Package ‘ctmva’

August 18, 2022

Type Package

Title Continuous-Time Multivariate Analysis

Version 1.0

Date 2022-08-06

Maintainer Biplab Paul <paul.biplab497@gmail.com>

Description Implements a basis function or functional data analysis framework for several techniques of multivariate analysis in continuous-time setting. Specifically, we introduced continuous-time analogues of several classical techniques of multivariate analysis, such as principal component analysis, canonical correlation analysis, Fisher linear discriminant analysis, K-means clustering, and so on. Details are in Philip T Reiss and Biplab Paul (2022) ``Continuous-time multivariate analysis"; James O Ramsay, Bernard W Silverman (2005) <ISBN:978-0-387-22751-1> ``Functional Data Analysis"; James O Ramsay, Giles Hooker and Spencer Graves (2009) <ISBN:978-0-387-98185-7> ``Functional Data Analysis with R and MATLAB".

License GPL (>= 2)

Imports fda, polynom

Suggests eegkit, corrplot

RoxygenNote 7.2.0

Author Biplab Paul [aut, cre],
Philip Tzvi Reiss [aut]

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2022-08-18 08:50:15 UTC

R topics documented:

`ctmva-package` *Continuous-time multivariate analysis*

Description

Implements continuous-time analogues of several classical techniques of multivariate analysis. The inputs are `"fd"` (functional data) objects from the **fda** package.

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

References

Reiss, Philip T. and Paul, Biplab (2022). Continuous-time multivariate analysis. Preprint.

Ramsay, James O. and Silverman, Bernard W. (2005). Functional Data Analysis, 2nd ed., Springer, New York.

Ramsay, J. O.; Hooker, Giles and Graves, Spencer (2009). Functional Data Analysis with R and Matlab, Springer, New York.

`cca.ct` *Continuous-time canonical correlation analysis*

Description

A continuous-time version of canonical correlation analysis (CCA).

Usage

```
cca.ct(fdobj1, fdobj2)
```

Arguments

`fdobj1, fdobj2` a pair of continuous-time multivariate data sets, of class `"fd"`

Value

A list consisting of

`vex1, vex2` matrices defining the canonical variates. The first columns of each give the coefficients defining the first pair of canonical variates; and so on.

`cor` canonical correlations, i.e., correlations between the pairs of canonical variates

Note

Columns of the output matrix `vex2` are flipped as needed to ensure positive correlations.

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

`cancor`, for classical CCA

Examples

```
# CCA relating Canadian daily temperature and precipitation data
require(fda)
data(CanadianWeather)
daybasis <- create.bspline.basis(c(0,365), nbasis=80)
tempfd <- smooth.basis(day.5, CanadianWeather$dailyAv[,,"Temperature.C"], daybasis)$fd
precfd <- smooth.basis(day.5, CanadianWeather$dailyAv[,,"log10precip"], daybasis)$fd
tpcor <- cca.ct(tempfd, precfd)
oldpar <- par(mfrow=1:2)
barplot(tpcor$vex1[,1], horiz=TRUE, las=1, main="Temperature",
            sub="First canonical coefficients vector")
barplot(tpcor$vex2[,1], horiz=TRUE, las=1, main="Log precipitation",
            sub="First canonical coefficients vector")
par(oldpar)
```

`cor.ct` *Continuous-time correlation or cross-correlation matrix*

Description

Computes the correlation matrix of a continuous-time multivariate data set represented as an `fd` object; or the cross-correlation matrix of two such data sets.

Usage

```
cor.ct(fdobj1, fdobj2 = fdobj1, common_trend = FALSE)
```

Arguments

`fdobj1` continuous-time multivariate data set of class "`fd`"

`fdobj2` an optional second data set

`common_trend` logical: centering wrt mean function if `TRUE`, without centering if `FALSE` (the default)

Value

A matrix of (cross-) correlations

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

`center.fd`, for centering of "`fd`" objects; `inprod.cent`

Examples

```
# Canadian temperature data

require(fda)
require(corrplot)
data(CanadianWeather)
daybasis <- create.fourier.basis(c(0,365), nbasis=55)
tempfd <- smooth.basis(day.5, CanadianWeather$dailyAv[,,"Temperature.C"], daybasis)$fd

## The following yields a matrix of correlations that are all near 1:
rawcor <- cor.ct(tempfd)
corrplot(rawcor, method = 'square', type = 'lower', tl.col="black", tl.cex = 0.6)
## This occurs due to a strong seasonal trend that is common to all stations
## Removing this common trend leads to a more interesting result:
dtcor  <- cor.ct(tempfd, common_trend = TRUE)
ord <- corrMatOrder(dtcor)
dtcord <- dtcor[ord,ord]
corrplot(dtcord, method = 'square', type = 'lower', tl.col="black", tl.cex = 0.6)
```

`cov.ct` *Continuous-time covariance or cross-covariance matrix*

Description

Computes the covariance matrix of a continuous-time multivariate data set represented as an `fd` object; or the cross-covariance matrix of two such data sets.

