Package 'puls'

October 14, 2022

Title Partitioning Using Local Subregions

Version 0.1.2

Description A method of clustering functional data using subregion information of the curves. It is intended to supplement the 'fda' and 'fda.usc' packages in functional data object clustering. It also facilitates the printing and plotting of the results in a tree format and limits the partitioning candidates into a specific set of subregions.

License GPL (>= 2)

URL https://vinhtantran.github.io/puls/, https://github.com/vinhtantran/puls

BugReports https://github.com/vinhtantran/puls/issues

Depends R (>= 3.3.0)

Imports cluster (>= 2.0.5), dplyr (>= 1.0.0), fda, fda.usc (>= 1.3.0), ggplot2, graphics, monoClust (>= 1.2.0), purrr (>= 0.3.0), rlang (>= 0.3.0), stats, tibble (>= 3.0.0), tidyr (>= 1.0.0)

Suggests covr, knitr, lubridate, rmarkdown, testthat, vdiffr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Mark Greenwood [aut] (<https://orcid.org/0000-0001-6933-1201>), Tan Tran [aut, cre] (<https://orcid.org/0000-0001-9881-6339>)

Maintainer Tan Tran <vinhtantran@gmail.com>

Repository CRAN

Date/Publication 2021-02-16 16:40:06 UTC

R topics documented:

`arctic_2019` *NOAA's Arctic Sea Daily Ice Extend Data*

Description

A data set containing the daily ice extent at Arctic Sea from 1978 to 2019, collected by National Oceanic and Atmospheric Administration (NOAA)

Usage

```
arctic_2019
```

Format

A data frame with 13391 rows and 6 variables:

Year Years of available data (1978–2019).

Month Month (01–12).

Day Day of the month indicated in Column Month.

Extent Daily ice extent, to three decimal places.

Missing Whether a day is missing (1) or not (0)).

Source Data data source in NOAA database.

Source

https://nsidc.org/data/G02135/versions/3

Examples

```
library(dplyr)
library(lubridate)
library(ggplot2)

data(arctic_2019)

# Create day in the year column to replace Month and Day
north <-
  arctic_2019 %>%
  mutate(yday = yday(make_date(Year, Month, Day)),
         .keep = "all") %>%
  select(Year, yday, Extent)

ggplot(north) +
  geom_linerange(aes(x = yday, ymin = Year - 0.2, ymax = Year + 0.2),
                 size = 0.5, color = "red") +
  scale_y_continuous(breaks = seq(1980, 2020, by = 5),
                     minor_breaks = NULL) +
  labs(x = "Day",
       y = "Year",
       title = "Measurement frequencies were not always the same")
```

---

as_MonoClust.PULS *Coerce a PULS Object to MonoClust Object*

Description

An implementation of the `monoClust::as_MonoClust()` S3 method for PULS object. The purpose of this is to reuse plotting and printing functions from monoClust package.

Usage

```
## S3 method for class 'PULS'
as_MonoClust(x, ...)
```

Arguments

x	A PULS object to be coerced to MonoClust object.
...	For extensibility.

Value

A MonoClust object coerced from PULS object.

See Also

monoClust::MonoClust.object and PULS.object

`fdistmatrix` *Distance Between Functional Objects*

Description

Calculate the distance between functional objects over the defined range.

Usage

```
fdistmatrix(fd, subrange, distmethod)
```

Arguments

`fd`	A functional data object `fd` of `fda` package.
`subrange`	A vector of two values indicating the value range of functional object to calculate on.
`distmethod`	The method for calculating the distance matrix. Choose between `"usc"` and `"manual"`. `"usc"` uses `fda.usc::metric.lp()` function while `"manual"` uses squared distance between functions. See Details.

Details

If choosing `distmethod = "manual"`, the L2 distance between all pairs of functions $y_i(t)$ and $y_j(t)$ is given by:

$$d_R(y_i, y_j) = \sqrt{\int_{a_r}^{b_r} [y_i(t) - y_j(t)]^2 dt}.$$

Value

A distance matrix with diagonal value and the upper half.

