Functional Data Analysis

Ishi Jain (220464)
BS-SDS
Indian Institute of Technology, Kanpur

Contents

1 Introduction

Functional data analysis (FDA) deals with the analysis and theory of data that are in the form of functions, images and shapes, or more general objects. The atom of functional data is a function, where for each subject in a random sample one or several functions are recorded. Functional data are intrinsically infinite dimensional. The high intrinsic dimensionality of these data poses challenges both for theory and computation, where these challenges vary with how the functional data were sampled. On the other hand, the high or infinite dimensional structure of the data is a rich source of information, which brings many opportunities for research and data analysis.

2 Mathematical formulism

2.1 Hilbertian random variables

In the Hilbert space viewpoint, one considers an H-valued random element X, where H is a separable Hilbert space such as the space of square-integrable functions $L^2[0,1]$. Under the integrability condition that $\mathbb{E}\|X\|_{L^2}^2 = \mathbb{E}\left(\int_0^1 |X(t)|^2 dt\right) < \infty$, one can define the mean of X as the unique element $\mu \in H$ satisfying

$$\mathbb{E}\langle X, h\rangle = \langle \mu, h\rangle, \quad h \in H.$$

This formulation is the Pettis integral but the mean can also be defined as Bochner integral $\mu = \mathbb{E}X$. Under the integrability condition that $\mathbb{E}\|X\|_{L^2}^2$ is finite, the covariance operator of X is a linear operator $\mathcal{C} : H \to H$ that is uniquely defined by the relation

$$\mathcal{C}h = \mathbb{E}\left[\langle h, X-\mu\rangle (X-\mu)\right], \quad h \in H,$$

The spectral theorem allows us to decompose X as the Karhunen-Loève decomposition

$$X = \mu + \sum_{i=1}^{\infty} \langle X, \varphi_i\rangle \varphi_i,$$

where φ_i are eigenvectors of $\mathcal{C}$, corresponding to the nonnegative eigenvalues of $\mathcal{C}$, in a non-increasing order. Truncating this infinite series to a finite order underpins functional principal component analysis.

2.2 Stochastic Processes

The stochastic process perspective views X as a collection of random variables $\{X(t)\}_{t\in[0,1]}$ indexed by the unit interval (or more generally interval $\mathcal{T}$). The mean and covariance functions are defined in a pointwise manner as

$$\mu(t) = \mathbb{E}X(t), \quad \Sigma(s,t) = \mathrm{Cov}(X(s), X(t)), \quad s,t \in [0,1]$$

(if $\mathbb{E}[X(t)^2] < \infty$ for all $t \in [0,1]$). Under the mean square continuity, μ and Σ are continuous functions and then the covariance function Σ defines a covariance operator $\mathcal{C} : H \to H$ given by

$$(\mathcal{C}f)(t) = \int_0^1 \Sigma(s,t) f(s)\, ds.$$

2.3 Mercer's theorem

Suppose K is a continuous symmetric positive-definite kernel. Then there is an orthonormal basis $\{e_i\}_i$ of $L^2[a,b]$ consisting of eigenfunctions of T_K such that the corresponding sequence of eigenvalues $\{\lambda_i\}_i$ is nonnegative. The eigenfunctions corresponding to non-zero eigenvalues are continuous on $[a,b]$ and K has the representation

$$K(s,t) = \sum_{j=1}^{\infty} \lambda_j e_j(s) e_j(t)$$

where the convergence is absolute and uniform.

3 Functional Principal Component Analysis

The main objective of FPCA is to identify the dominant modes of variation in a dataset of functional data. It decomposes the variability of the functions into principal components, which are orthogonal functions that capture the most significant patterns of variation.

In this section, we focus on first-generation functional data that are realizations of a stochastic process $X(\cdot)$ that is in L^2 and defined on the interval I with mean function $\mu(t) = \mathbb{E}(X(t))$ and covariance function $\Sigma(s,t) = \mathrm{cov}(X(s), X(t))$. The realization of the process for the ith subject is $X_i =$

$X_i(\cdot)$, and the sample consists of n independent subjects. For generality, we allow the sampling schedules to vary across subjects and denote the sampling schedule for subject i as $t_{i1}, \ldots, t_{ini}$ and the corresponding observations as $X_i = (X_{i1}, \ldots, X_{ini})$, where $X_{ij} = X_i(t_{ij})$. In addition, we allow the measurement of X_{ij} to be contaminated by random noise e_{ij} with $\mathbb{E}(e_{ij}) = 0$ and $\mathrm{var}(e_{ij}) = \sigma^2$, so the actual observed value is $Y_{ij} = X_{ij} + e_{ij}$, where e_{ij} are independent across i and j and often termed "measurement errors".

3.1 Estimation of Mean and Covariance Functions

When subjects are sampled at the same time schedule, i.e., $t_{ij} = t_j$ and $n_i = m$ for all i, the observed data are m-dimensional multivariate data, so the mean and covariance can be estimated empirically at the measurement times by the sample mean and sample covariance,

$$\hat{\mu}(t_j) = \frac{1}{n_1} \sum_{i=1}^{n_1} Y_{ij},$$

and

$$\hat{\Sigma}(t_k, t_l) = \frac{1}{n_1} \sum_{i=1}^{n_1} (Y_{ik} - \hat{\mu}(t_k))(Y_{il} - \hat{\mu}(t_l)), \quad \text{for } k \neq l.$$

Data that are missing completely at random can be handled easily by adjusting the available sample size at each time point t_j for the mean estimate or by adjusting the sample sizes of available pairs at (t_k, t_l) for the covariance estimate. An estimate of the mean and covariance functions on the entire interval I can then be obtained by smooth interpolation of the corresponding sample estimates or by mildly smoothing over the grid points. Once we have a smoothed estimate $\hat{\Sigma}$ of the covariance function Σ, the variance of the measurement error at time t_j can be estimated as

$$\hat{\sigma}^2(t_j) = \frac{1}{n_1} \sum_{i=1}^{n_1} (Y_{ij} - \hat{\mu}(t_j))^2 - \hat{\Sigma}(t_j, t_j),$$

because $\mathrm{var}(Y(t)) = \mathrm{var}(X(t)) + \sigma^2(t)$.