Usage

```
cov.ct(fdobj1, fdobj2 = fdobj1, common_trend = FALSE)
```

Arguments

`fdobj1`	continuous-time multivariate data set of class "`fd`"
`fdobj2`	an optional second data set
`common_trend`	logical: centering with respect to the mean function if `TRUE`, without centering if `FALSE` (the default)

Value

A matrix of (cross-) covariances

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

`cor.ct`

Examples

```
# see example for cor.ct, which works similarly
```

`inprod.cent` *Centered inner product matrix for a basis or pair of bases*

Description

Most methods of continous-time multivariate analysis require a matrix of inner products of pairs of functions from a basis, such as a B-spline basis, or pairs consisting of one function from each of two bases. This function computes such matrices via 7-point Newton-Cotes integration, which is exact for cubic B-splines.

Usage

```
inprod.cent(basis1, basis2 = basis1, rng = NULL)
```

Arguments

`basis1`	basis object from the `fda` package.
`basis2`	an optional second basis
`rng`	range (of times) spanned by the basis

Value

Matrix of inner products of each pair of centered basis functions.

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

create.bspline.basis from package fda, for the most commonly used basis object type.

Examples

```
require(fda)
basis8 <- create.bspline.basis(nbasis=8)
inprod.cent(basis8)
```

kmeans.ct *Continuous-time k-means clustering*

Description

A continuous-time version of k-means clustering in which each clusters is a time segments or set of time segments.

Usage

```
kmeans.ct(
  fdobj,
  k,
  common_trend = FALSE,
  init.pts = NULL,
  tol = 0.001,
  max.iter = 100
)
```

Arguments

fdobj	continuous-time multivariate data set of class "fd"
k	number of clusters
common_trend	logical: Should the curves be centered with respect to the mean function? Defaults to FALSE.
init.pts	a set of k time points. The observations at these time points serve as initial values for the k means. Randomly generated if not supplied.

`tol` convergence tolerance for the k means

`max.iter` maximum number of iterations

Value

Object of class "`kmeans.ct`", a list consisting of

`fdobj` the supplied `fdobj`

`means` means of the k clusters

`transitions` transition points between segments

`cluster` cluster memberships in the segments defined by the transitions

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

`plot.kmeans.ct`

Examples

```
require(fda)
data(CanadianWeather)
daybasis <- create.bspline.basis(c(0,365), nbasis=55)
tempfd <- smooth.basis(day.5, CanadianWeather$dailyAv[,,"Temperature.C"], daybasis)$fd
kmtemp3 <- kmeans.ct(tempfd, 3)
plot(kmtemp3)
```

`lda.ct` *Continuous-time Fisher's linear discriminant analysis*

Description

A continuous-time version of Fisher's LDA, in which segments of the time interval take the place of groups of observations.

Usage

```
## S3 method for class 'ct'
lda(fdobj, partition)
```

Arguments

`fdobj` continuous-time multivariate data set of class "`fd`"

`partition` a priori break points dividing the time interval into segments

Value

Object of class "lda.ct", a list consisting of

scaling	matrix of coefficients defining the discriminants (as in lda)
values	eigenvalues giving the ratios of between to within sums of squares
partition	the supplied partition
fdobj	linear discriminants represented as an "fd" object
nld	number of linear discriminants

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

plot.lda.ct; lda, for the classical version

Examples

```
## see end of example in ?pca.ct
```

meanbasis *Compute means of basis functions*

Description

Given a basis object as defined in the **fda** package (see basisfd), this function simply computes the vector of means of the basis functions. Used internally.

Usage

```
meanbasis(basis)
```

Arguments

basis	a basis object of class "basisfd"

Value

Vector of means of the basis functions

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

Examples

```
require(fda)
fbasis11 <- create.fourier.basis(nbasis=11)
zapsmall(meanbasis(fbasis11))    # the sine functions have mean 0
```

pca.ct *Continuous-time principal component analysis*

Description

A continuous-time version of principal component analysis.