Examples

```
library(fda)
# Examples taken from fda::Data2fd()
data(gait)
# Function only works on two dimensional data
gait <- gait[, 1:5, 1]
gaitbasis3 <- create.fourier.basis(nbasis = 5)
gaitfd3 <- Data2fd(gait, basisobj = gaitbasis3)

fdistmatrix(gaitfd3, c(0.2, 0.4), "usc")
```

ggwave *Plot the Partitioned Functional Wave by PULS*

Description

After partitioning using PULS, this function can plot the functional waves and color different clusters as well as their medoids.

Usage

```
ggwave(
  toclust.fd,
  intervals,
  puls.obj,
  xlab = NULL,
  ylab = NULL,
  lwd = 0.5,
  alpha = 0.4,
  lwd.med = 1
)
```

Arguments

toclust.fd	A functional data object (i.e., having class fd) created from fda package. See fda::fd().
intervals	A data set (or matrix) with rows are intervals and columns are the beginning and ending indexes of of the interval.
puls.obj	A PULS object as a result of PULS().
xlab	Labels for x-axis. If not provided, the labels stored in fd object will be used.
ylab	Labels for y-axis. If not provided, the labels stored in fd object will be used.
lwd	Linewidth of normal waves.
alpha	Transparency of normal waves.
lwd.med	Linewidth of medoid waves.

Value

A ggplot2 object.

Examples

```
library(fda)

# Build a simple fd object from already smoothed smoothed_arctic
data(smoothed_arctic)
NBASIS <- 300
```

```
NORDER <- 4
y <- t(as.matrix(smoothed_arctic[, -1]))
splinebasis <- create.bspline.basis(rangeval = c(1, 365),
                                    nbasis = NBASIS,
                                    norder = NORDER)
fdParobj <- fdPar(fdobj = splinebasis,
                  Lfdobj = 2,
                  # No need for any more smoothing
                  lambda = .000001)
yfd <- smooth.basis(argvals = 1:365, y = y, fdParobj = fdParobj)

Jan <- c(1, 31); Feb <- c(31, 59); Mar <- c(59, 90)
Apr <- c(90, 120); May <- c(120, 151); Jun <- c(151, 181)
Jul <- c(181, 212); Aug <- c(212, 243); Sep <- c(243, 273)
Oct <- c(273, 304); Nov <- c(304, 334); Dec <- c(334, 365)

intervals <-
  rbind(Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)

PULS4_pam <- PULS(toclust.fd = yfd$fd, intervals = intervals,
                  nclusters = 4, method = "pam")
ggwave(toclust.fd = yfd$fd, intervals = intervals, puls = PULS4_pam)
```

`plot.PULS` *Plot PULS Splitting Rule Tree*

Description

Print the PULS tree in the form of dendrogram.

Usage

```
## S3 method for class 'PULS'
plot(
  x,
  branch = 1,
  margin = c(0.12, 0.02, 0, 0.05),
  text = TRUE,
  which = 4,
  digits = getOption("digits") - 2,
  cols = NULL,
  col.type = c("l", "p", "b"),
  ...
)
```

Arguments

x	A PULS object.
branch	Controls the shape of the branches from parent to child node. Any number from 0 to 1 is allowed. A value of 1 gives square shouldered branches, a value of 0 give V shaped branches, with other values being intermediate.
margin	An extra fraction of white space to leave around the borders of the tree. (Long labels sometimes get cut off by the default computation).
text	Whether to print the labels on the tree.
which	Labeling modes, which are: • 1: only splitting variable names are shown, no splitting rules. • 2: only splitting rules to the left branches are shown. • 3: only splitting rules to the right branches are shown. • 4 (default): splitting rules are shown on both sides of branches.
digits	Number of significant digits to print.
cols	Whether to shown color bars at leaves or not. It helps matching this tree plot with other plots whose cluster membership were colored. It only works when text is TRUE. Either NULL, a vector of one color, or a vector of colors matching the number of leaves.
col.type	When cols is set, choose whether the color indicators are shown in a form of solid lines below the leaves ("l"), or big points ("p"), or both ("b").
...	Arguments to be passed to monoClust::plot.MonoClust().

Value

A plot of splitting order.

