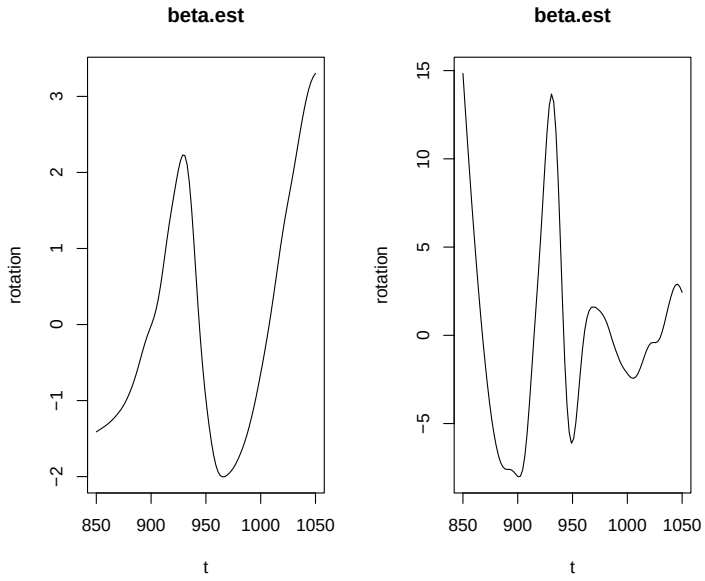
```
##  *** Summary Functional Data Regression with Principal Components ***
##
## Call:
## fregre.pc(fdataobj = X, y = y, basis.x = basis.pc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.471  -4.389   0.886   5.328  14.585
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.58837    0.72555  24.241   <2e-16 ***
## PC1           0.99689    0.09838  10.133   <2e-16 ***
## PC2          -1.50819    1.18685  -1.271    0.206
## PC3         -21.84347    1.90208 -11.484   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.241 on 125 degrees of freedom
## Multiple R-squared:  0.6231, Adjusted R-squared:  0.614
## F-statistic: 68.88 on 3 and 125 DF,  p-value: < 2.2e-16
##
##
## -With 3 Principal Components is  explained  99.47 %
##  of the variability of explicative variables.
##
## -Variability for each  principal components -PC- (%):
##   PC1   PC2   PC3
## 98.54  0.66  0.26
## -Names of possible atypical curves: No atypical curves
## -Names of possible influence curves: 18 185 99 140 44
```




```
res.pc2 = fregre.pc.cv(X, y)
summary(res.pc2)
```

```
##            Length Class     Mode
## fregre.pc  19     fregre.fd list
## pc.opt      5     -none-    numeric
## lambda.opt  1     -none-    numeric
## PC.order    8     -none-    numeric
## MSC.order   8     -none-    numeric
```

```
par(mfrow=c(1,2))
plot(res.pc1$beta.est)
plot(res.pc2[[1]]$beta.est)
```



2.2 FLM with functional and non functional covariates

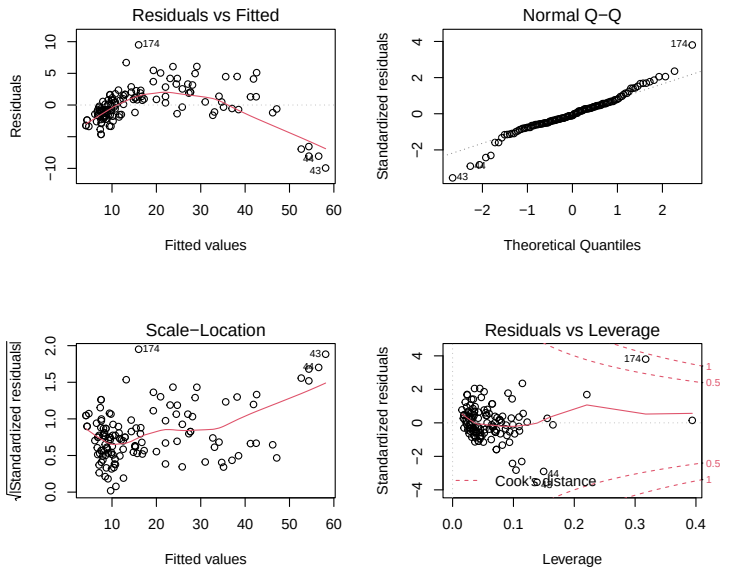
$$E(y) = \alpha + \mathbf{Z}\beta + \sum_{q=1}^{Q} \langle \mathcal{X}^q(t), \beta_q(t) \rangle$$

where $\{\mathcal{X}_q(t)\}_{q=1}^{Q}$ are function covariates and $\mathbf{Z} = \{Z_j\}_{j=1}^{J}$ the non–functional covariates.

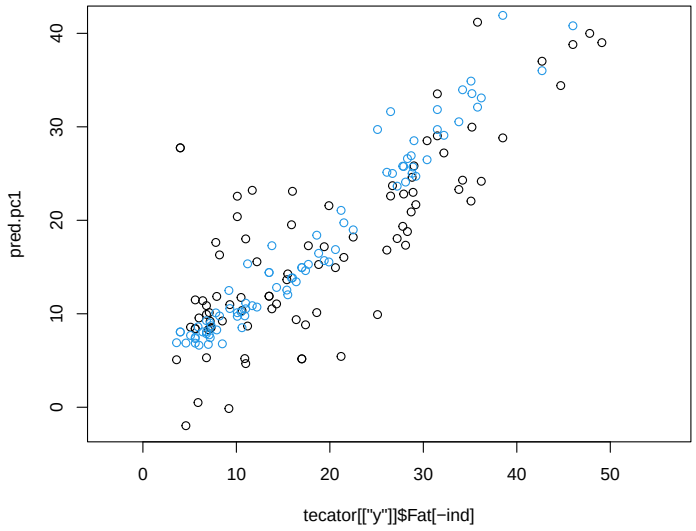
```
dataf = as.data.frame(tecator[["y"]][ind,]) # Fat, Protein, Water
basis.pc2 = create.pc.basis(X.d2,1:4)
basis.x = list(X = basis.pc0, X.d2 =basis.pc2)
f = Fat ~ X+X.d2
ldata = list(df = dataf, X=X,X.d2=X.d2)
res.lm1 = fregre.lm(f, ldata, basis.x = basis.x)
f = Fat ~ Water+X.d2
res.lm2 = fregre.lm(f, ldata, basis.x = basis.x)
```

```
##
## Call:
```

```
## lm(formula = pf, data = XX, x = TRUE)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9417 -1.6197 -0.2995  1.5864  9.4955
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    17.5884     0.2659  66.142  < 2e-16 ***
## X.PC1           0.1127     0.1023   1.102  0.27283
## X.PC2           7.1807     3.2173   2.232  0.02746 *
## X.PC3         -19.8307     6.9644  -2.847  0.00518 **
## X.d2.PC1     3066.6836   563.3407   5.444 2.78e-07 ***
## X.d2.PC2     5507.7858  2668.4922   2.064  0.04115 *
## X.d2.PC3     1879.3468  1017.4538   1.847  0.06717 .
## X.d2.PC4    -2644.9925  3131.4915  -0.845  0.39998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.02 on 121 degrees of freedom
## Multiple R-squared:  0.951,  Adjusted R-squared:  0.9482
## F-statistic: 335.4 on 7 and 121 DF,  p-value: < 2.2e-16
```



2.2.1 Predict method for functional regression model



2.3 Other procedures

4. Other procedures

- Partial Least Squares (FPLS). `fregre.pls()`, (Krämer and Sugiyama, 2011, Preda and Saporta (2005))
- Penalized versions and parameter selection: `fregre.pc.cv`, `fregre.basis.cv`, `fregre.np.cv` (Febrero-Bande and Oviedo de la Fuente, 2012)
- F-test for the FLM with scalar response: `flm.Ftest`, `F-test` (García-Portugués et al., 2014)
- Goodness-of-fit test for the FLM with scalar response: `flm.test` (García-Portugués et al., 2014)
- Measures of influence in FLM with scalar response: `influence.fdata`,(Febrero-Bande et al., 2010)
- Beta parameter estimation by wild or smoothed bootstrap procedure: `fregre.bootstrap`
- FLM with a functional response: `fregre.basis.fr` (Chiou et al., 2004)

2.4 Non Linear Model (Ferraty and Vieu, 2006)

Supose $(\mathcal{X}, Y)$ are a pair of r.v. with $y \in \mathbb{R}$ where $\mathbb{E}$ is a semi-metric space. To predict the resonse Y with $\mathcal{X}$, the estimation is:

$$m(\mathcal{X}) = \mathbb{E}(Y|X = \mathcal{X})$$

, where the NW estimator is given by:

$$\widehat{m}(\mathcal{X}) = \frac{\sum_{i=1}^{n} Y_i K(d(\mathcal{X}, X_i)/h)}{\sum_{i=1}^{n} K(d(\mathcal{X}, X_i)/h)}$$

where K is an asymmetric kernel function and h is the bandwidth parameter.

