Package ‘clustEff’

October 12, 2022

Type Package

Title Clusters of Effects Curves in Quantile Regression Models

Version 0.3.0

Description Clustering method to cluster both effects curves, through quantile regression coefficient modeling, and curves in functional data analysis. Sottile G. and Adelfio G. (2019) <doi:10.1007/s00180-018-0817-8>.

Depends qrcm, cluster, fda, ggpubr, ggplot2

License GPL-2

Encoding UTF-8

RoxygenNote 7.2.0

NeedsCompilation no

Author Gianluca Sottile [aut, cre],
Giada Adelfio [aut]

Maintainer Gianluca Sottile <`gianluca.sottile@unipa.it`>

Repository CRAN

Date/Publication 2022-06-28 11:50:02 UTC

R topics documented:

`clustEff-package` *Clusters of effects curves*

Description

This package implements a general algorithm to cluster coefficient functions (i.e. clusters of effects) obtained from a quantile regression (qrcm; Frumento and Bottai, 2016). This algorithm is also used for clustering curves observed in time, as in functional data analysis. The objectives of this algorithm vary with the scenario in which it is used, i.e. in the case of a cluster of effects, in a univariate case the objective may be to reduce its dimensionality or in the multivariate case to group similar effects on a covariate. In the case of a functional data analysis the main objective is to cluster waves or any other function of time or space. Sottile G. and Adelfio G. (2019) <https://doi.org/10.1007/s00180-018-0817-8>.

Details

Package:	clustEff
Type:	Package
Version:	0.3.0
Date:	2022-06-28
License:	GPL-2

The function `clustEff` allows to specify the type of the curves to apply the proposed clustering algorithm. The function `extract.object` extracts the matrices, in case of multivariate response, through the quantile regression coefficient modeling, useful to run the main algorithm. The auxiliary functions `summary.clustEff` and `plot.clustEff` can be used to extract information from the main algorithm. In the new version of the package you can also find a PCA-based clustering approach called Functional Principal Components Analysis Clustering (FPCAC). Main function of this algorithm is `fpcac`, and some auxiliary functions are `summary.fpcac` and `plot.fpcac`.

Author(s)

Gianluca Sottile

Maintainer: Gianluca Sottile <gianluca.sottile@unipa.it>

References

Sottile, G., Adelfio, G. *Clusters of effects curves in quantile regression models*. Comput Stat 34, 551–569 (2019). https://doi.org/10.1007/s00180-018-0817-8

Sottile, G and Adelfio, G (2017). *Clustering of effects through quantile regression*. Proceedings 32nd International Workshop of Statistical Modeling, Groningen (NL), vol.2 127-130, https://iwsm2017.webhosting.rug.nl/I

Frumento, P., and Bottai, M. (2015). *Parametric modeling of quantile regression coefficient functions*. Biometrics, doi: 10.1111/biom.12410.

Adelfio, G., Chiodi, M., D'Alessandro, A. and Luzio, D. (2011) *FPCA algorithm for waveform clustering*. Journal of Communication and Computer, 8(6), 494-502.

Examples

```
# Main functions:
set.seed(1234)
n <- 300
x <- 1:n/n

Y <- matrix(0, n, 30)

sigma2 <- 4*pmax(x-.2, 0) - 8*pmax(x-.5, 0) + 4*pmax(x-.8, 0)

mu <- sin(3*pi*x)
for(i in 1:10) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- cos(3*pi*x)
for(i in 11:23) Y[,i] <- mu + rnorm(length(x), 0, pmax(sigma2,0))

mu <- sin(3*pi*x)*cos(pi*x)
for(i in 24:28) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- 0 #sin(1/3*pi*x)*cos(2*pi*x)
for(i in 29:30) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

clustEff(Y)

fpcac(Y, K = opt.fpcac(Y)$K.opt)
```

`clustEff` *Cluster Effects Algorithm*

Description

This function implements the algorithm to cluster curves of effects obtained from a quantile regression (qrcm; Frumento and Bottai, 2015) in which the coefficients are described by flexible parametric functions of the order of the quantile. This algorithm can be also used for clustering of curves observed in time, as in functional data analysis.

