Package ‘fdadensity’

October 13, 2022

URL https://github.com/functionaldata/tDENS

BugReports https://github.com/functionaldata/tDENS/issues

Type Package

Title Functional Data Analysis for Density Functions by Transformation to a Hilbert Space

Version 0.1.2

Date 2019-12-03

Author A. Petersen, P. Z. Hadjipantelis and H.G. Mueller

Maintainer Alexander Petersen <petersen@pstat.ucsb.edu>

Description An implementation of the methodology described in Petersen and Mueller (2016) <doi:10.1214/15-AOS1363> for the functional data analysis of samples of density functions. Densities are first transformed to their corresponding log quantile densities, followed by ordinary Functional Principal Components Analysis (FPCA). Transformation modes of variation yield improved interpretation of the variability in the data as compared to FPCA on the densities themselves. The standard fraction of variance explained (FVE) criterion commonly used for functional data is adapted to the transformation setting, also allowing for an alternative quantification of variability for density data through the Wasserstein metric of optimal transport.

Depends R (>= 3.3.0)

License BSD_3_clause + file LICENSE

LazyData false

Imports Rcpp (>= 0.11.5), fdapace (>= 0.3.0)

LinkingTo Rcpp

NeedsCompilation yes

Suggests testthat

RoxygenNote 7.0.2

Encoding UTF-8

Repository CRAN

Date/Publication 2019-12-05 06:20:02 UTC
R topics documented:

BacteriaPI ................................................................. 2
CreateModeOfVarPlotLQ2D ............................................ 3
dens2lqd ................................................................. 4
dens2qd ................................................................. 5
dens2quantile .......................................................... 6
DeregulariseByAlpha .................................................... 7
FPCAdens ................................................................. 8
GetFVE ................................................................. 9
getWFmean ............................................................. 10
lqd2dens ................................................................. 12
lqd2quantile ............................................................ 13
MakeDENsample ........................................................ 14
MakeLQDsample ........................................................ 15
normaliseDensities ...................................................... 16
qd2dens ................................................................. 17
RegulariseByAlpha ...................................................... 18
Top50BabyNames ......................................................... 19

Description

The approximate kernel density estimates of the 813 bacterial organisms’ isoelectric point (pI) protein distributions.

Format

A matrix with 813 rows and 768 columns:

rowname  General organism identifier
colspace  pH in [0,14]

References

The authors would like to thank Dr. Chris Knight for providing the original data
CreateModeOfVarPlotLQ2D

Transformation Mode of Variation Plot

Description

Create the k-th transformation mode of variation plot.

Usage

CreateModeOfVarPlotLQ2D(
  fpcaObj, 
  domain = "D", 
  k = 1, 
  dSup = NULL, 
  Qvec = -2:2, 
  alpha = 0, 
  useAlpha = FALSE, 
  ... 
)

Arguments

- fpcaObj: An FPCA class object returned by FPCA() on the log quantile density functions.
- domain: The mode should be plotted in LQD ('Q') or density space ('D', the default).
- k: The k-th mode of variation to plot (default k = 1).
- dSup: The common support of the original densities. Only relevant for domain = 'D'.
- Qvec: Vector of values Q to be plotted. If 0 is not included, it will be added (default is -2:2). Only relevant for domain = 'D'.
- alpha: (De)regularisation parameter (default is 0). See details.
- useAlpha: logical - should deregularisation be performed? Default:FALSE
- ... Additional arguments for the 'plot' function.

Details

If domain = 'D' (the default), the a transformation mode of variation is plotted. The red-line is \( \psi^{-1}(\nu) \), where \( \nu \) is the mean in LQD space and \( \psi \) is the LQD transformation. Other lines correspond to perturbations by adding multiples of the LQD eigenfunctions \( \rho_k \) (with eigenvalues \( \tau_k \)):

\[ \psi^{-1}(\nu + Q \sqrt{\tau_k} \rho_k) \]

for the values \( Q \) in Qvec. If alpha is positive, will attempt to deregularise (see DeregulariseByAlpha). This will throw an error if alpha is too large.