2.5 Semi Linear Model (Aneiros-Pérez and Vieu, 2006)

Let $(\mathcal{X}, \mathbf{Z}, y)$ with $y \in \mathbb{R}$ (response), $\mathcal{X} \in \mathbb{E}$ (functional) and $\mathbf{Z} \in \mathbb{R}^p$ (MV covariates).

$$y = Z + m(X) + \varepsilon$$

Arguments for fregre.np() and fregre.plm() function

- –Ker–: type of asymmetric kernel function, by default asymmetric normal kernel (cosine, epanechnicov, quadratic,….).
- –metric–: type of metric or semimetric. –type.S–: type of smoothing matrix **S**: `S.NW`, `S.LLR`, `S.KNN`.

```
tecator<-list("df"=tecator$y,"absorp.fdata"=tecator$absorp.fdata)
X=tecator$absorp.fdata
y<-tecator$df$Fat

np<-fregre.np(X, y, metric = semimetric.deriv, nderiv = 1,type.S = S.KNN)
```

Again, it has also implemented the function `fregre.np.cv` to estimate the smoothing parameter h by the validation criteria.

```
np<-fregre.np(X, y, metric = semimetric.deriv, nderiv = 1,type.S = S.KNN)
np.cv<-fregre.np.cv(X, y, metric = semimetric.deriv, nderiv = 1,type.S = S.KNN,h=c(3:9)
c(np$h.opt,np.cv$h.opt)
```

```
## [1] 6 3
```

```
c(np$r2,np.cv$r2)
```

```
## [1] 0.8687145 0.9438220
```

2.6 Generalized Linear Models

One natural extension of LM model is the generalized functional linear regression model (GFLM) which allows various types of the response. In the GLM framework it is generally assumed that $y_i|X_i$ can be chosen within the set of distributions belonging to the exponential family (Muller and Stadtmuller, 2005).

In Generalized Functional Linear Model (FGLM), The scalar response y(belonging to a Exponential Family PDF) is estimated by functional $\{\mathcal{X}_q(t)\}_{q=1}^{Q}$ and also non–functional $\mathbf{Z} = \{Z_j\}_{j=1}^{J}$ covariates by:

$$E(y) = g^{-1}\left(\alpha + \mathbf{Z}\beta + \sum_{q=1}^{Q} \langle \mathcal{X}^q(t), \beta_q(t)\rangle\right)$$

where $g()$ is the inverse link function.

Example of logistic regression

In logistic regression, the probability, π_i , of the occurrence of an event, $Y_i = 1$, rather than the event $Y_i = 0$, conditional on a vector of covariates $\mathcal{X}_i(t)$ is expressed as:

$$p_i = \mathbb{P}[Y = 1|X_i(t) : t \in T] = \frac{+exp\left\{\alpha + \int_T X_i(t)\beta(t)dt\right\}}{1 + exp\left\{\alpha + \int_T X_i(t)\beta(t)dt\right\}} \ , i = 1, \ldots, n$$

with ϵ are the independent errors with zero mean.

```
data(tecator)
names(tecator)[2]<-"df"
tecator$df$fat15<-ifelse(tecator$df$Fat<15,0,1)
tecator$absorp.d2=fdata.deriv(tecator$absorp.fdata,nderiv=2)
res.glm<-fregre.glm(fat15 ~ absorp.d2,data=tecator,family=binomial())
#summary(a)
yfit<-ifelse(res.glm$fitted.values<.5,0,1)
table(tecator$df$fat15,yfit)
```

```
##    yfit
##       0   1
##   0 109   3
##   1   2 101
```

2.7 Generalized Functional Additive Model

1. Generalized Functional Spectral Additive Linear Model (FGSAM), (Müller and Yao, 2012)

$$E(y) = g^{-1}\left(\alpha + \sum_{j=1}^{J} f_j\left(\mathbf{Z}^j\right) + \sum_{q=1}^{Q} s_q\left(\mathcal{X}_i^q(t)\right)\right)$$

where $f(\cdot), s(\cdot)$ are the smoothed functions.

```
res.gsam<-fregre.gsam(fat15~ s(absorp.d2),data=tecator,family=binomial())
yfit<-ifelse(res.gsam$fitted<.5,0,1)
table(tecator$df$fat15,yfit)
```

```
##    yfit
##       0   1
##   0 112   0
##   1   0 103
```

2. Generalized Functional Kernel Additive Linear Model (FGKAM), (Febrero-Bande and González-Manteiga, 2013)

$$E(y) = g^{-1}\left(\alpha + \sum_{q=1}^{Q} \mathcal{K}\left(\mathcal{X}_i^q(t)\right)\right)$$

where $\mathcal{K}(\cdot)$ is the kernel estimator.

```
# tecator2<-tecator[-1]
# tecator$df$fat15 <- as.factor(tecator$df$fat15)
# res.gkam<-fregre.gkam(fat15 ~ absorp.d2,data=tecator2, family=binomial(),
#              control = list(maxit = 1))
# res.gkam
# yfit<-ifelse(res.gkam$fitted.values<.5,0,1)
# table(tecator$df$fat15,yfit)
```

2.8 Functional GLS model

See Oviedo de la Fuente et al. (2018) for more details about the below algorithm:

A. Jointly estimation (nlme package): Minimize for (β, θ) the GLS criteria, i.e,

$$\Psi(\beta, \theta) = (y - \langle X, \beta \rangle) \, \Sigma(\theta)^{-1} \, (y - \langle X, \beta \rangle)$$

B. Iterative Estimation: In multivariate case, Zivot and Wang (2006) show that estimation of β by $\widehat{\beta}_{ML}$ is equivalent to the iterative estimation of $\widehat{\beta}$ recomputed at each iteration by the update estimator of Σ.

1. Begin with a preliminary estimation of $\widehat{\theta} = \theta_0$. Compute $\widehat{W} = \Sigma(\theta_0)^{-1}$.
2. Estimate $b_\Sigma = (Z'\widehat{W}Z)^{-1}Z'\widehat{W}y$
3. Based on $\widehat{e} = (y - Zb_\Sigma)$, update $\widehat{\theta} = \rho(\widehat{e})$ where ρ depends on the dependence structure chosen.
4. Repeat steps 2 and 3 until convergence.

The generalized correlated cross-validation (GCCV) criterion is an extension to GCV within the context of correlated errors, Carmack et al. (2012). It is defined as follows:

$$GCCV(K_x, K_\beta, \mathbf{b}, \phi) = \frac{\sum_{i=1}^{n} \left(y_i - \widehat{y}_{i,\mathbf{b}}\right)^2}{\left(1 - \frac{tr(\mathbf{G})}{n}\right)^2}$$

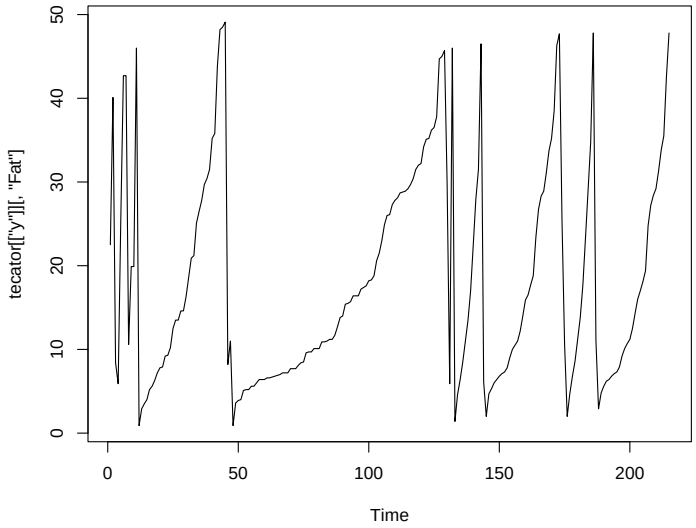
where $G = 2H\Sigma(\phi) - H\Sigma(\phi)H'$ takes into account the effect of the dependence, the trace of G is an estimation of the degrees of freedom consumed by the model and H is the hat matrix.

The important advantage of this criterion is that it is rather easy to compute because it avoids the need to compute the inverse of the matrix Σ. Even so, the complexity of the GLS criterion depends on the structure of Σ and it could sometimes be hard either to minimize or computationally expensive.