Examples

```
library(fda)

# Build a simple fd object from already smoothed smoothed_arctic
data(smoothed_arctic)
NBASIS <- 300
NORDER <- 4
y <- t(as.matrix(smoothed_arctic[, -1]))
splinebasis <- create.bspline.basis(rangeval = c(1, 365),
                                    nbasis = NBASIS,
                                    norder = NORDER)
fdParobj <- fdPar(fdobj = splinebasis,
                  Lfdobj = 2,
                  # No need for any more smoothing
                  lambda = .000001)
yfd <- smooth.basis(argvals = 1:365, y = y, fdParobj = fdParobj)

Jan <- c(1, 31); Feb <- c(31, 59); Mar <- c(59, 90)
Apr <- c(90, 120); May <- c(120, 151); Jun <- c(151, 181)
```

```
Jul <- c(181, 212); Aug <- c(212, 243); Sep <- c(243, 273)
Oct <- c(273, 304); Nov <- c(304, 334); Dec <- c(334, 365)

intervals <-
  rbind(Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)

PULS4_pam <- PULS(toclust.fd = yfd$fd, intervals = intervals,
                  nclusters = 4, method = "pam")
plot(PULS4_pam)
```

`print.PULS` *Print PULS Clustering Result*

Description

Render the PULS split tree in an easy to read format with important information such as terminal nodes, etc.

Usage

```
## S3 method for class 'PULS'
print(x, spaces = 2L, digits = getOption("digits"), ...)
```

Arguments

`x`	A `PULS` result object.
`spaces`	Spaces indent between 2 tree levels.
`digits`	Number of significant digits to print.
`...`	Arguments to be passed to `monoClust::print.MonoClust()`.

Value

A nicely displayed PULS split tree in text.

Examples

```
library(fda)

# Build a simple fd object from already smoothed smoothed_arctic
data(smoothed_arctic)
NBASIS <- 300
NORDER <- 4
y <- t(as.matrix(smoothed_arctic[, -1]))
splinebasis <- create.bspline.basis(rangeval = c(1, 365),
                                    nbasis = NBASIS,
                                    norder = NORDER)
```

```
fdParobj <- fdPar(fdobj = splinebasis,
                  Lfdobj = 2,
                  # No need for any more smoothing
                  lambda = .000001)
yfd <- smooth.basis(argvals = 1:365, y = y, fdParobj = fdParobj)

Jan <- c(1, 31); Feb <- c(31, 59); Mar <- c(59, 90)
Apr <- c(90, 120); May <- c(120, 151); Jun <- c(151, 181)
Jul <- c(181, 212); Aug <- c(212, 243); Sep <- c(243, 273)
Oct <- c(273, 304); Nov <- c(304, 334); Dec <- c(334, 365)

intervals <-
  rbind(Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)

PULS4_pam <- PULS(toclust.fd = yfd$fd, intervals = intervals,
                  nclusters = 4, method = "pam")
print(PULS4_pam)
```

PULS *Partitioning Using Local Subregions (PULS)*

Description

PULS function for functional data (only used when you know that the data shouldn't be converted into functional because it's already smooth, e.g. your data are step function)

Usage

```
PULS(
  toclust.fd,
  method = c("pam", "ward"),
  intervals = c(0, 1),
  spliton = NULL,
  distmethod = c("usc", "manual"),
  labels = toclust.fd$fdnames[2]$reps,
  nclusters = length(toclust.fd$fdnames[2]$reps),
  minbucket = 2,
  minsplit = 4
)
```

Arguments

`toclust.fd`	A functional data object (i.e., having class `fd`) created from `fda` package. See `fda::fd()`.
`method`	The clustering method you want to run in each subregion. Can be chosen between `pam` and `ward`.

intervals	A data set (or matrix) with rows are intervals and columns are the beginning and ending indexes of of the interval.
spliton	Restrict the partitioning on a specific set of subregions.
distmethod	The method for calculating the distance matrix. Choose between "usc" and "manual". "usc" uses fda.usc::metric.lp() function while "manual" uses squared distance between functions. See Details.
labels	The name of entities.
nclusters	The number of clusters.
minbucket	The minimum number of data points in one cluster allowed.
minsplit	The minimum size of a cluster that can still be considered to be a split candidate.