3.2 Functional Principal Component Analysis (FPCA)

Principal component analysis is a key dimension reduction tool for multivariate data that has been extended to functional data and termed functional principal component analysis (FPCA). Dimension reduction is achieved through an expansion of the underlying but often not fully observed random trajectories Xi(t) in a functional basis that consists of the eigenfunctions of the (auto)-covariance operator of the process X.

We define,

$$\sum(g) = \int_I \sum(s,t) g(s)\, ds$$

By Mercer's theorem, the kernel of $\mathcal{C}$, i.e., the covariance function $\Sigma(\cdot,\cdot)$, has spectral decomposition

$$\Sigma(s,t) = \sum_{k=1}^{\infty} \lambda_k \varphi_k(s) \varphi_k(t),$$

where the series convergence is absolute and uniform, and λ_k are real-valued nonnegative eigenvalues in descending order with the corresponding orthonormal eigenfunctions $\varphi_k(t)$. By the Karhunen–Loève theorem, the FPCA expansion of an underlying random trajectory is

$$X_i(t) = \mu(t) + \sum_{k=1}^{\infty} A_{ik} \varphi_k(t),$$

where $A_{ik} = \int_0^1 (X_i(t) - \mu(t)) \varphi_k(t) dt$ are the functional principal components (FPCs), sometimes referred to as scores. The Karhunen–Loève expansion facilitates dimension reduction in the sense that the partial sum converges uniformly, i.e., $\sup_{t \in [0,1]} \mathbb{E}[X_i(t) - \mu(t) - \sum_{k=1}^{K} A_{ik} \varphi_k(t)]^2 \to 0$ as $K \to \infty$ and thus the partial sum with a large enough K yields a good approximation to the infinite sum. Thereby, the information in X_i is reduced from infinite dimensional to a K-dimensional vector $A_i = (A_{i1}, ..., A_{iK})$ with the approximated process:

$$X_i^{(K)}(t) = \mu(t) + \sum_{k=1}^{K} A_{ik} \varphi_k(t).$$

3.3 Application of FPCA

Applied FPCA on the USArrests dataset. **CODE:**

```r
# Perform PCA

df <- USArrests
df <- na.omit(df)
df <- scale(df)
pca_result <- prcomp(df)

# Extract loadings (eigenvectors)

loadings <- pca_result$rotation

# Calculate percentage of variation explained

eigenvalues <- pca_result$sdev^2
percentage_variation <- eigenvalues / sum(eigenvalues) * 100

# Plot the first four eigenfunctions

par(mfrow = c(2, 2))  # Set up a 2x2 grid for plotting

for (i in 1:4) {
  # Plot the eigenfunction
  plot(loadings[, i], type = "l", main = paste("Eigenfunction
    ", i), xlab = "Index", ylab = "Value", col = "red")

  # Add text box with percentage of variation
  text(x = 1, y = max(loadings[, i]) - 0.2, labels = paste("
    Variation:", round(percentage_variation[i], 2), "%"),pos =
     4, col = "blue")
}
```

Listing 1: Performing PCA and plotting eigenfunctions

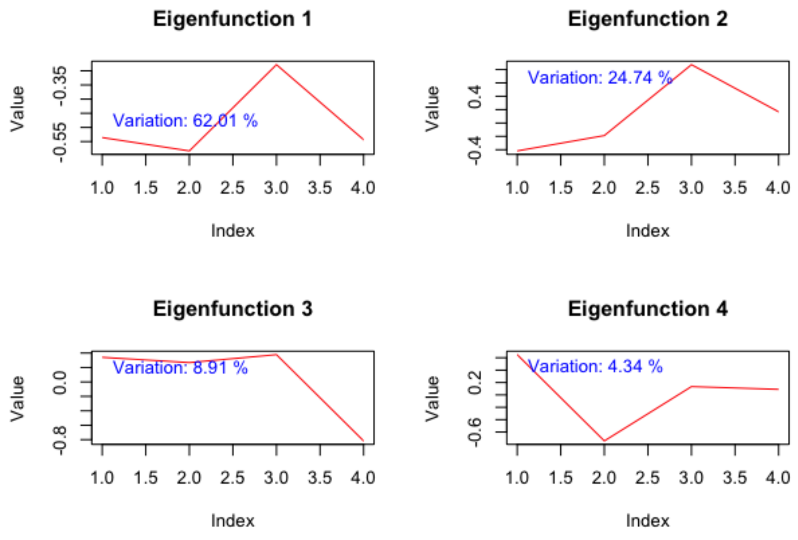

Figure 1: Eigen-Functions

4 Correlation and Regression: Inverse Problems and Dimension Reduction for Functional Data

4.1 Functional Canonical Correlation Analysis (FCCA)

FCCA aims to find correlations between two sets of functional data. It extends the concept of canonical correlation analysis to functional data, seeking linear combinations of functions in each set that maximize the correlation between the combinations. For a pair of random functions $X \in L^2(I_X)$ and $Y \in L^2(I_Y)$, the first canonical coefficient ρ_1 and its associated weight functions (u_1, v_1) are defined as:

$$\begin{aligned}
\rho_1 &= \sup_{u \in L^2(I_X), v \in L^2(I_Y)} \operatorname{corr}(\langle u, X \rangle, \langle v, Y \rangle) \\
&= \sup_{u \in L^2(I_X), v \in L^2(I_Y)} \frac{\operatorname{cov}(\langle u, X \rangle, \langle v, Y \rangle)}{\sqrt{\operatorname{var}(\langle u, X \rangle)}\sqrt{\operatorname{var}(\langle v, Y \rangle)}} \\
&= \operatorname{corr}(\langle u_1, X \rangle, \langle v_1, Y \rangle)
\end{aligned}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in L^2 space ($p = 2$), i.e., $\langle f_1, f_2 \rangle = \int_I f_1(t) f_2(t)\, dt,\ f_1, f_2 \in L^2(I)$.

The kth canonical coefficient ρ_k, given $\rho_1, \rho_2, \ldots, \rho_{k-1}$, and its associated weight functions (u_k, v_k) are defined as:

$$\begin{aligned}\rho_k &= \sup_{\substack{u \in L^2(I_X), v \in L^2(I_Y) \\ u \perp u_1, \ldots, u_{k-1}, v \perp v_1, \ldots, v_{k-1}}} \frac{\operatorname{cov}(\langle u, X\rangle, \langle v, Y\rangle)}{\sqrt{\operatorname{var}(\langle u, X\rangle)}\sqrt{\operatorname{var}(\langle v, Y\rangle)}} \\ &= \operatorname{corr}(\langle u_k, X\rangle, \langle v_k, Y\rangle)\end{aligned}$$

where $(U_k, V_k) = (\langle u_k, X\rangle, \langle v_k, Y\rangle)$ is uncorrelated with all previous pairs $(U_j, V_j) = (\langle u_j, X\rangle, \langle v_j, Y\rangle)$ for $j = 1, 2, \ldots, k-1$. Thus, FCCA implements projections in the directions of U_k and V_k for X and Y respectively, such that their linear combinations (inner products) (U_k, V_k) are maximally correlated. X and Y are uncorrelated if all their canonical correlations are zero, equivalently, if and only if $\rho_1 = 0$.

4.2 Functional Regression

Functional linear models can be viewed as an extension of the traditional multivariate linear models that associates vector responses with vector covariates. The traditional linear model with scalar response $Y \in \mathbb{R}$ and vector covariate $X \in \mathbb{R}^p$ can be expressed as

$$\begin{aligned}Y &= \beta_0 + \langle X, \beta \rangle + \varepsilon \\ &= \beta_0 + X_1\beta_1 + \cdots + X_p\beta_p + \varepsilon,\end{aligned}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in Euclidean space, $\beta_0 \in \mathbb{R}$ and $\beta \in \mathbb{R}^p$ denote the regression coefficients, and ε is a zero-mean finite-variance random error (noise). Functional linear models can be divided into two types based on the responses.

4.2.1 Functional regression models with scalar response

Replacing the vector covariate X and the coefficient vector β in model (3) by a centered functional covariate $X^c(t) = X(t) - \mu(t)$ and coefficient function $\beta = \beta(t)$ for $t \in [0, 1]$, and replacing the inner product in Euclidean space by that in Hilbert space L^2, one arrives at the functional linear model

$$\begin{aligned} Y &= \beta_0 + \langle X^c, \beta \rangle + \varepsilon \\ &= \beta_0 + \int_0^1 X^c(t)\beta(t)\, dt + \varepsilon. \end{aligned}$$

The simple functional linear model (4) can be extended to multiple functional covariates, $\{X_j\}_{j=1}^p$, also including additional vector covariates $Z = (Z_1, \ldots, Z_q)$, where $Z_1 = 1$, by

$$Y = \langle Z, \theta \rangle + \sum_{j=1}^{p} \int_0^1 X_j^c(t)\beta_j(t)\, dt + \varepsilon,$$

where $\theta \in \mathbb{R}^q$ is the regression coefficient for Z, the domain of X_j is $[0, 1]$, X_j^c is the centered functional covariate given by $X_j^c(t) = X_j(t) - \mu_j(t)$, and β_j is the regression coefficient function for X_j^c, for $j = 1, \ldots, p$.

Consider an orthonormal basis $\{\varphi_k\}_{k\geq 1}$ of the function space. Then expanding both X and β in this basis leads to

$$X(t) = \sum_{k=1}^{\infty} A_k \varphi_k(t), \quad \beta(t) = \sum_{i=1}^{\infty} \beta_k \varphi_k(t).$$

Hence,

$$Y = \beta_0 + \sum_{k=1}^{\infty} \beta_k A_k + e$$

The above simple functional linear model can be extended to multiple functional covariates $X_1, \ldots, X_p$, also including additional vector covariates $Z = (Z_1, \ldots, Z_q)$, where $Z_1 = 1$, by

$$Y = \langle Z, \theta \rangle + \sum_{j=1}^{p} \int_{I_j} X_j^c(t)\beta_j(t) dt + e$$

where I_j is the interval where X_j is defined. In theory, these intervals need not be the same.