2.8.1 Dependent data example,

We use the `fregre.gls()` function that has the same arguments as the `fregre.lm()` function and: **correlation** argument, same functionality as in `gls()` and **criteria** argument, it require `GCCV.S()` function to calculate the GCCV score proposed by Carmack et al. (2012).

```
data(tecator)
ts.plot(tecator[["y"]][,"Fat"])
```




```
cor(tecator[["y"]][,"Fat",drop=F],tecator[["y"]][,"Water",drop=F])
```

```
##           Water
## Fat -0.9881002
```

```
cor(tecator[["y"]][,"Fat",drop=F],tecator[["y"]][,"Protein",drop=F])
```

```
##         Protein
## Fat -0.8608965
```

```
dcor.xy(tecator[["y"]][,"Fat",drop=F],tecator[["y"]][,"Water",drop=F])
```

```
##
##  dcor t-test of independence
##
## data:  D1 and D2
## T = 571.71, df = 22789, p-value < 2.2e-16
## sample estimates:
## Bias corrected dcor
##           0.9668619
```

```
dcor.xy(tecator[["y"]][,"Fat",drop=F],tecator[["y"]][,"Protein",drop=F])
```

```
##
##  dcor t-test of independence
##
## data:  D1 and D2
## T = 155.43, df = 22789, p-value < 2.2e-16
## sample estimates:
## Bias corrected dcor
##           0.7173448
```

```
x.d2<-fdata.deriv(tecator[["absorp.fdata"]],nderiv=2)
ldata=list("df"=tecator[["y"]],"x.d2"=x.d2)

res.gls=fregre.gls(Fat~x.d2, data=ldata, correlation=corAR1())
coef(res.gls[["modelStruct"]],F)
```

```
## corStruct.Phi
##     0.4942661
```

The previous model is restricted to a structure determined by `gls()` function of **nlme** The function `fregre.igls()` is presented as an alternative because it allows any type of dependence structures designed by the user.

The code bellow shows a simple use of iterative scheme (iGLS). In particular, we use a iGLS-AR($p = 1$) scheme for error estimation.

```
res.igls=fregre.igls(Fat~x.d2, data=ldata, correlation=list("cor.ARMA"=list()),control=list("p"=1
coef(res.igls[["corStruct"]][[1]])
```

```
##      ar1
## 0.488854
```

```
res.igls
```

```
##
## Call:
## list("fregre.basis")
##
## Coefficients:
##  (Intercept)  x.d2.bspl4.1  x.d2.bspl4.2  x.d2.bspl4.3  x.d2.bspl4.4
##        18.12       -608.20       6203.09      -8252.76       6271.43
## x.d2.bspl4.5
##     -7156.85
```

```
res.igls$corStruct
```

```
## $ar
##
## Call:
## arima(x = x, order = c(p, d, q), include.mean = FALSE, transform.pars = TRUE)
##
## Coefficients:
##          ar1
##       0.4889
## s.e.  0.0600
##
## sigma^2 estimated as 8.076:  log likelihood = -529.76,  aic = 1063.53
```

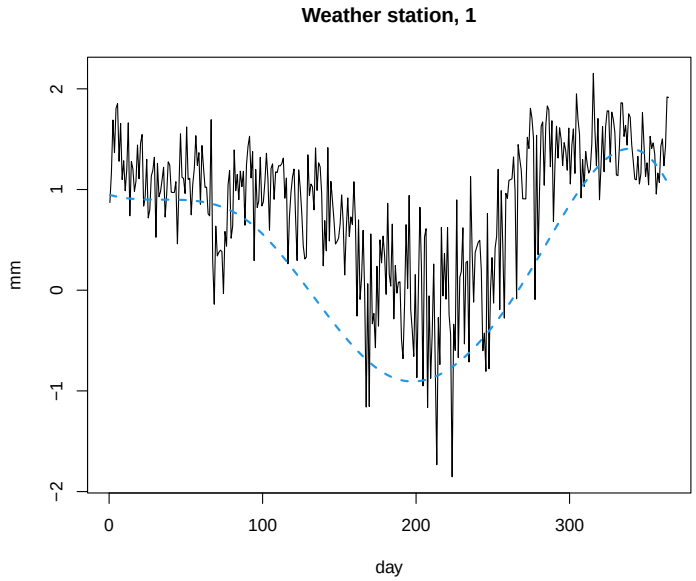
Both examples estimate an AR(1) with $\phi = 0.49$. Thus, the estimation and the prediction made with these models will be more accurate than the classical functional models in which it is assumed that the errors are independent.

2.8.2 Functional Response Model

Reference papers: Faraway (1997), Ferraty et al. (2012)

R expample of function `fregre.basis.fr()`

```
data(aemet)
log10precfdata<-aemet$logprec; tempfdata<-aemet$temp
res2<-fregre.basis.fr(tempfdata,log10precfdata)
i<-1

plot(log10precfdata[i],lty=1,main=paste0("Weather station, ",i))
lines(res2$fitted.values[i],lty=2,lwd=2,col=4)
```



2.8.3 Other Models:

- Functional Quantile Regession Model, see Kato et al. (2012), Cardot et al. (2005).
- Functional Single Index Model, see Ferraty et al. (2011).
- Functional Projection Pursuit Regression Model, see Ferraty et al. (2013).
- Functional Machine Learning methods (SVM, RPART, NNET, random Forest)

2.9 References

Capítulo 3

Functional Supervised Classification

This section describes the usage of functional classification using fda.usc package in R.

Let a sample $(\mathcal{X}, Y) \in E \times \mathbb{G} = 1, \cdots, G$.

Aim: How predict the class g of Y (categorical variable) given a functional variable $\mathcal{X}$

Bayes rule: Estimate the posterior probability of belonging to each group:

$$p_g(X) = \mathbb{P}(Y = g|\mathcal{X} = \chi) = \mathbb{E}(1_{Y=g}|\mathcal{X} = \chi)$$

The predicted class is given by the Bayes rule,

$$\hat{Y} = \arg\max_{g \in \mathbb{G}} \hat{p}_g(\chi)$$

The package allows the estimation of the groups in a training set of functional data by

- Logistic Classifier (linear model): `classif.glm`
- Logistic Classifier (additive model): `classif.gsam` and `classif.gkam`
- k-Nearest Neighbor Classifier: `classif.knn`
- Kernel Classifier: `classif.kernel`
- Distance Classifier: 'classif.distv
- Maximum Depth Classifier: `classif.depth`
- DD Clasisifier: `classif.DD`

```
library(fda.usc)
data(tecator)
x=tecator$absorp.fdata
tecator$y$Fat<-ifelse(tecator$y$Fat>20,1,0)

x.d1<-fdata.deriv(x)
dataf=as.data.frame(tecator$y)
ldata=list("df"=dataf,"x"=x,"x.d1"=x.d1)
ycat<-ldata$df$Fat
```

3.1 Logistic Regression Model (GLM): `classif.glm`

As a particular case of Generalized Linear Models, the logistic regression model models the posterior probability given $\mathbf{d}$ as

$$p(Y=i|\mathcal{X}(t)) = \log\left(\frac{p(Y=i|\mathcal{X}(t))}{1-p(Y=i|\mathcal{X}(t))}\right) = \alpha_i + \langle \mathcal{X}_i(t), \beta_i(t)\rangle$$

where the curve $\mathcal{X}(t)$ is assigned to class i if $p(i|\mathcal{X}) > p(j|\mathcal{X}), j = 1, \cdots, g, j \neq i$.

```
res.bin=fregre.glm(Fat~x,ldata,family=binomial())
res.gsam<-classif.glm(Fat~x,data=ldata)
summary(res.gsam)
```

```
##      - SUMMARY -
##
## -Probability of correct classification by group (prob.classification):
##         0         1
## 0.9782609 0.9610390
##
## -Confusion matrix between the theoretical groups (by rows)
##   and estimated groups (by column)
##
##       0   1
##   0 135   3
##   1   3  74
##
## -Probability of correct classification:  0.9721
```

3.2 Generalized Additive Models (GAM): classif.gsam and classif.gkam

Generalized Additive Models (see Wood (2004)) relax the linearity assumption in GLMs, allowing the use of a sum of general smooth functions f_j for the posterior probability; i.e.,

$$p(Y = i|\mathcal{X}(t)) = \log\left(\frac{p(Y = i|\mathcal{X}(t))}{1 - p(Y = i|\mathcal{X}(t))}\right) = \alpha_i + f_i\left(\mathcal{X}_i(t)\right)$$

where the functions f_i may belong to a known parametric family (polynomials, for instance) or they may even be functions to be estimated non-parametrically.

```
res.gsam<-classif.gsam(Fat~s(x),data=ldata)
summary(res.gsam)
```

```
##      - SUMMARY -
##
## -Probability of correct classification by group (prob.classification):
##         0         1
## 0.9855072 0.9610390
##
## -Confusion matrix between the theoretical groups (by rows)
##   and estimated groups (by column)
##
##       0   1
##   0 136   2
##   1   3  74
##
## -Probability of correct classification:  0.9767
```

```
#res.gkam<-classif.gkam(Fat~x,data=ldata)
```

3.3 Nonparametric classification methods: classif.knn and classif.np (Ferraty and Vieu, 2003)

These methods are based on non-parametric estimates of the densities of the groups. The most simple (and classical) one is k–nearest neighbour (kNN) in which, given $k \in \mathbb{N}$, the point $\mathbf{d}$ is assigned to the class containing a majority of the k nearest data points in the training sample.

