Functional Spectral Clustering (FSC)

This project introduces a new approach for clustering functional data based on spectral clustering analysis. This approach is flexiable and can employ the original trajectories or the first derivatives or the second derivatives to perform the clustering.

Getting Started

To apply the FSC method use the r function; *FSC.fun(matrix.data, t, basis, nclusters, d)*, the function requires the following arguments:

- matrix.data: the original data in matrix format,
- t: the timeline of the data,
- basis: the selected smoothing basis,
- nclusters: the number of clusters,
- d: the choice of d is 0, or 1, or 2, where d=0 refers to the original trajectories, d=1 refers to the first derivatives, and d=2 refers to the second derivatives. The user can apply each at a time.

Details

The clustering procedure is straightforward once the smoothing technique is chosen with care. The smoothing stage plays an important role in this technique and can determine the success of the clustering results, see fda pacakge for creating basis functions. While the number of clusters in the data can be determined by different ways, for instance NbClust can be used to estimate the number of clusters of the original data. Also, in some cases the number of clusters is known priori from the data. Finally, regarding the choice of d; usually clustering functional data is based on the original trajectories. However, it has been shown that the first derivatives (rate of change) can sometimes hold more information about the data and accordingly it can detect the similarities/dissimilarities better. Similary the same concept applied for the second derivatives (accelaration). From experinace, I noticed that the set of functional data that shows more variations in amplitude and phase between the curves is infact gives better clustering results when applying the FSC technique.

Prerequisites

The function requires the following packages:

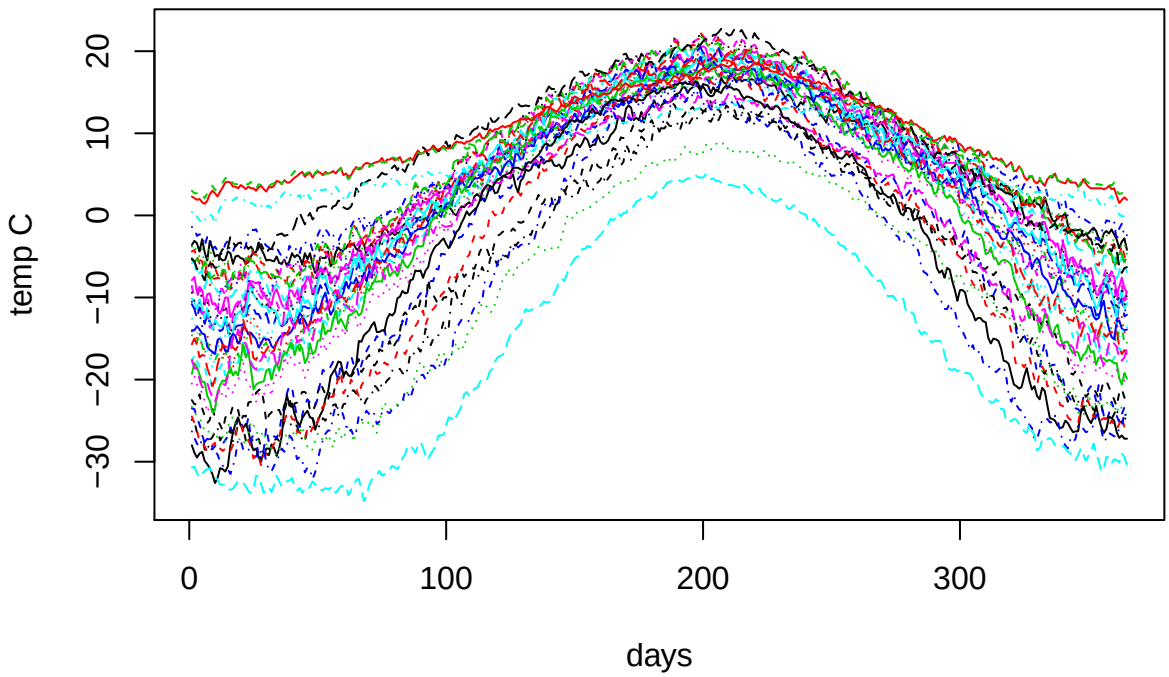
- fda
- fda.usc

Example

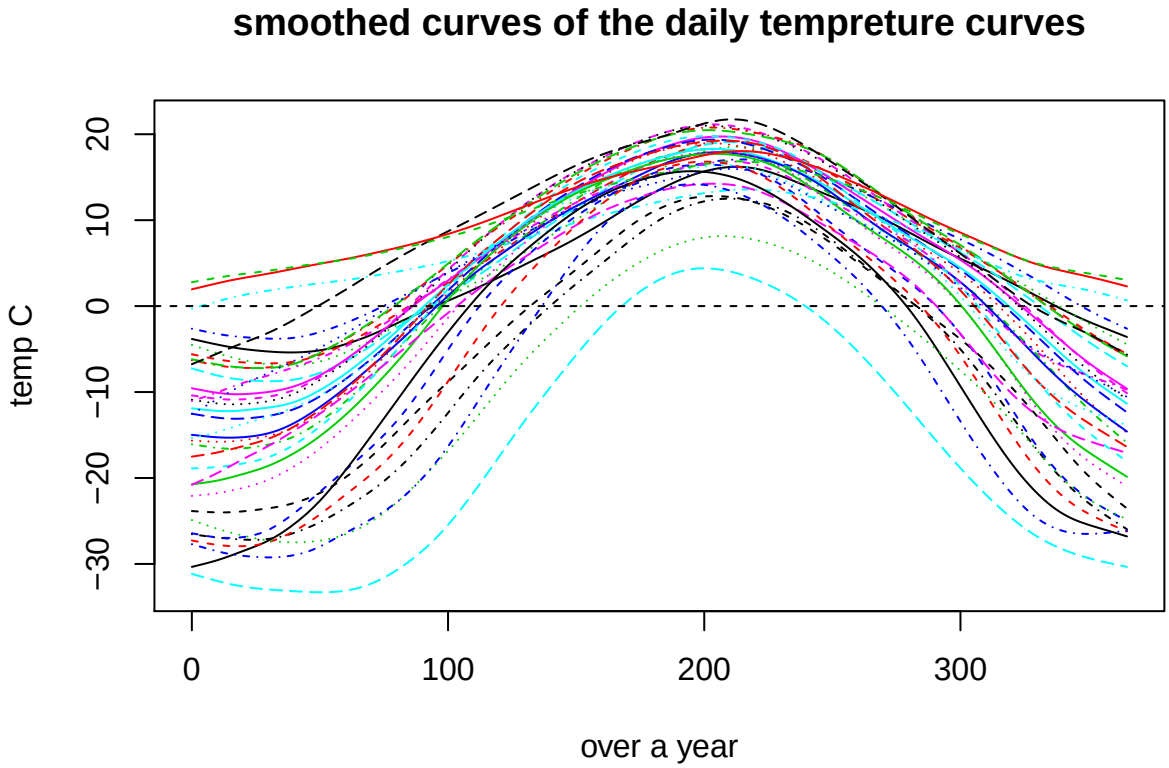
I will illustrate the methodology through the Canadian weather data example. This data consists of temperature measures for 35 selected cities distributed across Canada, it has been introduced in (Ramsay, and Silverman, 2005) and it has been frequently used in functional data analysis researches.

```
require("fda")
require("fda.usc")
data("CanadianWeather")
# name the temperature data as y
y <- CanadianWeather$dailyAv[,,1]
# define the timeline as 365 days
t <- 1:365
```