Usage

```
clustEff(Beta, Beta.lower = NULL, Beta.upper = NULL,
          k = c(2, min(5, (ncol(Beta)-1))), ask = FALSE, diss.mat, alpha = .5,
          step = c("both", "shape", "distance"),
          cut.method = c("mindist", "length", "conf.int"),
          method = "ward.D2", approx.spline = FALSE, nbasis = 50,
          conf.level = 0.9, stand = FALSE, plot = TRUE, trace = TRUE)
```

Arguments

`Beta`	A matrix `n x q`. `q` represents the number of curves to cluster and `n` is either the length of percentiles used in the quantile regression or the length of the time vector.
`Beta.lower`	A matrix `n x q`. `q` represents the number of lower interval of the curves to cluster and `n` the length of percentiles used in quantile regression. Used only if cluster.effects=TRUE.
`Beta.upper`	A matrix `n x q`. `q` represents the number of upper interval of the curves to cluster and `n` the length of percentiles used in quantile regression. Used only if cluster.effects=TRUE.
`k`	It represents the number of clusters to look for. If it is two-length vector (k.min - k.max) an optimization is performed, if it is a unique value it is fixed.
`ask`	If TRUE, after plotting the dendrogram, the user make is own choice about how many cluster to use.
`diss.mat`	a dissimilarity matrix, obtained by using distshape function.
`alpha`	It is the alpha-percentile used for computing the dissimilarity matrix. The default value is alpha=.5.
`step`	The steps used in computing the dissimilarity matrix. Default is "both"=("shape" and "distance")
`cut.method`	The method used in optimization step to look for the optimal number of clusters. Default is "mindist", however if Beta.lower and Beta.upper are available the suggested method is "conf.int".
`method`	The agglomeration method to be used.
`approx.spline`	If TRUE, Beta is approximated by a smooth spline.
`nbasis`	An integer variable specifying the number of basis functions. Only when approx.spline=TRUE
`conf.level`	the confidence level required.
`stand`	If TRUE, the argument Beta is standardized.
`plot`	If TRUE, dendrogram, boxplot and clusters are plotted.
`trace`	If TRUE, some informations are printed.

Details

Quantile regression models conditional quantiles of a response variabile, given a set of covariates. Assume that each coefficient can be expressed as a parametric function of p in the form:

$$\beta(p|\theta) = \theta_0 + \theta_1 b_1(p) + \theta_2 b_2(p) + \ldots$$

where $b_1(p), b_2(p, \ldots)$ are known functions of p.

Value

An object of class “`clustEff`”, a list containing the following items:

`call`	the matched call.
`p`	The percentiles used in quantile regression coefficient modeling or the time otherwise.
`X`	The curves matrix.
`clusters`	The vector of clusters.
`X.mean`	The mean curves matrix of dimension `n` x `k`.
`X.mean.dist`	The within cluster distance from the mean curve.
`X.lower`	The lower bound matrix.
`X.mean.lower`	The mean lower bound of dimension `n` x `k`.
`X.upper`	The upper bound matrix.
`X.mean.upper`	The mean upper bound of dimension `n` x `k`.
`Signif.interval`	The matrix of dimension `n` x `k` containing the intervals in which each mean lower and upper bounds don’t include the zero.
`k`	The number of selected clusters.
`diss.matrix`	The dissimilarity matrix.
`X.mean.diss`	The within cluster dissimilarity.
`oggSilhouette`	An object of class “`silhouette`”.
`oggHclust`	An object of class “`hclust`”.
`distance`	A vector of goodness measures used to select the best number of clusters.
`step`	The selected step.
`method`	The used agglomeration method.
`cut.method`	The used method to select the best number of clusters.
`alpha`	The selected alpha-percentile.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.it`>

References

Sottile, G., Adelfio, G. Clusters of effects curves in quantile regression models. Comput Stat 34, 551–569 (2019). https://doi.org/10.1007/s00180-018-0817-8

Sottile, G and Adelfio, G (2017). *Clustering of effects through quantile regression*. Proceedings 32nd International Workshop of Statistical Modeling, Groningen (NL), vol.2 127-130, https://iwsm2017.webhosting.rug.nl/IV

Frumento, P., and Bottai, M. (2015). *Parametric modeling of quantile regression coefficient functions*. Biometrics, doi: 10.1111/biom.12410.