Another possibility is to estimate $p(Y = g|\mathcal{X})$ through the Nadaraya–Watson estimator:

$$p(Y = g|\mathcal{X}) = \frac{\sum_{n=1}^{N} \mathbf{1}_{G_n=g} \, K\left(m(\mathcal{X}, \mathcal{X}_i(t))/h\right)}{\sum_{n=1}^{N} K\left(m(\mathcal{X}_i(t))/h\right)},$$

where N is the size of the training sample, G_n is the class of i-th curve in the training sample, K is a kernel and $m(\cdot, \cdot)$ is a measure of closeness between two curves (a suitable distancewhich is re-scaled by the bandwidth parameter h) .

```
## y
##         0         1
## 0.8550725 0.7142857
```

A kNN method could be considered an NP method using the uniform kernel and a locally selected bandwidth.

```
## y
##         0         1
## 0.7826087 0.7012987
```

3.4 Maximum depth: `classif.depth` (Li et al., 2012)

The most basic rule is to assign a new observation x_0 to the group that provides the highest depth to that observation (Maximum depth (MD)). The maximum depth classifier was the first attempt to use data depths instead of multivariate raw data to construct a classification rule.

Perform the following example with hidden code

Given a sample depth measure and a new observation x_0 (use the `xx.d1` curves):

1. Evaluate the depth of x_0 in both sub-samples defined by `ycat` variable (only the first 10 values are printed)

```
## Depth g1 0.4811594 0.164058 0.5717391 0.5602899 0.4068116 0.127971 0.1237681 0.7730
```

```
## Depth g2 0.4811594 0.164058 0.5717391 0.5602899 0.4068116 0.127971 0.1237681 0.7730
```

2. Assign x_0 according to the data set where it is more deeply placed.

```
## group.est: 0 1 0 0 1 1 1 0 1 1

## ycat     : 1 1 0 0 1 1 1 0 0 0
```

The function `classif.depth` performs previous tasks:

```
res.depth<-classif.depth(ycat,x.d1,depth="FM")
data.frame(res.depth$dep,group.est,res.depth$dep-cbind(d1,d2))[1:10,]
```

```
##           X1        X2 group.est d1 d2
## 1  0.4811594 0.4051948         0  0  0
## 2  0.1640580 0.4766234         1  0  0
## 3  0.5717391 0.2659740         0  0  0
## 4  0.5602899 0.2522078         0  0  0
## 5  0.4068116 0.5083117         1  0  0
## 6  0.1279710 0.3948052         1  0  0
## 7  0.1237681 0.3254545         1  0  0
## 8  0.7730435 0.3135065         0  0  0
## 9  0.3373913 0.4568831         1  0  0
## 10 0.2971014 0.4238961         1  0  0
```

3.5 The DD^G–classifier `classif.DD`

Suppose that we have implementations of a process in the product space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$ (multivariate (functional) data) where we have g groups (classes or distributions) to be separated using data depths, (Cuesta-Albertos et al., 2017). The DD^G–classifier begins by selecting a depth D and computing the following map (for $p = 1$):

$$\mathcal{X} \to \mathbb{R}^g x \to \mathbb{d} = (D_1(x), \cdots, D_g(x)).$$

We can now apply any available classification procedure that works in a g–dimensional space to separate the g groups.

where $D_0k(x)$ is the depth of x with respect to the group $k = 1, \cdots, g$. So, the DDG -Classifier compresses the information of y_i, x_i into a real space of dimension $(g + 1)$ with the form $\{y_i, D_1(x_i), \cdots, D_g(x_i)\}$.

Classification techniques in $\mathbb{R}^g$:

1. Linear Discriminant Analysis (LDA)
2. Quadratic Discriminant Analysis (QDA)
3. Generalized Linear Models (GLM)

4. Generalized Additive Models (GAM)
5. k-Nearest Neighbors (kNN)
6. Kernel Classification Method (NP)
7. Classification Trees (Tree)
8. ANN, SVMs, ...

The aim of the DD-classifier ((Li et al., 2012)) is to extend the Maximum depth classifier using a polynomial up to degree k passing through the origin as classification rule.

The DD–classifier has resolved several serious limitations of the maximum depth classifier .

Properties of the DDG -classifier:

1. A lot of classification methods available (All in the multivariate framework)

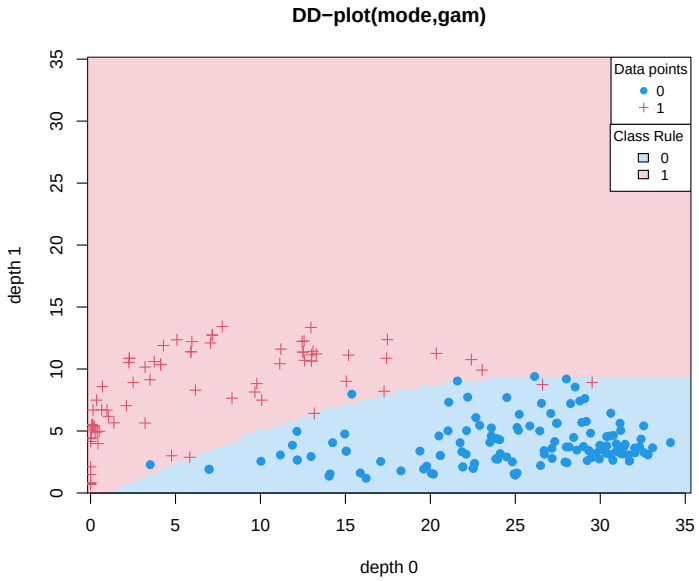
2. Using classical classification methods in the DD-plot can provide useful insights about what's going on (which depths are influential or probabilities of belonging to a certain group).

3. Possible reduction in the dimension of the classification problem, specially interesting in the Functional Framework (or in High Dimensional problems).

4. No matters how complex is the space to be analyzed, only matters that a depth function can be defined (for example, multivariate functional data MFD: $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$.

   Example DD with 2 groups

```
res.DD<-classif.DD(ycat,x.d1,classif="gam",depth="mode")
```




```
res.depth$prob.classification
```

```
## group
##         0         1
## 0.8260870 0.9350649
```

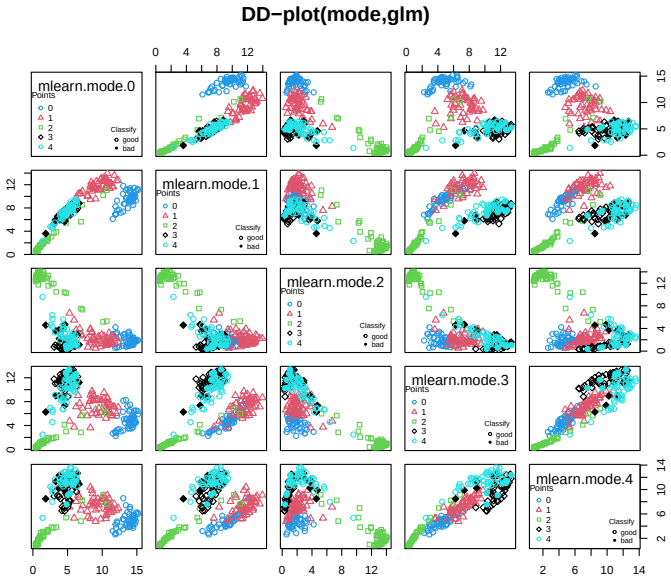
```
res.DD$prob.classification
```

```
## group
##         0         1
## 0.9782609 0.9610390
```

```
#res.DD
```

Example dDD with G groups

```
#ycat<-cut(ldata$df$Fat,3,labels=1:3)
# DD-classif for functional data: G levels
data(phoneme)
mlearn<-phoneme[["learn"]]
mlearn2<-phoneme[["test"]]
glearn<-as.numeric(phoneme[["classlearn"]])-1
out20=classif.DD(glearn,mlearn,depth="mode",classif="glm")
```




```
out21=classif.DD(glearn,list(mlearn,mlearn2),depth="modep",classif="glm",control=list(
out20 # univariate functional data
```

```
##
## -Call:
## classif.DD(group = glearn, fdataobj = mlearn, depth = "mode",     classif = "glm")
##
## -Probability of correct classification:  0.928
```

```
out21 # multivariate functional data
```

```
##
## -Call:
## classif.DD(group = glearn, fdataobj = list(mlearn, mlearn2),     depth = "modep", cl
##
## -Probability of correct classification:  0.952
```

```
#summary.classif(out21)
```

3.6 Classifiers adapted from Multivariate Framework

The idea is to recycle all the procedures known in the Multivariate Frameworkconverting an object of infinite dimension into a finite dimension. When to apply: + The basis is enough for accounting all information. + Binary/multiclass problems depends on multivariate classifier method.

```
data(phoneme)
ldata=list("df"=data.frame(glearn=phoneme$classlearn),"x"=phoneme$learn)
# require e1071 package
res.svm=classif.svm(glearn~x,data=ldata)
```

```
## [1] "fdata2model"
## [1] "sale fdata2model"
```

```
# require nnet package
res.nnet=classif.nnet(glearn~x,data=ldata,trace=FALSE)
```

```
## [1] "fdata2model"
## [1] "sale fdata2model"
```

```
# require rpart package
res.rpart=classif.rpart(glearn~x,data=ldata)
```

```
## [1] "fdata2model"
## [1] "sale fdata2model"
```

```
round(mean(res.svm$prob.classification),3)
```

```
## [1] 0.904
```

```
round(mean(res.nnet$prob.classification),3)
```

```
## [1] 0.892
```

```
round(mean(res.rpart$prob.classification),3)
```

```
## [1] 0.896
```