Application of Functional regression models with scalar response on CanadianWeather dataset

```r
#install.packages("fda")
library(fda)
#print(names(CanadianWeather))
annualprec <- log10(apply(CanadianWeather$dailyAv[,,"
    Precipitation.mm"], 2,sum))
smallbasis  <- create.fourier.basis(c(0, 365), 25)
## The covariate is the temperature curve for each station
tempfd <- smooth.basis(day.5, CanadianWeather$dailyAv[,,"
    Temperature.C"], smallbasis)$fd
## smooths the daily temperature data for each station using
    the Fourier basis created earlier and stores it in the
    variable "tempfd"
precip.Temp1 <- fRegress(annualprec ~ tempfd, method="
    fRegress")
## performs functional regression using the annual
    precipitation as the response variable and the smoothed
    temperature data as the predictor variable.
#names(precip.Temp1)
annualprec.fit1 <- precip.Temp1$yhatfdobj
plot(annualprec.fit1, annualprec, type="p", pch="o")
lines(annualprec.fit1, annualprec.fit1, lty=2)
## plots the observed annual precipitation against the fitted
     values obtained from the regression
RMSE <- round(sqrt(mean((annualprec-annualprec.fit1)^2)),3)
#print(paste("RMSE =",RMSE))
plot(precip.Temp1$betaestlist[[2]]) #plots the estimated
    regression function obtained from the functional
    regression



##
## Get the default setup and modify it
## the "model" value of the method argument causes the
    analysis
## to produce a list vector of arguments for calling the
## fRegress function
##
precip.Temp.mdl1 <- fRegress(annualprec ~ tempfd, method="
    model")
# First confirm we get the same answer as above by calling
    function fRegress() with these arguments:
precip.Temp.m <- do.call('fRegress', precip.Temp.mdl1)
```

```r
all.equal(precip.Temp.m, precip.Temp1)
#outputs 'TRUE'



#  set up a smaller basis for beta2 than for temperature so
   that we get a more parsimonious fit to the data
nbetabasis2 <- 21  #  not much less, but we add some
   roughness penalization
betabasis2  <- create.fourier.basis(c(0, 365), nbetabasis2)
betafd2     <- fd(rep(0, nbetabasis2), betabasis2)
# add smoothing
betafdPar2  <- fdPar(betafd2, lambda=10)
# replace the regress coefficient function with this fdPar
   object
precip.Temp.mdl2 <- precip.Temp.mdl1
precip.Temp.mdl2[['betalist']][['tempfd']] <- betafdPar2
# Now do re-fit the data
precip.Temp2 <- do.call('fRegress', precip.Temp.mdl2)


# Compare the two fits:
#  degrees of freedom
precip.Temp1[['df']] # 26
precip.Temp2[['df']] # 22
#  root-mean-squared errors:
RMSE1 <- round(sqrt(mean(with(precip.Temp1, (yhatfdobj-yvec)
   ^2))),3)
RMSE2 <- round(sqrt(mean(with(precip.Temp2, (yhatfdobj-yvec)
   ^2))),3)
#print(c(RMSE1, RMSE2))
annualprec.fit2 <- precip.Temp2$yhatfdobj
plot(annualprec.fit2, annualprec, type="p", pch="o", xlab= "
   Predicted Log10 Annual Precipitation", ylab="Observed
   Log10 Annual Precipitation")
lines(annualprec.fit2, annualprec.fit2, lty=2)
#plots the observed annual precipitation against the fitted
   values from the more parsimonious model.
plot(precip.Temp2$betaestlist[[2]], xlab="Day of Year", ylab=
   "Estimated Coefficients")
#plots the estimated regression function obtained from the
   more parsimonious model.


#it is primarily the temperatures in the early winter that
```

```
   provide the fit to log precipitation by temperature

## Manual construction of xfdlist and betalist
xfdlist <- list(const=rep(1, 35), tempfd=tempfd)

# The intercept must be constant for a scalar response
betabasis1 <- create.constant.basis(c(0, 365))
betafd1    <- fd(0, betabasis1)
betafdPar1 <- fdPar(betafd1)

betafd2     <- fd(matrix(0,7,1), create.bspline.basis(c(0,
   365),7))
# convert to an fdPar object
betafdPar2  <- fdPar(betafd2)

betalist <- list(const=betafdPar1, tempfd=betafdPar2)

precip.Temp3   <- fRegress(annualprec, xfdlist, betalist)
annualprec.fit3 <- precip.Temp3$yhatfdobj
#  plot the data and the fit
plot(annualprec.fit3, annualprec, type="p", pch="o", xlab="
   Predicted Log10 Annual Precipitation", ylab="Observed
   Log10 Annual Precipitation")
lines(annualprec.fit3, annualprec.fit3)
plot(precip.Temp3$betaestlist[[2]], xlab="Day of Year", ylab=
   "Estimated Coefficients")
```

In this code, the response variable is a scalar, representing the log10 of annual precipitation for 35 weather stations. The predictor variable, tempfd, represents the functional form of temperature curves for each station.

The code fits functional linear regression models using different approaches, such as Fourier basis functions and smoothing splines, and compares their performance in terms of model fit and complexity. It also explores methods to handle the regression coefficient functions, considering both automatic and manual approaches.