If domain = 'Q', ordinary modes of variation are plotted in LQD space (see documentation for CreateModeOfVarPlot in fdapace).
dens2lqd

Function for converting densities to log quantile density functions

Description

Function for converting densities to log quantile density functions

Usage

dens2lqd(dens, dSup, N = length(dSup), lqdSup = NULL)

Arguments

dens
density values on dSup - must be strictly positive and integrate to 1
dSup
support (grid) for Density domain
N
desired number of points on a [0,1] grid for lqd function; default length(dSup)
lqdSup
support for lqd domain - must begin at 0 and end at 1; default [0,1] with N-equidistant support points
dens2qd

Value

lqd log quantile density on lqdSup

References


See Also

normaliseDensities

Examples

x <- seq(0,2,length.out =512)
y <- rep(0.5,length.out =512)
y.lqd <- dens2lqd(dens=y, dSup = x) # should equate # log(2)

dens2qd Function for converting Densities to Quantile Densities

Description

Function for converting Densities to Quantile Densities

Usage

dens2qd(
dens,
dSup = seq(0, 1, length.out = length(dens)),
qdSup = seq(0, 1, length.out = length(dens)),
useSplines = TRUE
)

Arguments

dens density on dSup
dSup support for Density domain - max and min values mark the boundary of the support.
qdSup support for quantile density domain - must begin at 0 and end at 1
useSplines fit spline to the qd when doing the numerical integration (default: TRUE)

Value

qd quantile density values on qdSup
dens2quantile

Function for converting Densities to Quantile Functions

Usage

dens2quantile(
dens,
dSup = seq(0, 1, length.out = length(dens)),
qSup = seq(0, 1, length.out = length(dens)),
useSplines = TRUE
)

Arguments

dens density on dSup
dSup support for Density domain - max and min values mark the boundary of the support.
qSup support for quantile domain - must begin at 0 and end at 1
useSplines fit spline to the qd when doing the numerical integration (default: TRUE)

Value

Q quantile function on qSup

References

DeregulariseByAlpha

See Also

normaliseDensities

Examples

```r
x <- seq(0, 2, length.out = 512)
y <- rep(0.5, length.out = 512)
y.quantile <- dens2quantile(dens = y, dSup = x) # should equate # 2*seq(0, 1, length.out = 512)
```

---

**DeregulariseByAlpha**

*Function to deregularise densities to have (smaller) minimum value*

**Description**

If possible, deregularises the input density \( y \) to have minimum density value is \( \alpha \). See details.

**Usage**

```r
DeregulariseByAlpha(x, y, alpha = 0)
```

**Arguments**

- **x**: support of the density
- **y**: values of the density
- **alpha**: scalar to deregularise with (default = 0) - this will be the minimum value of the deregularised density, unless \( \min(y) < \alpha \), in which case no deregularisation will be performed

**Details**

If \( \min(y) \leq \alpha \), or \( y \) is the uniform distribution, no deregularisation is performed and \( y \) is returned. If \( \min(y) \times \text{diff(range}(x)) > 1 \), the deregularisation is not possible and an error is thrown. Otherwise, the deregularised density in an inverse manner to RegulariseByAlpha.

**Value**

\( \text{dens} \) density values on \( x \)

**See Also**

RegulariseByAlpha
Examples

```r
x = seq(0,1,length.out=122)
y = seq(0.1,1.9,length.out=122)
z = DeregulariseByAlpha(x=x, y=y, alpha = 0)
```

---

FPCAdens  

**FPCA for densities by log quantile density transformation**

---

Description

Perform FPCA on LQD-transformed densities

Usage

```r
FPCAdens(
  dmatrix,
  dSup,
  lqdSup = seq(0, 1, length.out = length(dSup)),
  useAlpha = FALSE,
  alpha = 0.01,
  optns = list(dataType = "Dense", error = FALSE)
)
```

Arguments

- **dmatrix**: Matrix holding the density values on dSup - all rows must be strictly positive and integrate to 1
- **dSup**: Support (grid) for Density domain
- **lqdSup**: Support grid for lqd domain (default = seq(0, 1, length.out = length(dSup)))
- **useAlpha**: should regularisation be performed (default=FALSE)
- **alpha**: Scalar to regularise the supports with (default=0.01)
- **optns**: A list of options for FPCA. See documentation for FPCA.