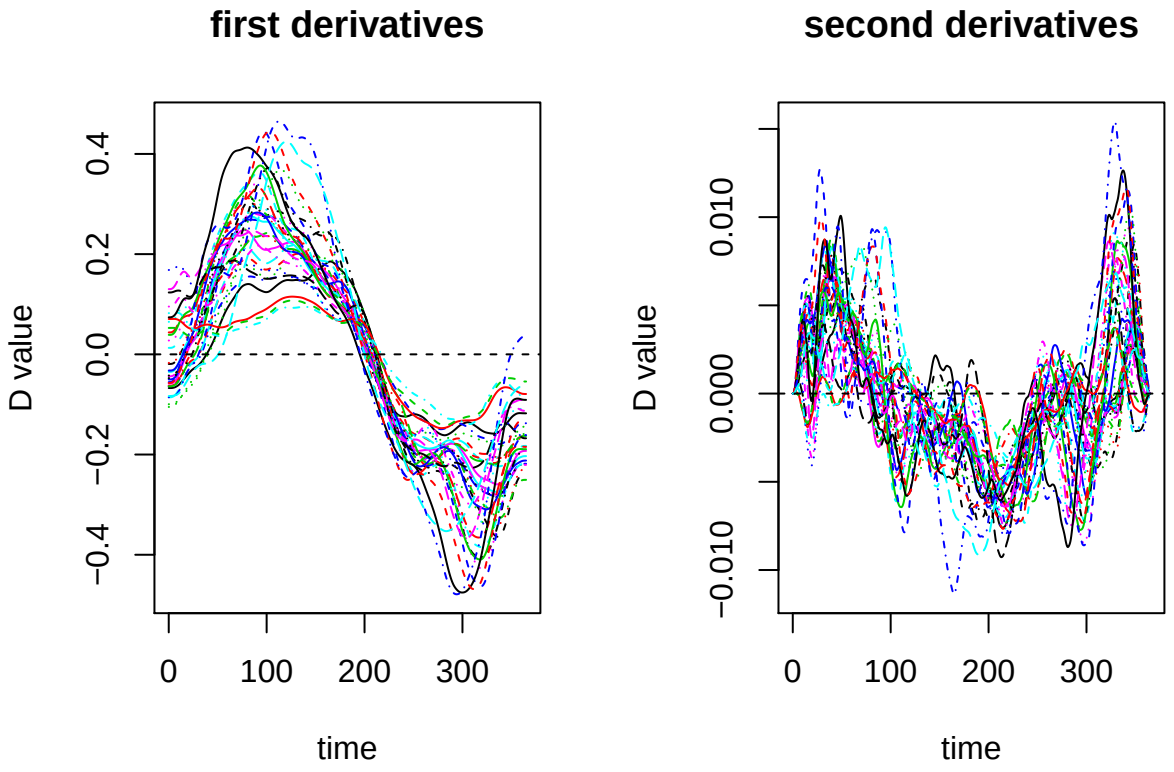
```r
# plot the original observations over the timeline
matplot(y, type="l", main=" temperature observations for 35 Canadian cities", xlab="days", ylab="temp C
```




```r
# choose smoothing technique for the daily temp data y,
bbasis.y <- create.bspline.basis(rangeval= c(0, 365), nbasis = 367, norder= 4)
penfd.y <- fdPar(bbasis.y,Lfdobj=int2Lfd(2),lambda= 10^{4})
smooth.y <- smooth.basis(1:365, y , penfd.y)
# plot the smoothed curves after applying the basis functions
plot(smooth.y ,main="smoothed curves of the daily tempreture curves", xlab="over a year", ylab="temp C"
```




```
# find the first and the second derivatives of the smoothed curves:
deriv1.y <- deriv.fd(smooth.y$fd, 1)
deriv2.y <- deriv.fd(smooth.y$fd, 2)
# plot the first and the second derivatives to examine them
par(mfrow=c(1,2))
plot(deriv1.y, main="first derivatives") # refers to rate of change in temperature
plot(deriv2.y, main="second derivatives") # refers to accelaration in temperature
```