See Also

`summary.clustEff`, `plot.clustEff`, for summary and plotting. `extract.object` to extract useful objects for the clustering algorithm through a quantile regression coefficient modeling in a multivariate case.

Examples

```
# CURVES EFFECTS CLUSTERING

set.seed(1234)
n <- 300
q <- 2
k <- 5
x1 <- runif(n, 0, 5)
x2 <- runif(n, 0, 5)

X <- cbind(x1, x2)
rownames(X) <- 1:n
colnames(X) <- paste0("X", 1:q)

theta1 <- matrix(c(1, 1, 0, 0, 0, .5, 0, .5, 1, 2, .5, 0, 2, 1, .5),
                 ncol=k, byrow=TRUE)

theta2 <- matrix(c(1, 1, 0, 0, 0, -.3, 0, .5, 1, .5, -1.5, 0, -1, -.5, 1),
                 ncol=k, byrow=TRUE)

theta3 <- matrix(c(1, 1, 0, 0, 0, .3, 0, -.5, -1, 2, -.5, 0, 1, -.5, -1),
                 ncol=k, byrow=TRUE)

rownames(theta3) <- rownames(theta2) <- rownames(theta1) <-
    c("(intercept)", paste("X", 1:q, sep=""))
colnames(theta3) <- colnames(theta2) <- colnames(theta1) <-
    c("(intercept)", "qnorm(p)", "p", "p^2", "p^3")

Theta <- list(theta1, theta2, theta3)

B <- function(p, k){matrix(cbind(1, qnorm(p), p, p^2, p^3), nrow=k, byrow=TRUE)}
Q <- function(p, theta, B, k, X){rowSums(X * t(theta %*% B(p, k)))}

Y <- matrix(NA, nrow(X), 15)
for(i in 1:15){
  if(i <= 5) Y[, i] <- Q(runif(n), Theta[[1]], B, k, cbind(1, X))
  if(i <= 10 & i > 5) Y[, i] <- Q(runif(n), Theta[[2]], B, k, cbind(1, X))
  if(i <= 15 & i > 10) Y[, i] <- Q(runif(n), Theta[[3]], B, k, cbind(1, X))
}

XX <- extract.object(Y, X, intercept=TRUE, formula.p= ~ I(p) + I(p^2) + I(p^3))

obj <- clustEff(XX$X$X1, Beta.lower=XX$Xl$X1, Beta.upper=XX$Xr$X1, cut.method = "conf.int")
summary(obj)
plot(obj, xvar="clusters", col = 1:3)
```

```
plot(obj, xvar="dendrogram")
plot(obj, xvar="boxplot")

obj2 <- clustEff(XX$X$X2, Beta.lower=XX$Xl$X2, Beta.upper=XX$Xr$X2, cut.method = "conf.int")
summary(obj2)
plot(obj2, xvar="clusters", col=1:3)
plot(obj2, xvar="dendrogram")
plot(obj2, xvar="boxplot")


## Not run:
set.seed(1234)
n <- 300
q <- 15
k <- 5
X <- matrix(rnorm(n*q), n, q); X <- scale(X)
rownames(X) <- 1:n
colnames(X) <- paste0("X", 1:q)

Theta <- matrix(c(1, 1, 0, 0, 0,
                  .5, 0, .5, 1, 1,
                  .5, 0, 1, 2, .5,
                   .5, 0, 1, 1, .5,
                  .5, 0, .5, 1, 1,
                   .5, 0, .5, 1, .5,
                 -1.5, 0, -.5, 1, 1,
                  -1, 0, .5, -1, -1,
                 -.5, 0, -.5, -1, .5,
                  -1, 0, .5, -1, -.5,
                -1.5, 0, -.5, -1, -.5,
                  2, 0, 1, 1.5, 2,
                  2, 0, .5, 1.5, 2,
                  2.5, 0, 1, 1, 2,
                  1.5, 0, 1.5, 1, 2,
                  3, 0, 2, 1, .5),
                 ncol=k, byrow=TRUE)
rownames(Theta) <- c("(intercept)", paste("X", 1:q, sep=""))
colnames(Theta) <- c("(intercept)", "qnorm(p)", "p", "p^2", "p^3")

B <- function(p, k){matrix(cbind(1, qnorm(p), p, p^2, p^3), nrow=k, byrow=TRUE)}
Q <- function(p, theta, B, k, X){rowSums(X * t(theta %*% B(p, k)))}

s <- matrix(1, q+1, k)
s[2:(q+1), 2] <- 0
s[1, 3:k] <- 0

Y <- Q(runif(n), Theta, B, k, cbind(1, X))
XX <- extract.object(Y, X, intercept = TRUE, formula.p= ~ I(p) + I(p^2) + I(p^3))

obj3 <- clustEff(XX$X, Beta.lower=XX$Xl, Beta.upper=XX$Xr, cut.method = "conf.int")
summary(obj3)

# changing the alpha-percentile clusters are correctly identified
```

```
obj4 <- clustEff(XX$X, Beta.lower=XX$Xl, Beta.upper=XX$Xr, cut.method = "conf.int",
                 alpha = 0.25)
summary(obj4)

# CURVES CLUSTERING IN FUNCTIONAL DATA ANALYSIS

set.seed(1234)
n <- 300
x <- 1:n/n

Y <- matrix(0, n, 30)

sigma2 <- 4*pmax(x-.2, 0) - 8*pmax(x-.5, 0) + 4*pmax(x-.8, 0)

mu <- sin(3*pi*x)
for(i in 1:10) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- cos(3*pi*x)
for(i in 11:23) Y[,i] <- mu + rnorm(length(x), 0, pmax(sigma2,0))

mu <- sin(3*pi*x)*cos(pi*x)
for(i in 24:28) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- 0 #sin(1/3*pi*x)*cos(2*pi*x)
for(i in 29:30) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

obj5 <- clustEff(Y)
summary(obj5)
plot(obj5, xvar="clusters", col=1:4)
plot(obj5, xvar="dendrogram")
plot(obj5, xvar="boxplot")

## End(Not run)
```