Usage

```
pca.ct(fdobj, cor = FALSE, common_trend = FALSE)
```

Arguments

`fdobj` continuous-time multivariate data set of class "`fd`"

`cor` logical: use correlation matrix if `TRUE`, covariance if `FALSE` (the default)

`common_trend` logical: Should the curves be centered with respect to the mean function? Defaults to `FALSE`.

Value

Returns a list including:

`var` variances of the principal components.

`loadings` the matrix of loadings (i.e., its columns are the eigenvectors of the continuous-time covariance).

`scorefd` score functions.

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

`cov.ct`; `princomp`, for the classical version

Examples

```
# Data for one session from a classic EEG data set
require(fda)
require(eegkit)
data(eegdata)
data(eegcoord)
longdat <- subset(eegdata, subject=="co2a0000369" & trial==0)
widedat <- reshape(longdat, direction="wide", drop=c("subject","group","condition","trial"),
                 v.names="voltage",idvar="channel")

# Convert time series for 64 channels to a functional data object
bsb <- create.bspline.basis(c(0,255),nbasis=30)
fdo <- Data2fd(argvals=0:255, y=t(as.matrix(widedat[,-1])), basisobj=bsb)
plot(fdo)

# Now do PCA and display first loadings for 3 PC's,
# along with percent variance explained by each
pcc <- pca.ct(fdo)
pve <- 100*pcc$var/sum(pcc$var)
oldpar <- par(mfrow=c(1,3))
cidx <- match(widedat[,1],rownames(eegcoord))
eegspace(eegcoord[cidx,4:5],pcc$loadings[,1], colorlab="PC1 loadings",
         main=paste0(round(pve[1],0), "%"), mar=c(17,3,12,2), cex.main=2)
eegspace(eegcoord[cidx,4:5],pcc$loadings[,2], colorlab="PC2 loadings",
         main=paste0(round(pve[2],0), "%"), mar=c(17,3,12,2), cex.main=2)
eegspace(eegcoord[cidx,4:5],pcc$loadings[,3], colorlab="PC3 loadings",
         main=paste0(round(pve[3],0), "%"), mar=c(17,3,12,2), cex.main=2)

# Linear discriminant analysis: discriminating among the 1st, 2nd and 3rd portions
#  of the time interval
ld <- lda.ct(fdo, c(85,170))
plot(ld)
eegspace(eegcoord[cidx,4:5],ld$scaling[,1], colorlab="LD1 coefficients",
         mar=c(17,3,12,2), cex.main=2)
eegspace(eegcoord[cidx,4:5],ld$scaling[,2], colorlab="LD2 coefficients",
         mar=c(17,3,12,2), cex.main=2)
par(oldpar)
```

`plot.kmeans.ct` *Plot a kmeans.ct object*

Description

Plots a continuous-time k-means clustering object generated by a call to `kmeans.ct`.

Usage

```
## S3 method for class 'kmeans.ct'
plot(
  x,
  type = "functions",
  mark.transitions = TRUE,
  col = NULL,
  lty = NULL,
  xlab = "Time",
  ylab = NULL,
  legend = TRUE,
  ncol.legend = 1,
  ...
)
```

Arguments

`x` clustering object produced by `kmeans.ct`

`type` either `"functions"` (the default), to display each variable as a smooth function of time, or `"distance"`, to plot distances from the k cluster means versus time.

`mark.transitions` logical: Should transitions between clusters be marked with vertical lines? Defaults to `TRUE`.

`col` plot colours

`lty` line type

`xlab, ylab` x- and y-axis labels

`legend` logical: should a legend be included? Default is `TRUE`.

`ncol.legend` number of columns for legend

`...` other arguments passed to `matplot`

Value

None; a plot is generated.

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

`kmeans.ct`, which includes an example

plot.lda.ct *Plot an lda.ct object*

Description

Plots the Fisher's linear discriminant functions generated by a call to lda.ct.

Usage

```
## S3 method for class 'lda.ct'
plot(x, ylab = "Discriminants", xlab = "Time", ...)
```

Arguments

x	linear discriminant analysis object produced by lda.ct
ylab, xlab	y- and x-axis labels
...	other arguments passed to matplot

Value

None; a plot is generated.

Author(s)

Biplab Paul <paul.biplab497@gmail.com> and Philip Tzvi Reiss <reiss@stat.haifa.ac.il>

See Also

lda.ct

Examples

```
## see the example at the end of ?pca.ct
```

Index