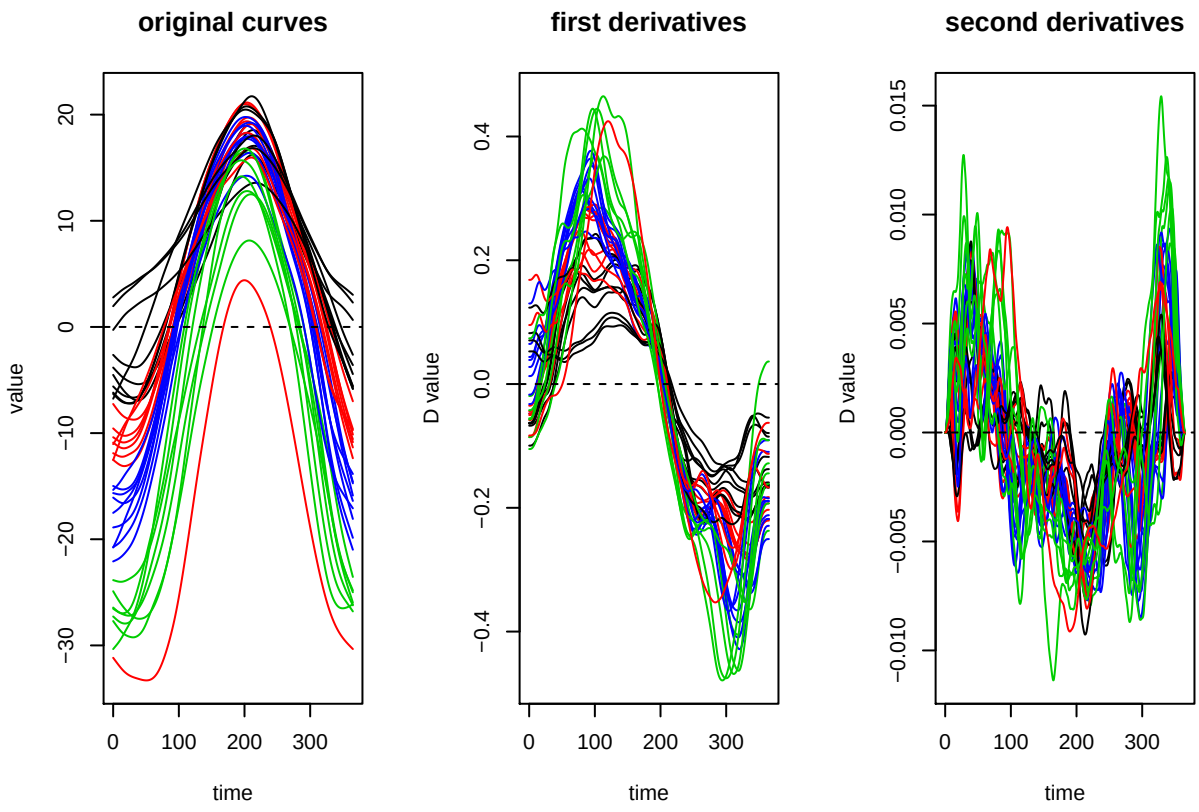
Run the function

```
FSC.fun <- function(matrix.data= matrix.data , t= NULL , basis= basis,  nclusters= NULL, d=d){

  if(class(matrix.data)=="matrix"){

    ifelse(missing(t) , t <- (1:ncol(matrix.data)), t<- t)
    ifelse(missing(nclusters) , print("Please choose number of clusters"), nclusters <- nclusters)

    # find the number of curves
    n <- dim(matrix.data)[2]
    # smooth the data according to the supplied basis
    smoothedfd <- smooth.basis( t , matrix.data, basis)
    deriv1 <- deriv.fd(smoothedfd$fd, 1) # for plotting purposes
    deriv2 <- deriv.fd(smoothedfd$fd, 2) # for plotting purposes

    # evaluate the smoothed data
    evaluatedata <- eval.fd(t, as.fd(smoothedfd) )
    # calculate the distance using fda.usc metric.lp() measure
    fobject <- fdata(t(evaluatedata))
    # assign the type of clustering according to d
    if(d==0){
      M <- metric.lp(fobject)
    }else if(d==1){
      M <- semimetric.deriv(fobject, nderiv = 1)
    }else if(d==2){
      M <- semimetric.deriv(fobject, nderiv = 2)
    }else{
```