`distshape` *Dissimilarity matrix*

Description

This function implements the dissimilarity matrix based on shape and distance of curves.

Usage

```
distshape(Beta, alpha=.5, step=c("both", "shape", "distance"), trace=TRUE)
```

Arguments

`Beta`	A matrix `n x q`. `q` represents the number of curves to cluster and `n` is either the length of percentiles used in the quantile regression or the length of the time vector.
`alpha`	It is the alpha-percentile used for computing the dissimilarity matrix. If not fixed, the algorithm choose alpha=.25 (cluster.effects=TRUE) or alpha=.5 (cluster.effects=FALSE).
`step`	The steps used in computing the dissimilarity matrix. Default is "both"=("shape" and "distance")
`trace`	If TRUE, some informations are printed.

Value

The dissimilarity matrix of class “`dist`”.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.it`>

References

Sottile, G., Adelfio, G. Clusters of effects curves in quantile regression models. Comput Stat 34, 551–569 (2019). https://doi.org/10.1007/s00180-018-0817-8

Sottile, G and Adelfio, G (2017). *Clustering of effects through quantile regression*. Proceedings 32nd International Workshop of Statistical Modeling, Groningen (NL), vol.2 127-130, https://iwsm2017.webhosting.rug.nl/IV

Frumento, P., and Bottai, M. (2015). *Parametric modeling of quantile regression coefficient functions*. Biometrics, doi: 10.1111/biom.12410.