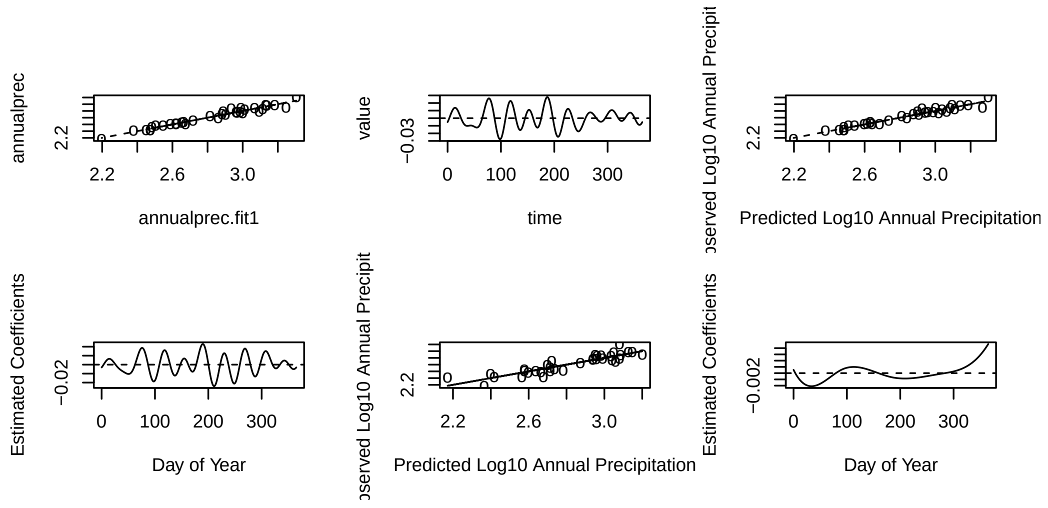

Figure 2: Functional regression models with scalar response

4.2.2 Functional regression models with functional response

Consider a functional response $Y(s)$ on $[0,1]$ and multiple functional covariates $X_j(t)$, $t \in [0,1]$, $j = 1,\dots,p$. Two major models have been considered in this setup. One of these two models, generally referred to as functional linear model (FLM), can be written as:

$$Y(s) = \alpha_0(s) + \sum_{j=1}^{p} \int_0^1 \alpha_j(s,t) X_j^c(t)\, dt + \varepsilon(s),$$

where $\alpha_0(s)$ is the functional intercept, for $j = 1,\dots,p$, $X_j^c(t) = X_j(t) - \mu_j(t)$ is a centered functional covariate on $[0,1]$, $\alpha_j(s,t)$ is the corresponding functional slopes with the same domain, respectively, and $\varepsilon(s)$ is usually a random process with mean zero and finite variance. In this case, at any given time $s \in [0,1]$, the value of Y, i.e., $Y(s)$, depends on the entire trajectories of $\{X_j(t)\}_{j=1}^p$. Model (6) has been studied extensively.

Function-on-scalar regression

In particular, taking $X_j(\cdot)$ as a constant function yields a special case of model (6):

$$Y(s) = \alpha_0(s) + \sum_{j=1}^{p} X_j \alpha_j(s) + \varepsilon(s),$$

which is a functional linear model with functional responses and scalar covariates.

Concurrent regression models

This model is given by,

$$Y(s) = \beta_0(s) + \sum_{j=1}^{p} \beta_j(s) X_j(s) + \varepsilon(s),$$

where $X_1,\dots,X_p$ are functional covariates on $[0,1]$, $\beta_0, \beta_1, \dots, \beta_p$ are the coefficient functions defined on the same interval, and $\varepsilon(s)$ is usually assumed to be a random process with mean zero and finite variance. This model assumes that the value of $Y(s)$ depends only on the current value of $\{X_j(s)\}_{j=1}^p$ and not the history $\{X_j(t) : t \leq s\}_{j=1}^p$ or future value. Hence, it is a "concurrent regression model," which is also referred to as a "varying-coefficient" model. Various estimation methods have been proposed.

Application of functional regression on Canadian Weather dataset.

CODE:

```
## simplest:  formula interface
daybasis65 <- create.fourier.basis(rangeval=c(0, 365), nbasis
    =65,
                                   axes=list('axesIntervals')
    )
Temp.fd <- with(CanadianWeather, smooth.basisPar(day.5,
                                    dailyAv[,,'Temperature.C'
    ], daybasis65)$fd)
TempRgn.f <- fRegress(Temp.fd ~ region, CanadianWeather)

## Get the default setup and possibly modify it
TempRgn.mdl <- fRegress(Temp.fd ~ region, CanadianWeather,
    method='model')

# make desired modifications here then run

TempRgn.m <- do.call('fRegress', TempRgn.mdl)

# no change, so match the first run

all.equal(TempRgn.m, TempRgn.f)
```

Listing 2: functional response with scalar explanatory variables

CODE:

```
## formula interface

gaittime    <- as.matrix((1:20)/21)
gaitrange   <- c(0,20)
gaitbasis   <- create.fourier.basis(gaitrange, nbasis=21)
gaitnbasis  <- gaitbasis$nbasis
gaitcoef    <- matrix(0,gaitnbasis,dim(gait)[2])
harmaccelLfd <- vec2Lfd(c(0, (2*pi/20)^2, 0), rangeval=
    gaitrange)
gaitfd      <- smooth.basisPar(gaittime, gait, gaitbasis,
                               Lfdobj=harmaccelLfd, lambda=1e
    -2)$fd
hipfd  <- gaitfd[,1]
```

```r
kneefd <- gaitfd[,2]

knee.hip.f <- fRegress(kneefd ~ hipfd)

## manual set-up
#  set up the list of covariate objects
const  <- rep(1, dim(kneefd$coef)[2])
xfdlist  <- list(const=const, hipfd=hipfd)

beta0 <- with(kneefd, fd(gaitcoef, gaitbasis, fdnames))
beta1 <- with(hipfd,  fd(gaitcoef, gaitbasis, fdnames))

betalist  <- list(const=fdPar(beta0), hipfd=fdPar(beta1))

fRegressout <- fRegress(kneefd, xfdlist, betalist)
par(oldpar)
```

Listing 3: functional response with functional explanatory variables

5 Clustering and Classification of Functional Data

For vector-valued multivariate data, k-means partitioning methods and hierarchical clustering are two main approaches. For clustering of functional data, k-means clustering methods are more popular than hierarchical clustering methods. For k-means clustering on functional data, mean functions are usually regarded as the cluster centers. Covariance structures have also been taken into consideration.