Details

Densities are transformed to log-quantile densities, followed by standard FPCA. If useAlpha = TRUE, densities are regularized before transformation

References

*Functional Data Analysis for Density Functions by Transformation to a Hilbert space, Alexander Petersen and Hans-Georg Mueller, 2016*
GetFVE

### Computed Metric-based Fraction of Variance Explained

**Description**

When FPCA is performed on the log quantile density functions, the fraction of variance explained by the first K components is computed based on the density reconstruction and chosen metric.

**Usage**

\[
\text{GetFVE}(\text{fpcaObj}, \text{dmatrix}, \text{dSup}, \text{metric} = \text{"L2"}, \text{useAlpha} = \text{FALSE}, \alpha = 0.01)
\]

**Arguments**

- `fpcaObj`: PACE output (FPCA on LQDs)
- `dmatrix`: matrix of original densities measures on grid `dSup`, rows correspond to individual densities
- `dSup`: support for Density domain - max and min mark the boundary of the support.
- `metric`: metric for measuring variance - 'L2' for Euclidean or 'W' for Wasserstein
- `useAlpha`: should regularisation be performed to densities in `dmatrix`? This should be set to TRUE if densities were regularised prior to FPCA (default = FALSE)
alpha scalar to regularise before computing FVE. If useAlpha = TRUE, this should match the value used to regularise prior to FPCA (default = 0.01)

Details
The fraction of variance explained (FVE) by the first K principal components corresponding to the LQD functions is computed by taking the K-dimensional LQD representations, transforming back to densities, and comparing the reconstruction to the original densities using the chosen metric. If densities were regularised prior to transformation and FPCA, the same regularisation parameters should be used here.

Value
FVEvector

References

See Also
RegulariseByAlpha, lqd2quantile

Examples

data(Top50BabyNames)

# Perform Transformation FPCA for male baby name densities
dSup = Top50BabyNames$x
X = FPCAdens(dmatrix = t(Top50BabyNames$dens$male), dSup = dSup, useAlpha = TRUE,
    optns = list(dataType = 'Dense', error = FALSE, methodSelectK = 8))

# Compute FVE - must compare to regularized densities
fveL2 = GetFVE(fpcaObj = X, dmatrix = t(Top50BabyNames$dens$male), dSup = dSup, useAlpha = TRUE)
fveW = GetFVE(fpcaObj = X, dmatrix = t(Top50BabyNames$dens$male), dSup = dSup,
    metric = 'W', useAlpha = TRUE)
getWFmean

Usage

getWFmean(
  dmatrix,
  dSup,
  N = length(dSup),
  qdSup = seq(0, 1, length.out = N),
  useAlpha = FALSE,
  alpha = 0.01
)

Arguments

dmatrix matrix of density values on dSup - must be strictly positive and each row must integrate to 1
dSup support (grid) for Density domain
N desired number of points on a [0,1] grid for quantile density functions; default length(dSup)
qdSup support for LQ domain - must begin at 0 and end at 1; default [0,1] with N-equidistant support points
useAlpha should regularisation be performed (default=FALSE)
alpha Scalar to regularise the supports with (default=0.01)

Value

wfmean the Wasserstein-Frechet mean density

References


Examples

x <- seq(0,1,length.out = 101)
# linear densities on (0, 1)
y <- t(sapply(seq(0.5, 1.5, length.out = 10), function(b) b + 2*(1 - b)*x))
wfmean = getWFmean(y, x)

# Plot WF mean with Euclidean Mean
matplot(x, t(y), ylab = 'Density', type = 'l', lty = 1, col = 'black')
lines(x, wfmean, lwd = 2, col = 'red')
lines(x, colMeans(y), lwd = 2, col = 'blue')
legend('topright', col = c('black', 'red', 'blue'), lwd = c(1, 2, 2),
       legend = c('Densities', 'WF Mean', 'Euclidean Mean'))
lqd2dens