Add utilities in the classification functions: for example, majority voting scheme (by default ONE vs REST). R example (work in progress)

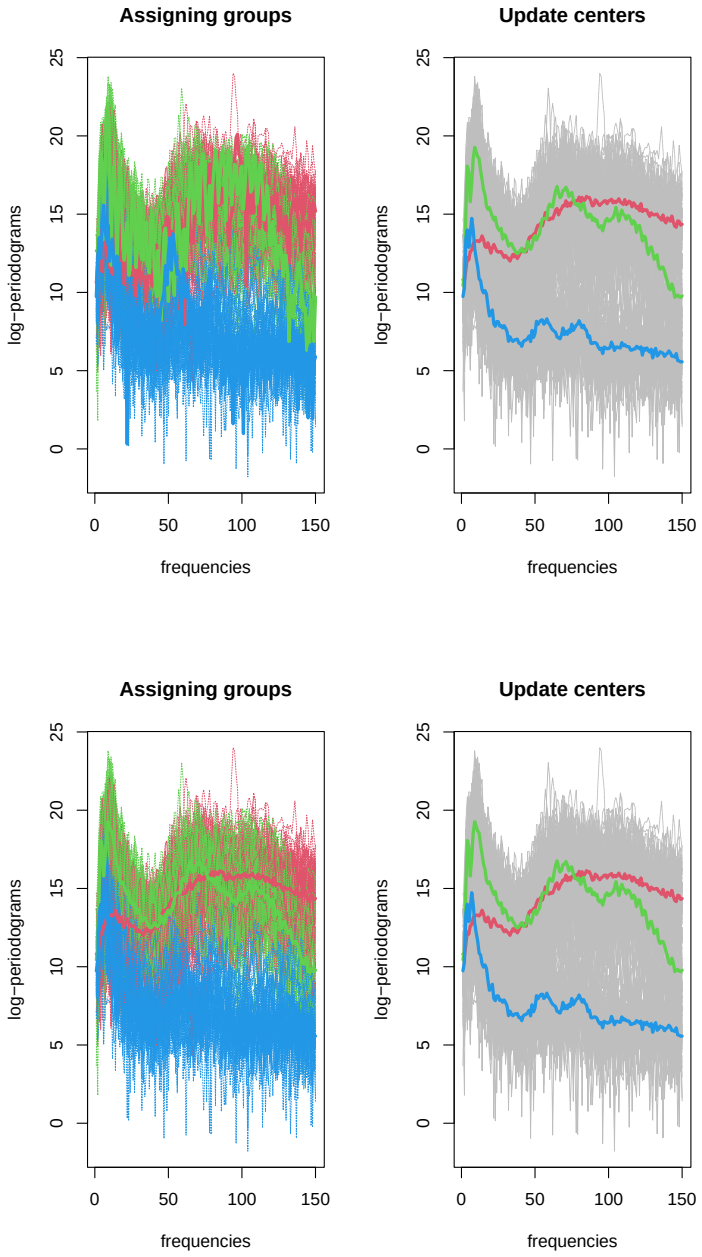
```
ii<-  c(1:10,51:60,101:110,151:160,201:250)
mlearn<-phoneme[["learn"]][ii];glearn<-phoneme[["classlearn"]][ii]
mtest<-phoneme[["test"]];gtest<-phoneme[["classtest"]]
dataf<-data.frame(glearn);ldata=list("df"=dataf,"x"=mlearn);newdat<-list("x"=mtest)
a1<-classif.glm(glearn~x, data = ldata)
a2<-classif.glm(glearn~x, data = ldata,type="majority")
a3<-classif.glm(glearn~x, data = ldata,type="majority",weights=c(rep(4,len=40),rep(1,5
# mean(predict(a1,newdat)==gtest);mean(predict(a2,newdat)==gtest);mean(predict(a3,newd
```

3.7 K-Means Clustering for functional data

Perform k-means clustering on functional data (Hartigan and Wong, 1979). The method searches the locations around which are grouped data (for a predetermined number of groups). + If ncl=NULL, randomizes the initial centers, ncl=2 using kmeans.center.ini function. + If ncl is an integer, indicating the number of groups to classify, are selected ncl initial centers using kmeans.center.ini function. + If ncl is a vector of integers, indicating the position of the initial centers with length(ncl) equal to number of groups. + If ncl is a fdata class objecct, ncl are the initial centers curves with nrow(ncl) number of groups.

The fucntion return a list with: + cluster: Indexes of groups assigned. + centers: Curves centers.

```
data(phoneme)
mlearn<-phoneme$learn[1:150,]
ylearn<-as.numeric(phoneme$classlearn[1:150])
# Unsupervised classification
kmeans.assig.groups <- fda.usc:::kmeans.assig.groups
kmeans.center.ini <- fda.usc:::kmeans.center.ini
kmeans.centers.update<-fda.usc:::kmeans.centers.update
out.fd1=fda.usc:::kmeans.fd(mlearn,ncl=c(1,51,101),draw=TRUE)
```




```
table(out.fd1$cluster,ylearn)
```

```
##    ylearn
```

```
##      1  2  3
##   1 50 10  3
##   2  0 40  1
##   3  0  0 46
```

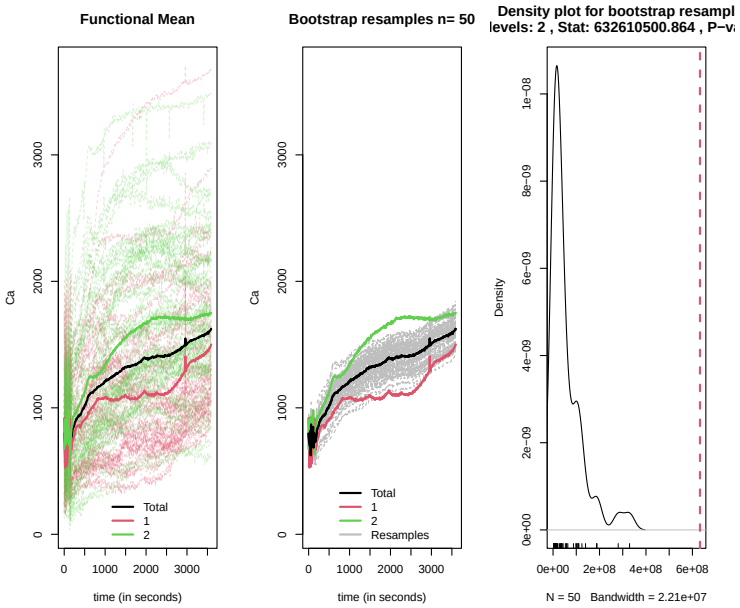
```
# Time consuming
# out.fd2=kmeans.fd(mlearn,ncl=3,draw=FALSE,par.ini=list(method="exact"))

# Different Depth function
# ind=c(17,77,126)
# out.fd3=kmeans.fd(mlearn,ncl=mlearn[ind,],draw=FALSE,
# dfunc=func.trim.FM,par.dfunc=list(trim=0.1))
```

3.8 Functional ANOVA

3.8.1 One–way anova model for functional data, Cuevas et al. (2004)

One–way anova model for k independent samples of functional data. The function contrasts the null hypothesis of equality of mean functions of functional data based on the an asymptotic version of the anova F–test. The function returns the p–value of test using one–way anova model over nboot runs.

```
data(MCO)
grupo<-MCO$classintact
datos<-MCO$intact
res=fanova.onefactor(datos,grupo,nboot=50,plot=TRUE)
```




```
res$pvalue
```

```
## [1] 0
```