Functional classification assigns a group membership to a new data object either based on functional regression or functional discriminant analysis. Functional data classification methods based on functional regression models use class levels as responses and the observed functional data and other covariates as predictors. For regression based functional classification models, functional generalized linear models or more specifically, functional binary regression, such as functional logistic regression for binary responses, are commonly used classification approaches. More generally, the generalized functional linear regression model based on the FPCA approach is used.

Hierarchical Clustering on US Arrests dataset.

CODE:

```r
df <- USArrests
dim(df)
head(df)
df <- na.omit(df)
df <- scale(df)
d <- dist(df, method = "euclidean")
hc <- hclust(d)
sub_grp <- cutree(hc, k = 4)
library(factoextra)
fviz_cluster(list(data = df, cluster = sub_grp))
plot(hc, cex = 0.6)
```

Listing 4: Hierarchical Clustering on US Arrests dataset.

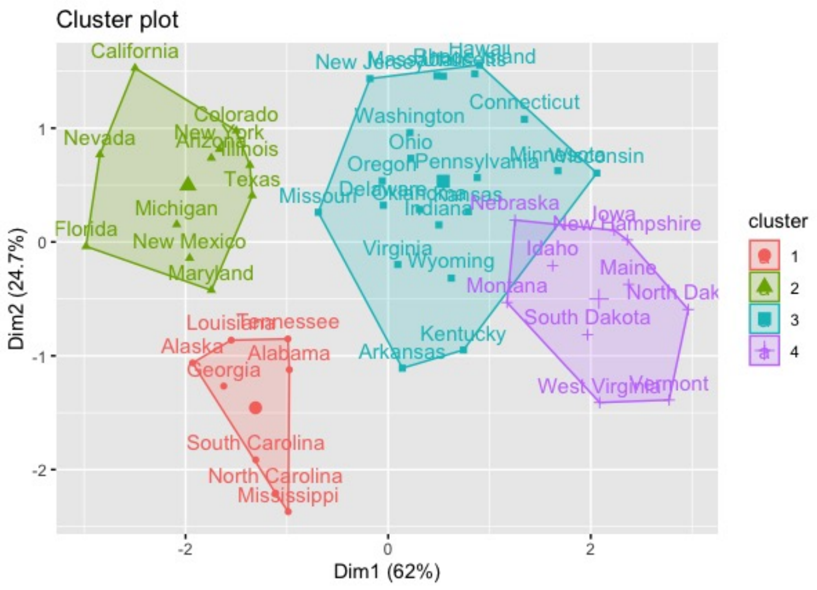

Figure 3: Hierarchical Clustering

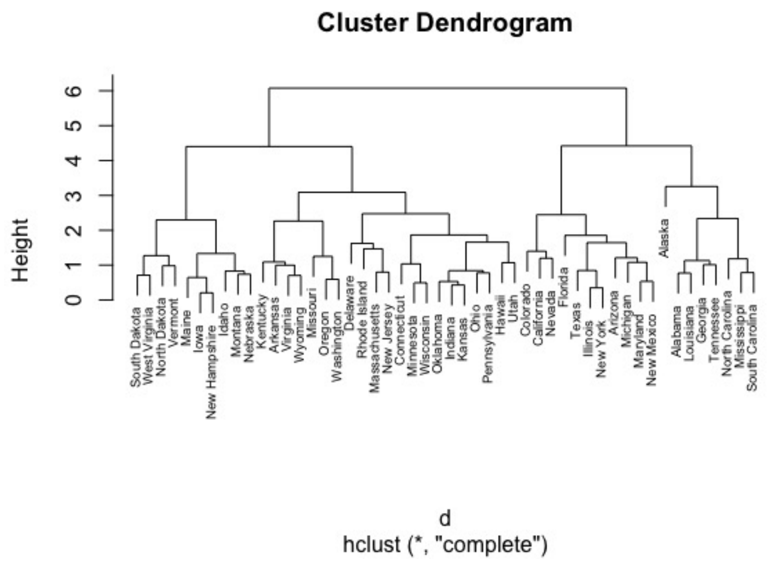

Figure 4: Hierarchical Clustering

kmeans clustering on US Arrests dataset. CODE:

```r
df <- USArrests
df <- na.omit(df)
df <- scale(df)
# nstart is the number of random starting points
clusters <- kmeans(df, 4, nstart = 10)
library(factoextra)
fviz_cluster(clusters, df)
### Practical Applications
library(animation)
kmeans.ani(df, 4)
fviz_nbclust(df, kmeans, method = "wss")
```

Listing 5: kMeans Clustering on US Arrests dataset.