See Also

`clustEff`,`summary.clustEff`, `plot.clustEff`, for summary and plotting. `extract.object` to extract useful objects for the clustering algorithm through a quantile regression coefficient modeling in a multivariate case.

Examples

```
set.seed(1234)
n <- 300
x <- 1:n/n

Y <- matrix(0, n, 30)

sigma2 <- 4*pmax(x-.2, 0) - 8*pmax(x-.5, 0) + 4*pmax(x-.8, 0)

mu <- sin(3*pi*x)
for(i in 1:10) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- cos(3*pi*x)
```

```
for(i in 11:23) Y[,i] <- mu + rnorm(length(x), 0, pmax(sigma2,0))

mu <- sin(3*pi*x)*cos(pi*x)
for(i in 24:28) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- 0 #sin(1/3*pi*x)*cos(2*pi*x)
for(i in 29:30) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

diss <- distshape(Y)
diss
```

`extract.object` — `extract.object` *fits a multivariate quantile regression and extracts objects for the cluster effects algorithm.*

Description

`extract.object` fits a multivariate quantile regression and extracts objects for the cluster effects algorithm.

Usage

```
extract.object(Y, X, intercept=TRUE, formula.p=~slp(p, 3), s, object, p, which)
```

Arguments

`Y`	A multivariate response matrix of dimension `n` x `q1`, or a vector of length `n`.
`X`	The covariates matrix of dimension `n` x `q2`.
`intercept`	If TRUE, the intercept is included in the model.
`formula.p`	a one-sided formula of the form ~ `b1(p, ...)` + `b2(p, ...)` + `...`
`s`	An optional 0/1 matrix that allows to exclude some model coefficients (see 'Examples').
`object`	An object of class "`iqr`". If missing, Y and X have to be supplied.
`p`	The percentiles used in quantile regression coefficient modeling. If missing a default sequence is choosen.
`which`	If fixed, only the selected covariates are extraced from the model. If missing all the covariates are extracted.

Details

A list of objects useful to run the cluster effect algorithm is created.

Value

`p`	The percentiles used in the quantile regression.
`X`	A list containing as many matrices as covariates, where for each matrix the number of columns corresponds to the number of the responses. Each column of a matrix corresponds to one curve effect. In the case of a univariate model it is a unique matrix.
`Xl`	A list as `X`. Each column of a matrix corresponds to the lower interval of the curve effect. In the case of a univariate model it is a unique matrix.
`Xr`	A list as `X`. Each column of a matrix corresponds to the upper interval of the curve effect. In the case of a univariate model it is a unique matrix.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.it`>

See Also

`clustEff`, for clustering algorithm; `summary.clustEff` and `plot.clustEff`, for summarizing and plotting `clustEff` objects.

Examples

```
# using simulated data

# see the documentation for 'clustEff'
```

`fpcac` *Functional Principal Components Analysis Clustering*

Description

This function implements the algorithm FPCAC for curves clustering as a variant of a k-means algorithm based on the principal component rotation of data

Usage

```
fpcac(X, K = 2, fd = NULL, nbasis = 5, norder = 3, nharmonics = 3,
      alpha = 0, niter = 30, Ksteps = 25, conf.level = 0.9, seed, disp = FALSE)
```

Arguments

`X`	Matrix of ‘curves’ of dimension `n x q`.
`K`	the number of clusters.
`fd`	If not NULL it overrides X and must be an object of class fd.
`nbasis`	an integer variable specifying the number of basis functions. The default value is 5.
`norder`	an integer specifying the order of b-splines, which is one higher than their degree. The default value is 3.
`nharmonics`	the number of harmonics or principal components to use. The default value is 3.
`alpha`	trimming size, that is the given proportion of observations to be discarded.
`niter`	the number or random restarting (larger values provide more accurate solutions.
`Ksteps`	the number of k-mean steps (not too many ksteps are needed).
`conf.level`	the confidence level required.
`seed`	the seed used for reproducibility.
`disp`	if TRUE, it is used to print some information across the algorithm.