3.8.2 Functional ANOVA with Random Project, Cuesta-Albertos and Febrero-Bande (2010)

The procedure is based on the analysis of randomly chosen one-dimensional projections. The function tests ANOVA models for functional data with continuous covariates and perform special contrasts for the factors in the formula.

```
data(phoneme)
names(phoneme)
```

```
## [1] "learn"      "test"       "classlearn" "classtest"
```

```
# A MV matrix obtained from functional data
data=as.data.frame(phoneme$learn$data[,c(1,seq(0,150,10)[-1])])
group=phoneme$classlearn
n=nrow(data)
group.rand=as.factor(sample(rep(1:3,len=n),n))
```

```
RP=c(2,5,15,30)
#ex 1: real factor and random factor
m03=data.frame(group,group.rand)
resul1=fanova.RPm(phoneme$learn,~group+group.rand,m03,RP=c(5,30))
summary(resul1)
```

```
##      - SUMMARY fanova.RPm -
##
##  p-value for Bonferroni method
##      group group.rand
## RP5      0    0.31629
## RP30     0    1.00000
##
##   p-value for False Discovery Rate method
##      group group.rand
## RP5      0    0.26453
## RP30     0    0.37664
```

```
#ex 2: real factor with special contrast
m0=data.frame(group)
cr5=contr.sum(5)   #each level vs last level
resul03c1=fanova.RPm(data,~group,m0,contrast=list(group=cr5))
summary(resul03c1)
```

```
##      - SUMMARY fanova.RPm -
##
##  p-value for Bonferroni method
##      group C1.group C2.group C3.group C4.group
## RP16     0        0        0        0        0
##
##   p-value for False Discovery Rate method
##      group C1.group C2.group C3.group C4.group
## RP16     0        0        0        0        0
```

```
#ex 3: random factor with special contrast. Same projs as ex 2.
m0=data.frame(group.rand)
zz=resul03c1$proj
cr3=contr.sum(3)   #each level vs last level
resul03c1=fanova.RPm(data,~group.rand,m0,contrast=list(group.rand=cr3),zproj=zz)
summary(resul03c1)
```

```
##      - SUMMARY fanova.RPm -
##
```

```
##  p-value for Bonferroni method
##      group.rand C1.group.rand C2.group.rand
## RP16          1       0.90142       0.65644
##
##   p-value for False Discovery Rate method
##      group.rand C1.group.rand C2.group.rand
## RP16    0.74012       0.56936       0.51361
```

Capítulo 4

Variable Selection

4.1 Functiondal regression with points of impact

4.1.1 State of Art

- Nonparametric variable selection approach (NOVAS). NOVAS is quite expensive from a computational perspective, Ferraty et al. (2010).

- A wavelet-based weighted LASSO functional linear (FWLASSO). FWLASSO requires transforming the original variables and assuming a linear model, Zhao et al. (2015).

- Berrendero et al. (2016) use the Maxima-hunting proposal to choose the most relevant design points in functional classification setting.

4.1.2 Local maxima distance correlation approach (LMDC), (Ordóñez et al., 2018)

- In this work we study the utility of distance correlation Székely et al. (2007) as an intrinsic method for variable selection.

- Neither projection nor transformation of the variables is needed. Moreover, it is unnecessary to assume an a priori regression model.

- LMDC approach consists in calculating the local maxima of the distance correlation along the curve.

4.1.3 LMDC Algorithm: `LMDC.select()` function

1. Calculate de distance correlation (DC) $R(t) = \left\{R(X(t_j), Y)\right\}_{j=1}^{N}$, from the data $\left\{X_i(t_j), Y_i\right\}_{i=1}^{n}$.

2. Calculate the LM of the $\hat{\mathcal{R}}(t)$. Only the significant local maxima for a default level of significance are selected. Denoting the arguments values (argvals) of the local maxima a $\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_{\tilde{N}}$ ($\tilde{N} < N$), we ordered them from highest to lowest values of DC, that is $\hat{\mathcal{R}}(\tilde{t}_1) \geq \hat{\mathcal{R}}(\tilde{t}_2) > \ldots \geq \hat{\mathcal{R}}(\tilde{t}_{\tilde{N}})$

```
library(fda.usc)
```

```
data(tecator)
X.d2<-fdata.deriv(tecator[["absorp.fdata"]],
nderiv = 2)
colnames(X.d2[["data"]])<-paste0("X",round(X.d2[["argvals"]]))
dat <- data.frame("y"=tecator[["y"]][["Fat"]],X.d2[["data"]] )
tol<-.2
dc.raw <- LMDC.select("y",data = dat, tol = tol,pvalue = 0.05,
plot=F)
# Preselected impact points
covar<-names(dat)[-1][dc.raw[["maxLocal"]]]
covar
```

```
##  [1] "X933"  "X1046" "X907"  "X886"  "X896"  "X1010" "X1020" "X1030" "X945"
## [10] "X876"  "X915"  "X862"  "X993"
```

```
length(covar)
```

```
## [1] 13
```

4.1.4 LMDC Algorithm: `LMDC.regre()` function

3. (Optionally) Check if the relationship between the reponse and the predictor variables is linear: $H_0 : Y = \langle X, \beta \rangle + \epsilon$, versus a general alternative using a test of linearity proposed in García-Portugués et al. (2014).

```
ftest<-flm.test(dat[,-1], dat[,"y"],
verbose=F,plot.it=F)
```

```
## [1] "PLS1"
## [1] "PLS2"
```

```
ftest
```

```
##
##  PCvM test for the functional linear model using optimal PLS basis
##  representation
##
## data:  Y=<X,b>+e
## PCvM statistic = 216.51, p-value < 2.2e-16
```

```
ftest[["p.value"]]
```

```
## [1] 0
```

4. Fit a regression model to the response of interest Y using the vector of covariates $X(\tilde{t}) = \{X(\tilde{t}_1), \ldots, X(\tilde{t}_{\tilde{N}})\}$. A linear model will be used if the null hypothesis is not rejected and a nonparametric (e.g. generalized additive model) model otherwise.

5. (Optionally) Once the type model has been selected, we propose to Apply a forward stepwise regression method to determine the significant covariates, taking advantage of the fact that the local maxima have been ordered. This means we start with a model with the first covariate (the one with the highest value of distance correlation), and the rest of the ordered covariates are added to the model in turn. This substantially reduces the computing time.

```
if (ftest$p.value > 0.05) { # Linear relationship, step-wise lm is recommended
out <- LMDC.regre(y = "y", covar = covar, data = dat, pvalue=.05, method ="lm")
} else {# Non-Linear relationship, step-wise gam is recommended
out <- LMDC.regre(y = "y", covar = covar, data = dat,pvalue=.05, method ="gam")}
out
```

```
## $model
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(X933, k = 4) + s(X1046, k = 4) + s(X907, k = 4) + s(X886,
##     k = 4) + s(X896, k = 4) + s(X1010, k = 4) + s(X1020, k = 4) +
##     s(X1030, k = 4) + s(X945, k = 4) + s(X876, k = 4) + s(X915,
##     k = 4) + s(X993, k = 4)
##
## Estimated degrees of freedom:
```

```
## 3.00 2.55 2.83 2.00 1.00 1.00 2.94
## 2.92 2.81 2.57 1.00 2.85  total = 28.48
##
## GCV score: 0.4403176
##
## $xvar
##  [1] "X933"  "X1046" "X907"  "X886"  "X896"  "X1010" "X1020" "X1030" "X945"
## [10] "X876"  "X915"  "X993"
##
## $pred
## NULL
##
## $edf
## [1] 28.48142
##
## $nvar
## [1] 12
```

Differences in mean square prediction error between linear (usign `lm` model) and non-linear (usign `gam` model) model

```
out <- LMDC.regre(y = "y", covar = covar, data = dat[1:165,],newdata=dat[166:215,], pva
mean((out$pred-dat$y[166:215])^2)
```

```
## [1] 11.75474
```

```
out <- LMDC.regre(y = "y", covar = covar, data = dat[1:165,],newdata=dat[166:215,], pva
mean((out$pred-dat$y[166:215])^2)
```

```
## [1] 1.148573
```

Binary classification example (Impact point selection, model estimation and prediction)

```
data(tecator)
X.d2<-fdata.deriv(tecator[["absorp.fdata"]],
nderiv = 2)
colnames(X.d2[["data"]])<-paste0("X",round(X.d2[["argvals"]]))
y2groups <- ifelse(tecator[["y"]][["Fat"]]<12,0,1)
dat <- data.frame("y2groups"=y2groups,X.d2[["data"]] )
tol<-.1
dc.raw <- LMDC.select("y2groups",data = dat, tol = tol,pvalue = 0.05,
plot=F)
```

```
# Preselected impact points
covar<-names(dat)[-1][dc.raw[["maxLocal"]]]
covar
```

```
##  [1] "X945"  "X905"  "X933"  "X886"  "X1020" "X876"  "X1030" "X896"  "X1046"
## [10] "X862"  "X1010" "X915"  "X965"
```

```
length(covar)
```

```
## [1] 13
```

```
# GLM model (using binomial family), other multivariate model can be used
ind <- 1:129
ldata<-list("df"=dat[ind,])
form.glm<-formula(paste0("y2groups~",paste0(covar,collapse="+")))
out.glm  <- classif.glm(form.glm, data = ldata)
summary(out.glm)
```

```
##      - SUMMARY -
##
## -Probability of correct classification by group (prob.classification):
## 0 1
## 1 1
##
## -Confusion matrix between the theoretical groups (by rows)
##   and estimated groups (by column)
##
##      0  1
##   0 58  0
##   1  0 71
##
## -Probability of correct classification:  1
```

```
# Prediction
newldata<-list("df"=dat[-ind,])
pred.glm<-predict(out.glm,newldata)

# Confusion matrix
table(newldata$df$y2groups,pred.glm)
```

```
##    pred.glm
##      0  1
##   0 40  0
##   1  3 43
```

4.2 Variable selection in functional regression

Febrero-Bande et al. (2019) consider the problem of variable selection in regression models in the case of functional variables that may be mixed with other type of variables (scalar, multivariate, directional, etc.).