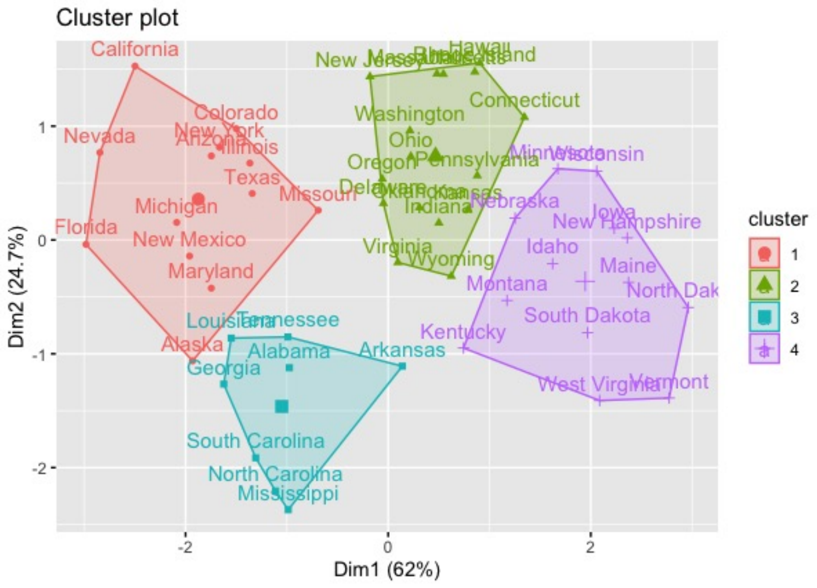

Figure 5: kMeans Clustering

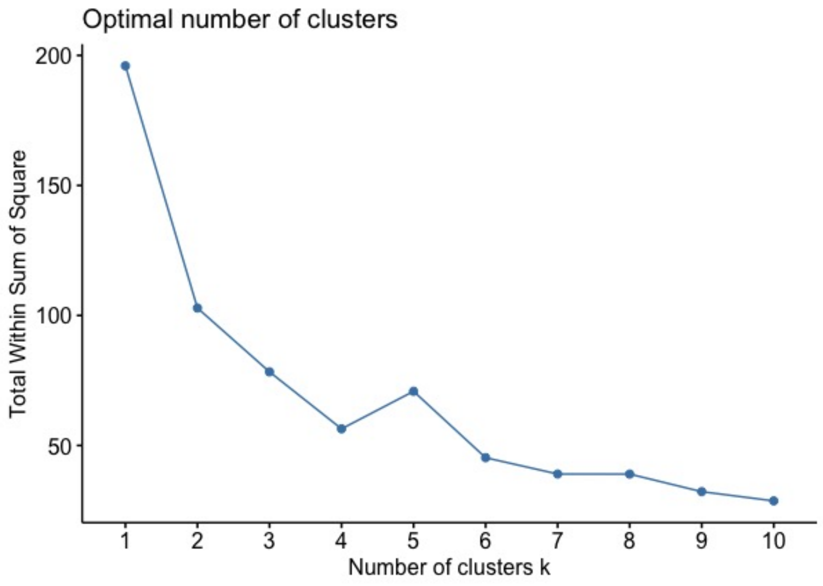

Figure 6: kMeans Clustering

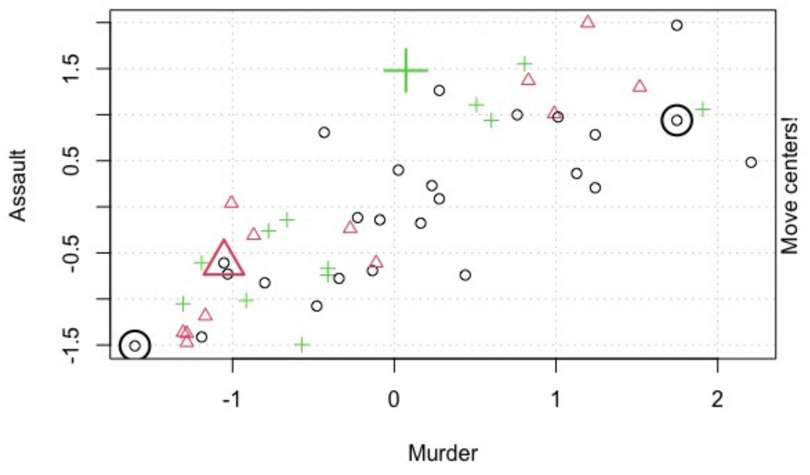

Figure 7: Formation of Clusters

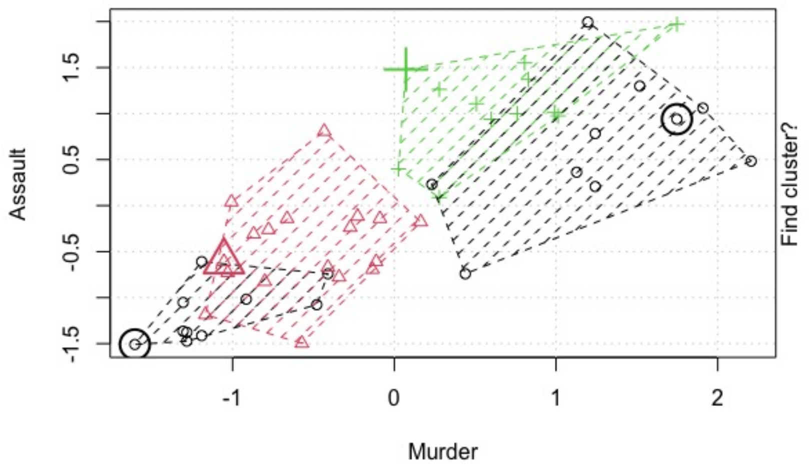

Figure 8: Formation of Clusters

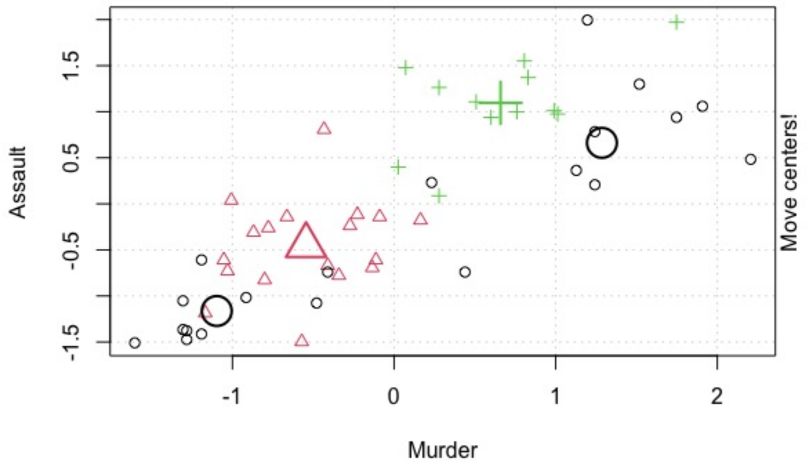

Figure 9: Formation of Clusters

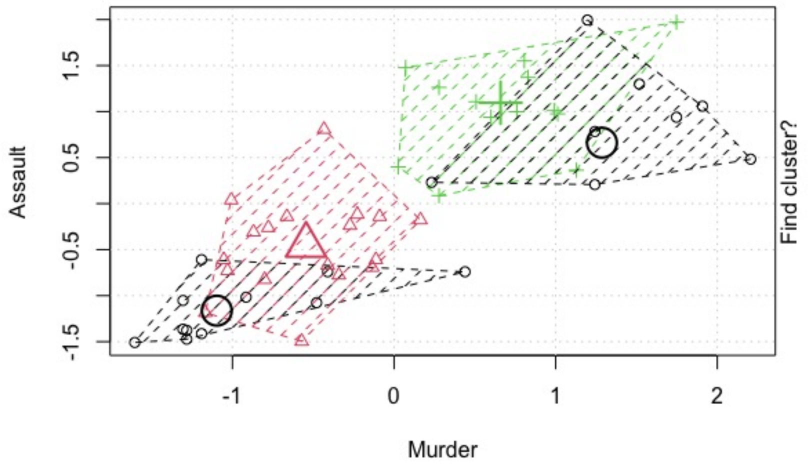

Figure 10: Formation of Clusters

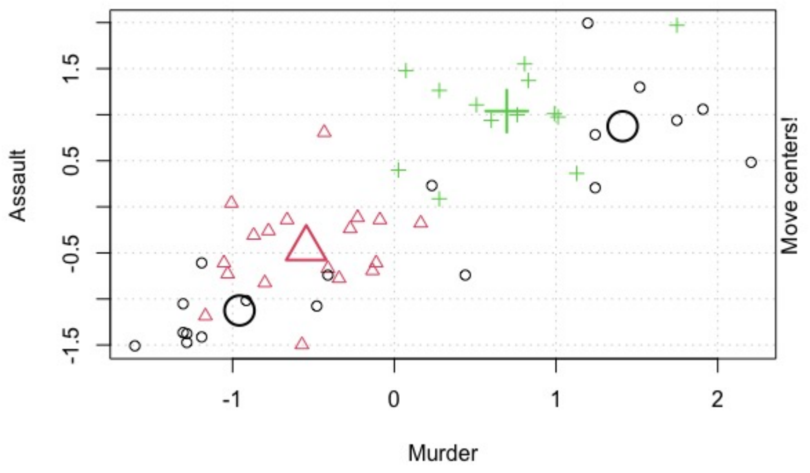

Figure 11: Formation of Clusters

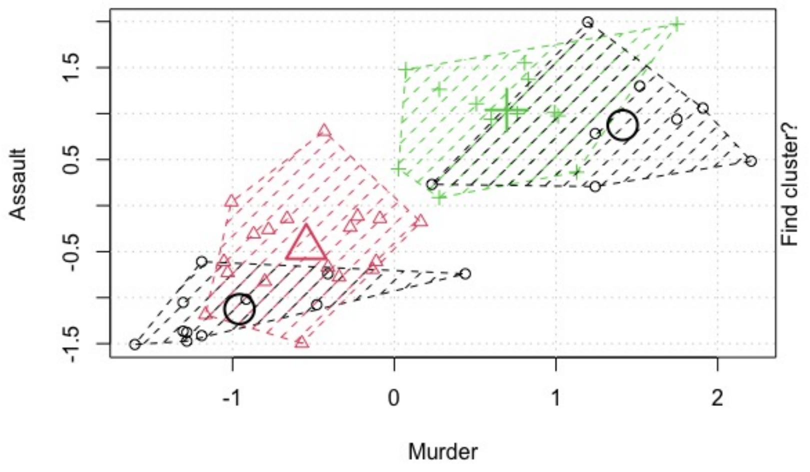

Figure 12: Formation of Clusters

6 Time Warping

Time warping, also known as curve registration, curve alignment or time synchronization, aims to identify and separate amplitude variation and time variation. If both time and amplitude variation are present, then the observed functional data Y_i can be modeled as

$$Y_i(t) = X_i[h_i^{-1}(t)], \quad t \in [0,1],$$

where $X_i \sim$ i.i.d. X is a latent amplitude function and $h_i \sim$ i.i.d. h is a latent time warping function that corresponds to a cumulative distribution function. The time warping functions h are assumed to be invertible and to satisfy $\mathbb{E}(h^{-1}(t)) = t$. The simplest case of a family of warping functions to specify phase variation is linear transformation, that is $h(t) = \delta + \gamma t$, which warps the time of an underlying template function by subjected-specific shift and scale.

7 References

1. Wikipedia page of FDA
2. Functional Data Analysis by Ramsay Silverman(2nd Edition)
3. Introduction to Mathematical Statistics by Hogg and Craig
4. `https://cran.r-project.org/web/packages/fda/index.html`
5. Review of Functional Data Analysis by JL Wang
6. Ruriko Yoshida - Linear Algebra and Its Applications with R