Details

FPCAC is a functional PCA-based clustering approach that provides a variation of the algorithm for curves clustering proposed by Garcia-Escudero and Gordaliza (2005).

The starting point of the proposed FPCAC is to find a linear approximation of each curve by a finite \$p\$ dimensional vector of coefficients defined by the FPCA scores.

The number of starting clusters `k` is obtained on the basis of the scores volume, such that we assign events to the clusters defined by events that have a distance less than a fixed threshold (e.g. 90-th percentile) in the space of PCA scores. Once `k` is obtained we use a modified version of the trimmed `k`-means algorithm, that considers the matrix of FPCA scores instead of the coefficients of a linear fitting to B-spline bases.

The trimmed `k`-means clustering algorithm looks for the `k` centers $C_1, ..., C_k$ that are solution of the minimization problem:

$$O_k(\alpha) = \min_{Y} \min_{C_1,\cdots,C_k} \frac{1}{[n(1-\alpha)]} \sum_{X_i \in Y} \inf_{1 \leq j \leq k} ||X_i - C_j||^2$$

We think that the proposed approach has the advantage of an immediate use of PCA for functional data avoiding some objective choices related to spline fitting as in RCC. Simulations and applications suggest also the well behavior of the FPCAC algorithm, both in terms of stable and easily interpretable results.

Value

An object of class “`fpcac`”, a list containing the following items:

`call`	the matched call.

`obj.function`	The percentiles used in the quantile regression coefficient modeling or objective function `O_k(\alpha)`.
`centers`	The curves matrix.
`radius`	The vector of clusters.
`clusters`	The mean curves matrix of dimension `n` x `k`.
`Xorig`	The atrix of ‘curves’ of dimension `n` x `q`.
`fd`	The object obtained by the call of FPCA of class ‘fd’
`X`	The matrix of ‘curves’ transformed through FPCA of dimension `p` x `nharmonics`.
`X.mean`	The mean curves matrix of dimension `n` x `k`.
`diss.matrix`	The Euclidean distance matrix of the transformed curves.
`oggSilhouette`	An object of class ‘silhouette’.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.it`>

References

Adelfio, G., Chiodi, M., D’Alessandro, A. and Luzio, D. (2011) FPCA algorithm for waveform clustering. Journal of Communication and Computer, 8(6), 494-502.

Adelfio, G., Chiodi, M., D’Alessandro, A., Luzio, D., D’Anna, G., Mangano, G. (2012) Simultaneous seismic wave clustering and registration. Computers & Geosciences 44, 60-69.

Garcia-Escudero, L. A. and Gordaliza, A. (2005). A proposal for robust curve clustering, Journal of classification, 22, 185-201.

See Also

`opt.fpcac`.

Examples

```
set.seed(1234)
n <- 300
x <- 1:n/n

Y <- matrix(0, n, 30)

sigma2 <- 4*pmax(x-.2, 0) - 8*pmax(x-.5, 0) + 4*pmax(x-.8, 0)

mu <- sin(3*pi*x)
for(i in 1:10) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- cos(3*pi*x)
for(i in 11:23) Y[,i] <- mu + rnorm(length(x), 0, pmax(sigma2,0))

mu <- sin(3*pi*x)*cos(pi*x)
for(i in 24:28) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))
```

```
mu <- 0 #sin(1/3*pi*x)*cos(2*pi*x)
for(i in 29:30) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

obj <- fpcac(Y, K = 4, disp = FALSE)
obj
```

`opt.fpcac` *Optimal cluster selection in Functional Principal Components Analysis Clustering*

Description

This function provides the optimal selection of clusters for the algorithm FPCAC, as a variant of a k-means algorithm based on the principal component rotation of data