Our proposal begins with a simple null model and sequentially selects a new variable to be incorporated into the model based on the use of distance correlation proposed by (Székely et al., 2007). For the sake of simplicity, this paper only uses additive models.

$$Y_i = \alpha + \sum_{j=1}^{J} f_j(X_i^{(j)}) + \varepsilon_i, \quad i = 1, \dots, N$$

The proposed algorithm may assess the type of contribution (linear, non linear, ...) of each variable. The algorithm has shown quite promising results when applied to simulations and real data sets.

4.2.1 State of Art

- Stepwise regression, Akaike (1973). The main idea is to use some diagnostic tools, directly derived from the linear model, to evaluate the contribution of a new covariate and decide whether it should be included in the model. The final subset is usually constructed using: the forward and/or the backward selection.
- Feature Selection using LASSO. The work by Tibshirani, 1996 proposing the LASSO estimator includes a l_1-type constraint for the coefficient vector β. Several examples following the same line but using penalties or constraints such as: LARS (Efron et al. (2004)) and COSSO (Lin et al. (2006)). Each methods is based on a specific model, all the covariates must be included in the model at the same time and for functional data problems, the previous steps that commonly include variable standardization.
- Berrendero et al. (2016) use the Minimum Redundance Maximum Relevance (mRMR) procedure to choose the most relevant design points in functional classification setting.
- A pure feature selection methods where the covariate is selected without a model. This is the approach employed in minimum Redundancy Maximum Relevance (mRMR), (Peng et al. (2005)) where a new candidate covariate must have a great relevancy with the response while maintaining a lower redundancy with the covariates already selected in the model. he main advantage of this approach is that it is an incremental rule but the

measures for redundancy and relevancy must be chosen in function of the regression model applied to ensure good predictive results in the final model. Berrendero et al. (2018) used the Reproducing Kernel Hilbert Space (RKHS) for variable selection in FLM.

- Boosting, see Ferraty and Vieu (2009) in a functional data context. Boosting selects at each step the best covariate/model with respect to the unexplained part of the response. The final prediction is constructed as a combination of the different steps.
- Partial distance correlation (PDC): used in Yenigün and Rizzo (2015) for VS in multivariate linear models, a definition of PDC among X and Y given Z was introduced based on computing the distance correlation among the residuals of two models: Y respect to Z and X respect to Z. PDC is constructed under linear relationship assumptions among variables. Its implementation only uses the distance matrices among elements of X, Y and Z (variables should have a similar scale).

Specifically, Z (the variables already in the model) could be a mix of functional, scalar or multivariate variables where an appropriate distance using all of them must be hard to compute. Even restricting ourselves to the scalar case, those variables should have a similar scale.

4.2.2 Algorithm

All the previous solutions are not completely satisfactory in a functional data framework, specially when the number of possible covariates can be arbitrarily large. We are interested in an automatic regression procedure capable of dealing with a large number of covariates of different nature, possibly very closely related to one another.

The key of the whole procedure is the extensive use of the DC that presents two important advantages: the choice of the variate is made without considering a model and it is possible to compute this quantity for variates of different nature as it is only computed from distances. The distance correlation (DC) is computed among the residuals of the current model with each candidate. Taking into account that the residuals have the same nature as the response variable, the DC can always be computed at each step.

Our proposal is presented is a very general way, we have restricted ourselves to additive models that offer a balanced compromise between predictive ability and simplicity. The obtained results are quite promising in scenarios where no competitors are available because no other procedure can deal with variates of different nature in a homogeneous way.

The procedure was applied to a real problem related with the Iberian Energy Market (Price and Demand) where the number of possible covariates is really

big. The algorithm was able to find synthetic regression models offering interesting insights about the relationship among the response and the covariates. The final selected models mix functional, scalar and categorical information.

Our algorithm can be formalized as follows:

1. Let Y the response and $S = \{X^1, \dots, X^p\}$ the set of all possible predictors.
2. Set $\widehat{Y} = \bar{Y}$, and let $M^{(0)} = \emptyset$ the initial set of the variates included in the model. Set $i = 0$.
3. Compute the residuals of the current model: $\hat{\varepsilon} = Y - \widehat{Y}$.
4. Choose $X^j \in S$ such that: 1) $\mathcal{R}\{\hat{\varepsilon}, X^j\} \geq \mathcal{R}\{\hat{\varepsilon}, X^k\}, \forall k \neq j \in S$ and 2) the null hypothesis for the test of independence among $\{X^j\}$ and $\hat{\varepsilon}$ is rejected. IF NOT, END.
5. Update the sets M and S: $M^{(i+1)} = M^{(i)} \cup \{X^j\}$, and $S = S \backslash \{X^j\}$.
6. Compute the new model for Y using $M^{(i+1)}$ choosing the best contribution of the new covariate. Typically, there will be a catalog of all possible ways of constructing correct models with the variates in $M^{(i+1)}$ fixing the contributions of the variates in $M^{(i)}$ and adding the new one.
7. Analyze the contribution of X^j in the new model respect to the current:

- IF this contribution is not relevant (typically comparing with the current model) THEN $M^{(i+1)} = M^{(i+1)} \backslash \{X^j\}$ and the current model remains unalterable
- ELSE the new model becomes the current model and provides new predictions ($\widehat{Y}$). Along the paper we have employed an additive model: $\widehat{Y} = \bar{Y} + \sum_{m \in M} \hat{f}_m \left(X^{(m)}\right)$ where at each step $\hat{f}_m$ could be linear or nonlinear.

8. Update the number of iterations: $i = i + 1$ and go to 3
9. END.

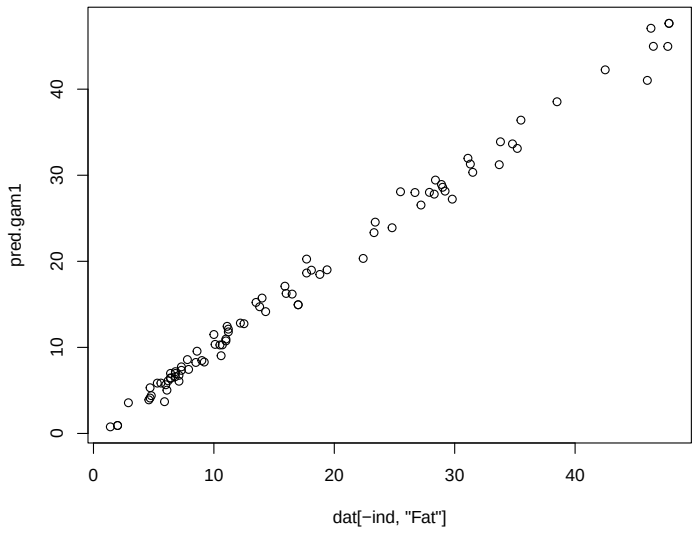
The current model is the final model with the variates included in $M^{(i)}$. S is either the empty set or contains those variables that accept the null hypothesis of the test of independence respect to the residuals of the current model.

4.2.3 R example

```
data(tecator)
y=tecator$y$Fat

# Potential functional covariates
x=tecator$absorp.fdata
x1<-fdata.deriv(x)
x2<-fdata.deriv(x,nderiv=2)

# Potential factor covariates
xcat0<-cut(rnorm(length(y)),4)
xcat1<-cut(tecator$y$Protein,4)
xcat2<-cut(tecator$y$Water,4)
ind <- 1:129

# 3 functionals (x,x1,x2), 3 factors (xcat0, xcat1, xcat2)
# and 100 potential scalars covariates (impact poitns of x1)
dat    <- data.frame("Fat"=y, x1$data, xcat1, xcat2)
ldat <- list("df"=dat[ind,],"x"=x[ind,],"x1"=x1[ind,],"x2"=x2[ind,])

# Time consuming
res.gam1<-fregre.gsam.vs(data=ldat,y="Fat")
summary(res.gam1$model)
```

```
##       Fat             X97                x2.PC1              x2.PC2
##  Min.   : 0.90   Min.   :-0.015836   Min.   :-0.003367   Min.   :-3.082e-03
##  1st Qu.: 7.70   1st Qu.:-0.010165   1st Qu.:-0.002367   1st Qu.:-7.165e-04
##  Median :14.60   Median :-0.009397   Median :-0.000950   Median :-2.801e-05
##  Mean   :18.24   Mean   :-0.009414   Mean   : 0.000000   Mean   : 0.000e+00
##  3rd Qu.:27.80   3rd Qu.:-0.008574   3rd Qu.: 0.001717   3rd Qu.: 6.665e-04
##  Max.   :49.10   Max.   :-0.006377   Max.   : 0.009573   Max.   : 6.246e-03
##      x2.PC3               x2.PC4
##  Min.   :-0.0032986   Min.   :-1.041e-03
##  1st Qu.:-0.0001912   1st Qu.:-1.193e-04
##  Median : 0.0001008   Median : 2.363e-05
##  Mean   : 0.0000000   Mean   : 0.000e+00
##  3rd Qu.: 0.0002728   3rd Qu.: 1.270e-04
##  Max.   : 0.0034901   Max.   : 1.131e-03
```

```
# Prediction like fregre.gsam()
newldat <- list("df"=dat[-ind,],"x"=x[-ind,],"x1"=x1[-ind,],"x2"=x2[-ind,])
pred.gam1<-predict(res.gam1,newldat)
plot(dat[-ind,"Fat"],pred.gam1)
```


pred.gam1
0
10
20
30
40
0
10
20
30
40
dat[−ind, "Fat"]

References

Documentation

RPubs documents

1. fda.usc vignette: Installation and Descriptive Statistics
2. fda.usc vignette:Functional Regression
3. fda.usc vignette: Functional Classification and ANOVA
4. fda.usc vignette: Variable Selection in Functional Regression (and Classification)
5. fda.usc reference card Rcard

Links:

1. fda.usc R package
2. JSS paper
3. fda.usc R Manual
4. Manuel Oviedo's website

Back to Top

Capítulo 5

Funding and Financial Support:

This work has been supported by Project MTM2016-76969-P from Ministerio de Economía y Competitividad - Agencia Estatal de Investigacion and European Regional Development Fund (ERDF).