Details

If choosing distmethod = "manual", the L2 distance between all pairs of functions $y_i(t)$ and $y_j(t)$ is given by:

$$d_R(y_i, y_j) = \sqrt{\int_{a_r}^{b_r} [y_i(t) - y_j(t)]^2 dt}.$$

Value

A PULS object. See PULS.object for details.

See Also

fda::is.fd()

Examples

```
library(fda)

# Build a simple fd object from already smoothed smoothed_arctic
data(smoothed_arctic)
NBASIS <- 300
NORDER <- 4
y <- t(as.matrix(smoothed_arctic[, -1]))
splinebasis <- create.bspline.basis(rangeval = c(1, 365),
                                    nbasis = NBASIS,
                                    norder = NORDER)
fdParobj <- fdPar(fdobj = splinebasis,
                  Lfdobj = 2,
                  # No need for any more smoothing
                  lambda = .000001)
yfd <- smooth.basis(argvals = 1:365, y = y, fdParobj = fdParobj)

Jan <- c(1, 31); Feb <- c(31, 59); Mar <- c(59, 90)
Apr <- c(90, 120); May <- c(120, 151); Jun <- c(151, 181)
Jul <- c(181, 212); Aug <- c(212, 243); Sep <- c(243, 273)
Oct <- c(273, 304); Nov <- c(304, 334); Dec <- c(334, 365)
```

```
intervals <-
  rbind(Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)

PULS4_pam <- PULS(toclust.fd = yfd$fd, intervals = intervals,
                  nclusters = 4, method = "pam")
PULS4_pam
```

`PULS.object` *PULS Tree Object*

Description

The structure and objects contained in PULS, an object returned from the `PULS()` function and used as the input in other functions in the package.

Value

frame Data frame in the form of a `tibble::tibble()` representing a tree structure with one row for each node. The columns include:

- **number** Index of the node. Depth of a node can be derived by `number %/% 2`.
- **var** Name of the variable used in the split at a node or `"<leaf>"` if it is a leaf node.
- **n** Cluster size, the number of observations in that cluster.
- **wt** Weights of observations. Unusable. Saved for future use.
- **inertia** Inertia value of the cluster at that node.
- **bipartsplitrow** Position of the next split row in the data set (that position will belong to left node (smaller)).
- **bipartsplitcol** Position of the next split variable in the data set.
- **inertiadel** Proportion of inertia value of the cluster at that node to the inertia of the root.
- **medoid** Position of the data point regarded as the medoid of its cluster.
- **loc** y-coordinate of the splitting node to facilitate showing on the tree. See `plot.PULS()` for details.
- **inertia_explained** Percent inertia explained as described in Chavent (2007). It is `1 - (sum(current inertia)/inert`
- **alt** Indicator of an alternative cut yielding the same reduction in inertia at that split.

membership Vector of the same length as the number of rows in the data, containing the value of `frame$number` corresponding to the leaf node that an observation falls into.

dist Distance matrix calculated using the method indicated in `distmethod` argument of `PULS()`.

terms Vector of subregion names in the data that were used to split.

medoids Named vector of positions of the data points regarded as medoids of clusters.

alt Indicator of having an alternate splitting route occurred when splitting.

References

- Chavent, M., Lechevallier, Y., & Briant, O. (2007). DIVCLUS-T: A monothetic divisive hierarchical clustering method. Computational Statistics & Data Analysis, 52(2), 687-701. doi: 10.1016/j.csda.2007.03.013.

See Also

PULS().

`smoothed_arctic` *Discrete Form of Smoothed Functional Form of Arctic Data*

Description

Raw Arctic data were smoothed and then transformed into functional data using `fda` package. To overcome the difficulty of exporting an `fda` object in a package, the object was discretized into a data set with 365 columns corresponding to 365 days a year and 39 rows corresponding to 39 years. The years are from 1979 to 1986, then from 1989 to 2018. The years 1978, 1987, and 1988 were removed because the measurements were not complete.

Usage

```
smoothed_arctic
```

Format

A data frame with 39 rows corresponding to 39 years (1979 to 1986, 1989 to 2019) and 366 columns.

See Also

NOAA's raw data at arctic_2019 and the code to generate this data in data-raw/ folder of source code.

Index