Usage

```
opt.fpcac(X, k.max = 5, method = c("silhouette", "wss"),
          fd = NULL, nbasis = 5, norder = 3, nharmonics = 3,
          alpha = 0, niter = 30, Ksteps = 10, seed,
          diss = NULL, trace=FALSE)
```

Arguments

`X`	Matrix of ‘curves’ of dimension `n` x `q`.
`k.max`	the number of cluster used in the optimization step to select the optimal one.
`method`	the method used to select the optimal number of clusters, "silhouette" or "wss" (whithin sum of squares.
`fd`	If not NULL it overrides X and must be an object of class fd.
`nbasis`	an integer variable specifying the number of basis functions. The default value is 5.
`norder`	an integer specifying the order of b-splines, which is one higher than their degree. The default value is 3.
`nharmonics`	the number of harmonics or principal components to use. The default value is 3.
`alpha`	trimming size, that is the given proportion of observations to be discarded.
`niter`	the number or random restarting (larger values provide more accurate solutions.
`Ksteps`	the number of k-mean steps (not too many ksteps are needed).
`seed`	the seed used for reproducibility.
`diss`	the dissimilarity matrix used to compute measures "silhouette" or "wss".
`trace`	if TRUE, it is used to print some information across the algorithm.

Details

Silhouette is a method for validate the consistency within clusters, providing a measure of how similar an object is to its own cluster compared to other clusters. The silhouette score S belongs to the interval [-1,1]. S close to one means that the data is appropriately clustered. If S is close to negative one, datum should be clustered in its neighbouring cluster. S near zero means that the datum is on the border of two natural clusters.

The wss is obtained as the classical sum of the squared deviations from each observation and the cluster centroid, providing a measure of the variability of the observations within each cluster. Clusters with higher values exhibit greater variability of the observations within the cluster.

Value

a list containing the following items:

`obj.function`	the sequence of objective functions.
`clusters`	the matrix in which each columns identify clusters for each fixed K.
`K`	the sequence of K used.
`K.opt`	the optimal number of clusters
`plot`	a ggplot object to plot the curve of silhouette or whithin sum of squares.

Author(s)

Gianluca Sottile `<gianluca.sottile@unipa.it>`

References

Peter J. Rousseeuw (1987). Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis. Computational and Applied Mathematics. 20, 53-65

K. V. Mardia, J. T. Kent and J. M. Bibby (1979). Multivariate Analysis. Academic Press.

See Also

`fpcac`.

Examples

```
set.seed(1234)
n <- 300
x <- 1:n/n

Y <- matrix(0, n, 30)

sigma2 <- 4*pmax(x-.2, 0) - 8*pmax(x-.5, 0) + 4*pmax(x-.8, 0)

mu <- sin(3*pi*x)
for(i in 1:10) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- cos(3*pi*x)
```

```
for(i in 11:23) Y[,i] <- mu + rnorm(length(x), 0, pmax(sigma2,0))

mu <- sin(3*pi*x)*cos(pi*x)
for(i in 24:28) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

mu <- 0 #sin(1/3*pi*x)*cos(2*pi*x)
for(i in 29:30) Y[, i] <- mu + rnorm(length(x), 0, pmax(sigma2, 0))

num.clust <- opt.fpcac(Y)
obj2 <- fpcac(Y, K = num.clust$K.opt, disp = FALSE)
obj2
```

`plot.clustEff` *Plot Clustering Effects*

Description

Produces a dendrogram, a cluster plot and a boxplot of average distance cluster of an object of class “`clustEff`”.