Function for converting log quantile densities to densities

Description

Function for converting log quantile densities to densities

Usage

lqd2dens(
  lqd,
  lqdSup = seq(0, 1, length.out = length(lqd)),
  dSup,
  useSplines = TRUE
)

Arguments

lqd log quantile density on lqdSup
lqdSup support for lqd domain - must begin at 0 and end at 1
dSup support for Density domain - max and min values mark the boundary of the support.
useSplines fit spline to the lqd when doing the numerical integration (default: TRUE)

Value

dens density values on dSup

References


Examples

x <- seq(0, 2, length.out = 512)
y.lqd <- rep(log(2), times = 512)
y <- lqd2dens(dSup = x, lqd = y.lqd) # should equate # 1/2
lqd2quantile

Function for converting log quantile densities to quantile functions

Description

Function for converting log quantile densities to quantile functions

Usage

lqd2quantile(
  lqd,
  lqdSup = seq(0, 1, length.out = length(lqd)),
  lb = 0,
  useSplines = TRUE
)

Arguments

  lqd       log quantile density on lqdSup
  lqdSup    support for lqd domain - must begin at 0 and end at 1
  lb        lower bound of support for Density domain - default is 0.
  useSplines fit spline to the lqd when doing the numerical integration (default: TRUE)

Value

quantile values on lqdSup

References


Examples

  x <- seq(1,3,length.out =512)
  y.lqd <- rep(log(2), times = 512)
  y <- lqd2quantile(lqd = y.lqd, lb = 1) # should equate # seq(1, 3, length.out = 512)
MakeDENsample

Convenience function for converting log quantile densities to densities

Description

See 'lqd2dens' and 'DeregulariseByAlpha' for more details. This function transforms the log quantile densities in 'qmatrix' to density functions, optionally followed by deregularisation.

Usage

MakeDENsample(
  qmatrix,
  lqdSup = seq(0, 1, length.out = ncol(qmatrix)),
  dSup = seq(0, 1, length.out = ncol(qmatrix)),
  useAlpha = FALSE,
  alpha = 0
)

Arguments

qmatrix  Matrix holding the log quantile density values on [0,1]
lqdSup   Support grid for input log quantile densities (default = seq(0, 1, length.out = ncol(qmatrix)))
dSup     Support grid for output densities (default = seq(0, 1, length.out = ncol(qmatrix)))
useAlpha Logical indicator to deregularise the densities (default = FALSE)
alpha    Scalar to deregularise the density - where possible, this will be the minimum value for the deregularised densities (default=0)

Value

list with the 'DEN' transformed data, and 'dSup' that matches the input argument.

References

*Functional Data Analysis for Density Functions by Transformation to a Hilbert space, Alexander Petersen and Hans-Georg Mueller, 2016*

See Also

*DeregulariseByAlpha, lqd2dens*
Examples

```r
x <- seq(0, 1, length.out = 101)
# linear densities on (0, 1)
y <- t(sapply(seq(0.5, 1.5, length.out = 10), function(b) b + 2*(1 - b)*x))

# Get LQDs
y.lqd = MakeLQDsample(dmatrix = y, dSup = x)
matplot(y.lqd$lqdSup, t(y.lqd$LQD), ylab = "LQD", type = "l", lty = 1, col = "black")

# Get Densities Back
y.dens = MakeDENsample(y.lqd$LQD, lqdSup = x, dSup = x) # should equate to y above
matplot(y.dens$dSup, t(y.dens$DEN), ylab = "Density", type = "l", lty = 1, col = "blue")
matplot(x, t(y), ylab = "Original Density", type = "l", lty = 1, col = "red")
```

MakeLQDsample

Convenience function for converting densities to log-quantile densities

Description

See 'dens2lqd' and 'RegulariseByAlpha' for more details. This function first (transforms the densities in 'dmatrix' to log quantile density functions, optionally followed by regularisation.