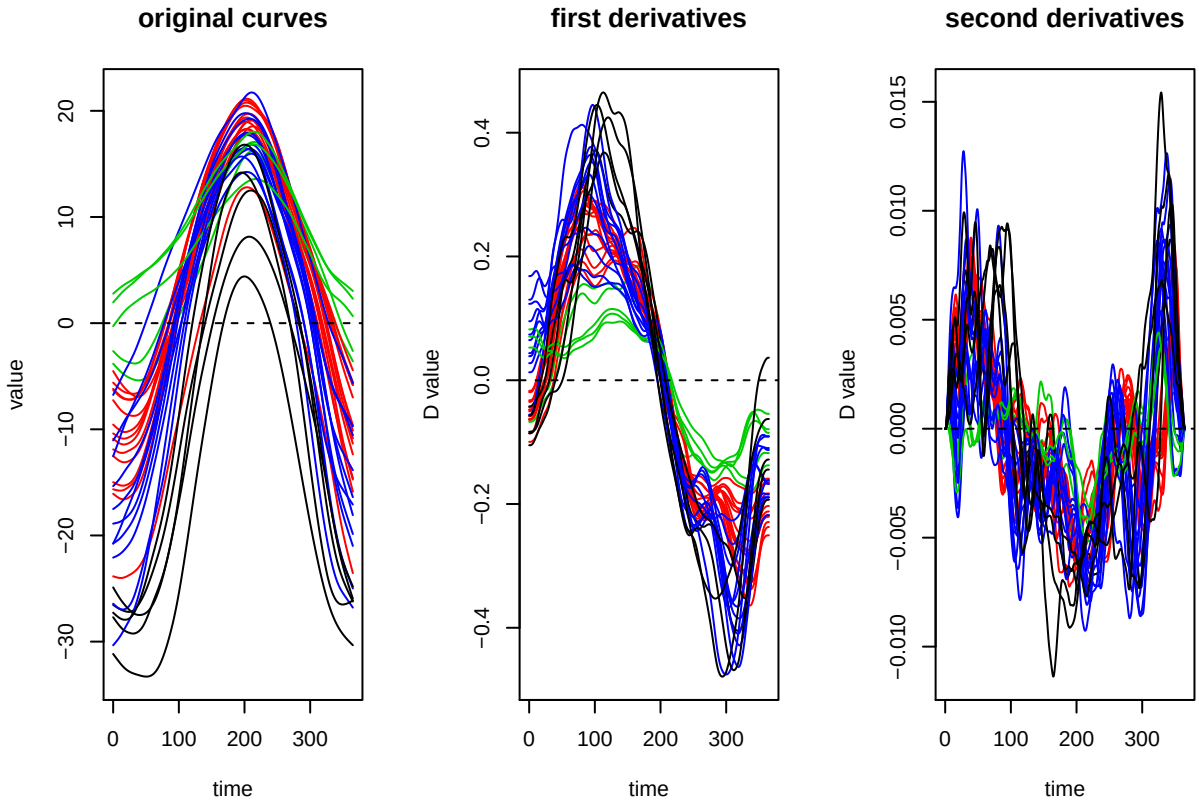
```
      print("Warning: d must take the value: 0 or 1 or 2.")
    }
    # calculate sd of the elements of M
    sdM <- sd(M)
    # form the afinity matrix
    A <- exp(-M/(2*sdM))
    # form the Diagonal Matrix D, where D = sum of W across rows
    D <- diag(apply(A,1,sum), n, n)
    # form the L matrix, such that L=D*AD*, where *=-1/2
    L <- diag(1/sqrt(diag(D))) %*% A %*% diag(1/sqrt(diag(D)))
    # form the matrix X from the eigenvectors
    X <- eigen(L, symmetric = TRUE)$vectors[,1:nclusters]
    # form Y from X
    Y <- X/sqrt(rowSums(X^2))
    # Cluster Y rows using (k-means)
    kmY <- kmeans(Y, centers = nclusters, iter.max = 100L, nstart = 1000L)
    # get the clusters members
    cls.kmY <- kmY$cluster

    # to plot the curves in their clusters
    par(mfrow=c(1,3))
    plot(smoothedfd, col=cls.kmY ,lwd=1,lty=1, main="original curves")
    plot(deriv1, col=cls.kmY ,lwd=1,lty=1, main= "first derivatives")
    plot(deriv2, col=cls.kmY ,lwd=1,lty=1, main="second derivatives")

    return(cls.kmY)

  }else{
    print("the data must be in a matrix format")
  }

}
```

According to he geographical distribution of the Canadian cities, we will assume the true number of clusters is 4 as suggested in (Ramsay, and Silverman, 2005). Thus we will cluster the data as follow:

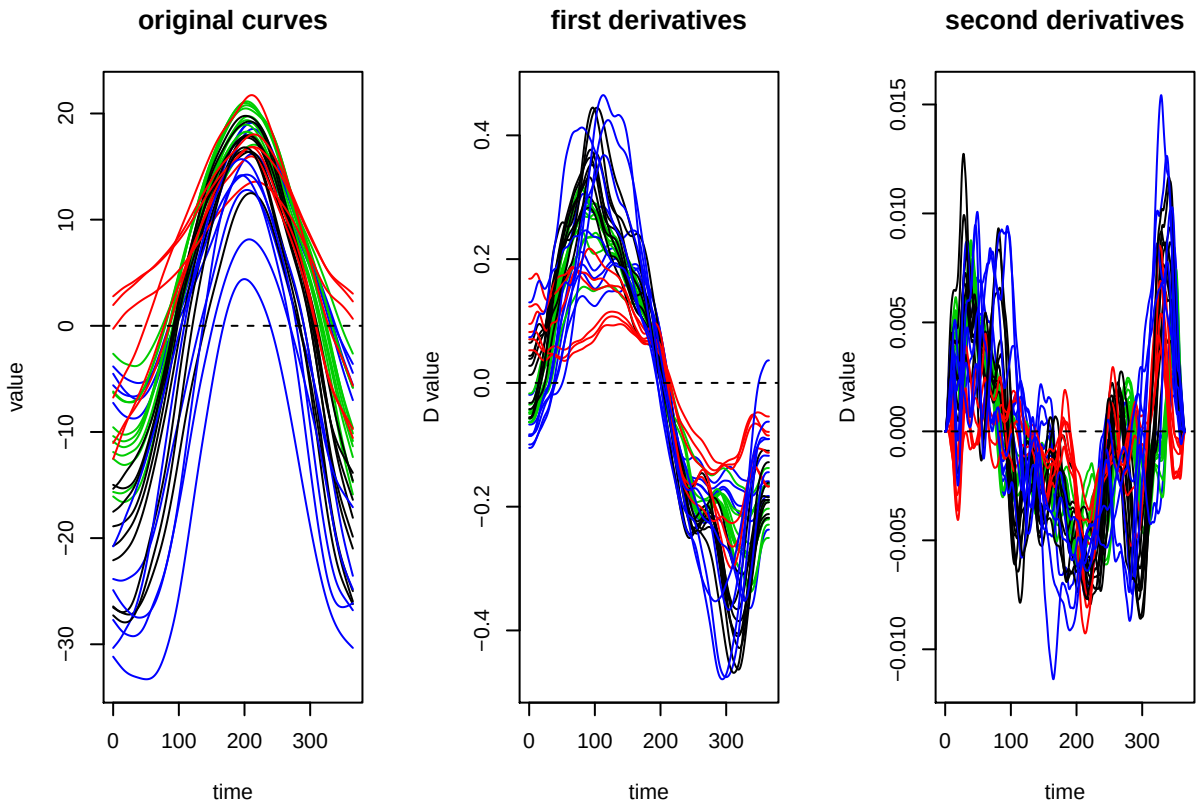
```
# using original curves for clustering
clust.d0 <- FSC.fun(y, t, penfd.y, 4, 0)
```




```
# using first derivatives for clustering
clust.d1 <- FSC.fun(y, t, penfd.y, 4, 1)
```



```r
# using second derivatives for clustering
clust.d2 <- FSC.fun(y, t, penfd.y, 4, 2)
```



According to the accuracy rate for the above results, FSC with d=1 outperforms the other methods. It worth mentioning that, the first derivative of the Canadian weather data shows clear variations in the amplitude and phase between groups of curves.

Note

- **Smoothing stage**: The defualt smoothing technique in the FSC approach is B-spline bases of order 4 with a knot at every time-point in the data (i.e. a saturated model). With fitting a smoothing parameter to penalize the representation of the data, the smoothing parameter is selected through cross-validation. Note that a B-spline of order n is equivalent to polynomial of degree n-1, thus B-spline of order n gives up to n-2 continuous derivatives. It is of importance to choose basis function that can give at least 2 continuous derivatives (first derivatives, and second derivatives).
- **Distance metric**: FSC is a distance-based clustering method. The distance matrix is calculated by measuring the distances between the original trajectories when $d = 0$. While, the distance matrix is calculated by measuring the distances between the first derivatives when $d = 1$. Similarily, the distance matrix is calculated by measuring the distances between the second derivatives when $d = 2$. The user can choose the distance matrix according to his/her perception which set holds more information about the data.

Source

Al Alawi, M., Ray, S. and Gupta, M. (2019) A New Framework for Distance-based Functional Clustering. In: 34th International Workshop on Statistical Modelling, Guimarães, Portugal, 07-12 Jul 2019,

Authors

- **Maryam Al Alawi**
- **Surajit Ray**
- **Mayetri Gupta**