Bibliografía

Akaike, H. (1973). Maximum likelihood identification of gaussian autoregressive moving average models. *Biometrika*, 60(2):255–265.

Aneiros-Pérez, G. and Vieu, P. (2006). Semi-functional partial linear regression. *Statist. Probab. Lett.*, 76(11):1102–1110.

Berrendero, J. R., Cuevas, A., and Torrecilla, J. L. (2016). Variable selection in functional data classification: a maxima-hunting proposal. *Statistica Sinica*, pages 619–638.

Berrendero, J. R., Cuevas, A., and Torrecilla, J. L. (2018). On the use of reproducing kernel hilbert spaces in func2tional classification. *Journal of the American Statistical Association*, 113:1210–1218.

Cardot, H., Crambes, C., and Sarda, P. (2005). Quantile regression when the covariates are functions. *Nonparametric Statistics*, 17(7):841–856.

Cardot, H., Ferraty, F., and Sarda, P. (1999). Functional linear model. *Statist. Probab. Lett.*, 45(1):11–22.

Carmack, P. S., Spence, J. S., and Schucany, W. R. (2012). Generalised correlated cross-validation. *Journal of Nonparametric Statistics*, 24(2):269–282.

Chiou, J.-M., Muller, H.-G., Wang, J.-L., et al. (2004). Functional response models. *Statistica Sinica*, 14(3):675–694.

Cuesta-Albertos, J. and Febrero-Bande, M. (2010). A simple multiway anova for functional data. *Test*, 19(3):537–557.

Cuesta-Albertos, J. and Nieto-Reyes, A. (2008). The random tukey depth. *Computational Statistics and Data Analysis*, 52(11):4979–4988.

Cuesta-Albertos, J. A., Febrero-Bande, M., and Oviedo de la Fuente, M. (2017). The ddg-classifier in the functional setting. *Test*, 26(1):119–142.

Cuevas, A., Febrero, M., and Fraiman, R. (2004). An anova test for functional data. *Comput. Statist. Data Anal.*, 47(1):111–122.

Cuevas, A., Febrero, M., and Fraiman, R. (2007). Robust estimation and classification for functional data via projection-based depth notions. *Comput. Statist.*, 22(3):481–496.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of statistics*, 32(2):407–499.

Faraway, J. J. (1997). Regression analysis for a functional response. *Technometrics*, 39(3):254–261.

Febrero-Bande, M., Galeano, P., and González-Manteiga, W. (2010). Measures of influence for the functional linear model with scalar response. *J. Multivariate Anal.*, 101(2):327–339.

Febrero-Bande, M., Galeano, P., and González-Manteiga, W. (2008). Outlier detection in functional data by depth measures, with application to identify abnormal NO_x levels. *Environmetrics*, 19(4):331–345.

Febrero-Bande, M. and González-Manteiga, W. (2013). Generalized additive models for functional data. *Test*, 22(2):278–292.

Febrero-Bande, M., González-Manteiga, W., and Oviedo de la Fuente, M. (2019). Variable selection in functional additive regression models. *Computational Statistics*, 34(2):469–487.

Febrero-Bande, M. and Oviedo de la Fuente, M. (2012). Statistical computing in functional data analysis: the R package fda.usc. *J. Statist. Software*, 51(4):1–28.

Ferraty, F., Goia, A., Salinelli, E., and Vieu, P. (2013). Functional projection pursuit regression. *Test*, 22(2):293–320.

Ferraty, F., Hall, P., and Vieu, P. (2010). Most-predictive design points for functional data predictors. *Biometrika*, 97(4):807–824.

Ferraty, F., Park, J., and Vieu, P. (2011). Estimation of a functional single index model. In *Recent advances in functional data analysis and related topics*, pages 111–116. Springer.

Ferraty, F., Van Keilegom, I., and Vieu, P. (2012). Regression when both response and predictor are functions. *Journal of Multivariate Analysis*, 109:10–28.

Ferraty, F. and Vieu, P. (2003). Curves discrimination: a nonparametric functional approach. *Comput. Statist. Data Anal.*, 44(1):161–173.

Ferraty, F. and Vieu, P. (2006). *Nonparametric Functional Data Analysis*. Springer Series in Statistics. Springer-Verlag, New York. Theory and practice.

Ferraty, F. and Vieu, P. (2009). Additive prediction and boosting for functional data. *Comput. Statist. Data Anal.*, 53(4):1400–1413.

Fraiman, R. and Muniz, G. (2001). Trimmed means for functional data. *Test*, 10(2):419–440.

García-Portugués, E., González-Manteiga, W., and Febrero-Bande, M. (2014). A goodness-of-fit test for the functional linear model with scalar response. *Journal of Computational and Graphical Statistics*, 23(3):761–778.

Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108.

Kato, K. et al. (2012). Estimation in functional linear quantile regression. *The Annals of Statistics*, 40(6):3108–3136.

Krämer, N. and Sugiyama, M. (2011). The degrees of freedom of partial least squares regression. *Journal of the American Statistical Association*, 106(494):697–705.

Li, J., Cuesta-Albertos, J. A., and Liu, R. Y. (2012). *dd*–classifier: Nonparametric classification procedure based on *dd*–plot. *J. Amer. Statist. Assoc.*, 107(498):737–753.

Lin, Y., Zhang, H. H., et al. (2006). Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34(5):2272–2297.

López-Pintado, S. and Romo, J. (2009). On the concept of depth for functional data. *Journal of the American Statistical Association*, 104:718–734.

Müller, H.-G. et al. (2008). Functional modeling of longitudinal data. In *Longitudinal data analysis*, pages 237–266. Chapman and Hall/CRC.

Muller, H.-G. and Stadtmuller, U. (2005). Generalized functional linear models. *Annals of Statistics*, pages 774–805.

Müller, H.-G. and Yao, F. (2012). Functional additive models. *Journal of the American Statistical Association*.

Ordóñez, C., Oviedo de la Fuente, M., Roca-Pardiñas, J., and Rodríguez-Pérez, J. R. (2018). Determining optimum wavelengths for leaf water content estimation from reflectance: A distance correlation approach. *Chemometrics and Intelligent Laboratory Systems*, 173:41–50.

Oviedo de la Fuente, M., Febrero-Bande, M., Muñoz, M. P., and Domínguez, À. (2018). Predicting seasonal influenza transmission using functional regression models with temporal dependence. *PloS one*, 13(4):e0194250.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (8):1226–1238.

Preda, C. and Saporta, G. (2005). PLS regression on a stochastic process. *Comput. Statist. Data Anal.*, 48(1):149–158.

Ramsay, J. and Silverman, B. (2005a). *Functional Data Analysis*. Springer.

Ramsay, J. O. and Silverman, B. W. (2005b). *Functional Data Analysis*. Springer Series in Statistics. Springer-Verlag, New York, second edition.

Székely, G. J. and Rizzo, M. L. (2013). The distance correlation t-test of independence in high dimension. *J. Multivariate Anal.*, 117:193–213.

Székely, G. J., Rizzo, M. L., and Bakirov, N. K. (2007). Measuring and testing dependence by correlation of distances. *Ann. Statist.*, 35(6):2769–2794.

Wood, S. N. (2004). Stable and efficient multiple smoothing parameter estimation for generalized additive models. *J. Amer. Statist. Assoc.*, 99(467):673–686.

Yenigün, C. D. and Rizzo, M. L. (2015). Variable selection in regression using maximal correlation and distance correlation. *Journal of Statistical Computation and Simulation*, 85(8):1692–1705.

Zhao, Y., Chen, H., and Ogden, R. T. (2015). Wavelet-based weighted lasso and screening approaches in functional linear regression. *Journal of Computational and Graphical Statistics*, 24(3):655–675.

Zivot, E. and Wang, J. (2006). *Modeling financial time series with S-PLUS*, volume 2. Springer.