Usage

```
## S3 method for class 'clustEff'
plot(x, xvar=c("clusters", "dendrogram", "boxplot", "numclust"), which,
        polygon=TRUE, dissimilarity=TRUE, par=FALSE, ...)
```

Arguments

`x`	An object of class “`clustEdd`”, typically the result of a call to `clustEff`.
`xvar`	Clusters: plot of the k clusters; Dendrogram: plot of the tree after computing the dissimilarity measure and applying a hierarchical clustering algorithm; Boxplot: plot the average distance within clusters; Numclust: plot the curve to minimize to select the best number of clusters;
`which`	If missing all curves effect are plotted.
`polygon`	If TRUE confidence intervals are represented by shaded areas via polygon. Otherwise, dashed lines are used. If NULL no confidence intervals are represented
`dissimilarity`	If TRUE dissimilarity measure within each cluster is used to do boxplot representation.
`par`	If TRUE the screen is automaticcaly splitted.
`...`	additional graphical parameters, that can include `xlim, ylim, xlab, ylab, col, lwd, lty`. See `par`.

Details

Different plot for the clustering algorithm.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.ot`>

See Also

`clustEff` for cluster algorithm; `extract.object` for extracting information through a quantile regression coefficient modeling in a multivariate case; `summary.clustEff` for clustering summary.

Examples

```
# using simulated data

# see the documentation for 'clustEff'
```

`plot.fpcac` *Plot Functional Principal Component Analysis Clustering*

Description

Produces a cluster plot of an object of class “`fpcac`”.

Usage

```
## S3 method for class 'fpcac'
plot(x, which, polygon=TRUE, conf.level, ...)
```

Arguments

`x`	An object of class “`clustEdd`”, typically the result of a call to `fpcac`.
`which`	If missing all curves effect are plotted.
`polygon`	If TRUE confidence intervals are represented by shaded areas via polygon. Otherwise, dashed lines are used. If NULL no confidence intervals are represented
`conf.level`	the confidence level required.
`...`	additional graphical parameters, that can include `xlim, ylim, xlab, ylab, col, lwd, lty`. See `par`.

Details

Different plot for the clustering algorithm.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.ot`>

See Also

`fpcac`, `summary.fpcac`, `opt.fpcac`.

Examples

```
# using simulated data

# see the documentation for 'fpcac'
```

`summary.clustEff` *Summary clustEff algorithm*

Description

Summary of an object of class “`clustEff`”.

Usage

```
## S3 method for class 'clustEff'
summary(object, ...)
```

Arguments

`object`	An object of class “`clustEff`”, the result of a call to `clustEff`.
`...`	for future methods.

Details

A summary of the clustering algorithm is printed.

Value

The following items is returned:

`k`	The number of selected clusters.
`n`	The number of observations.
`p`	The number of curves.
`step`	The selected step for computing the dissimilarity matrix.
`alpha`	The alpha-percentile used for computing the dissimilarity matrix.
`method`	The selected method to compute the hierarchical cluster analysis.
`cut.method`	The selected method to choose the best number of clusters.
`tabClust`	The table of clusters.
`avClust`	The average distance within clusters.
`avSilhouette`	Silhouette widths for clusters.
`avDiss`	The average dissimilarity measure within clusters.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.it`>

See Also

`clustEff`, for cluster algorithm`extract.object` for extracting information through a quantile regression coefficient modeling in a multivariate case and plotting objects of class “`clustEff`”.

Examples

```
# using simulated data

# see the documentation for 'clustEff'
```

`summary.fpcac` *Summary FPCAC algorithm*

Description

Summary of an object of class “`fpcac`”.

Usage

```
## S3 method for class 'fpcac'
summary(object, ...)
```

Arguments

`object`	An object of class “`fpcac`”, the result of a call to `fpcac`.
`...`	for future methods.

Details

A summary of the clustering algorithm is printed.

Value

The following items is returned:

`k`	The number of selected clusters.
`n`	The number of curves.
`p`	The number of harmonics used.
`trimmed`	The number of trimmed curves.
`tabClust`	The table of clusters.
`avClust`	The average distance within clusters.

Author(s)

Gianluca Sottile <`gianluca.sottile@unipa.it`>

See Also

`fpcac`, `opt.fpcac`

Examples

```
# using simulated data

# see the documentation for 'fpcac'
```

Index