Usage

```r
MakeLQDsample(
  dmatrix,
  dSup,
  lqdSup = seq(0, 1, length.out = length(dSup)),
  useAlpha = FALSE,
  alpha = 0.01
)
```

Arguments

- **dmatrix**: Matrix holding the density values on dSup - all rows must be strictly positive and integrate to 1
- **dSup**: Support (grid) for Density domain
- **lqdSup**: Support grid for lqd domain (default = seq(0, 1, length.out = length(dSup)))
- **useAlpha**: should regularisation be performed (default=FALSE)
- **alpha**: Scalar to regularise the supports with (default=0.01)
normaliseDensities

Value

list with 'LQD', a matrix of log quantile density functions, and 'lqdSup' that matches the input argument

References

*Functional Data Analysis for Density Functions by Transformation to a Hilbert space, Alexander Petersen and Hans-Georg Mueller, 2016*

See Also

RegulariseByAlpha, dens2lqd

Examples

```r
x <- seq(0,1,length.out = 101)
# some log quantile densities on (0, 1)
y <- t(sapply(seq(0.5, 1.5, length.out = 10), function(b) -log(b^2 + 4*(1-b)*x)/2))
# Get densities
y.dens = MakeDENsample(qmatrix = y, lqdSup = x, dSup = x)$DEN
matplot(x, t(y.dens), ylab = 'Density', type = 'l', lty = 1, col = 'black')
# Get LQDs Back
y.lqd = MakeLQDsample(y.dens, lqdSup = x, dSup = x)
# These should match
matplot(y.lqd$lqdSup, t(y.lqd$LQD), ylab = 'LQD', type = 'l', lty = 1, col = 'blue')
matplot(x, t(y), ylab = 'LQD', type = 'l', lty = 1, col = 'red')
```

Description

Preprocessing function to ensure densities integrate to 1

Usage

```r
normaliseDensities(dmatrix, dSup = 1:ncol(dmatrix))
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dmatrix</td>
<td>Matrix with rows representing distinct densities on dSup - all entries must be nonnegative</td>
</tr>
<tr>
<td>dSup</td>
<td>Support (grid) for Density domain</td>
</tr>
</tbody>
</table>
qd2dens

Details
Uses trapezoidal integration to normalise the densities to have integral 1

Value
matrix 'dmatrix' consisting of rows of input of the same name that have been normalised to have integral 1

References

Examples
## Normalise collection of truncated normal densities

mu <- seq(-2, 2, by = 0.5)
dSup = seq(-3, 3, length.out = 101)
y <- t(sapply(mu, function(m) dnorm(x = dSup, mean = m)))

# Should return warnings about densities not integrating to 1
lqd = MakeLQDsample(dmatrix = y, dSup = dSup)

# Normalise and rerun without warning
dens <- normaliseDensities(dmatrix = y, dSup = dSup)
lqd = MakeLQDsample(dmatrix = dens, dSup = dSup)

qd2dens

Function for converting Quantile Densities to Densities

Description
Function for converting Quantile Densities to Densities

Usage
qd2dens(
  qd,
  qdSup = seq(0, 1, length.out = length(qd)),
  dSup,
  useSplines = TRUE
)
RegulariseByAlpha

Arguments

qd        quantile density on qdSup
qdSup     support for quantile domain - must begin at 0 and end at 1 (default = seq(0, 1, length.out = length(qd)))
dSup      support for Density domain - max and min values mark the boundary of the support.
useSplines fit spline to the qd when doing the numerical integration (default: TRUE)

Value
dens density values on dSup

References


Examples

x <- seq(0,1,length.out =512)
y <- rep(2,length.out =512)
y.dens <- qd2dens(qd=y, qdSup = x, dSup = seq(0, 2, length.out = 512)) # should equate # 1/2

RegulariseByAlpha Function to regularise densities to have (larger) minimum value

Description

If possible, regularises the input density y to have minimum density value is alpha. See details.

Usage

RegulariseByAlpha(x, y, alpha = 0.01)

Arguments

x       support of the density
y       values of the density
alpha   scalar to regularise with (default = 0.01) - this will be the minimum value of the regularised density, unless min(y) > alpha, in which case no regularisation will be performed
Details

If \( \min(y) \geq \alpha \) or \( y \) is the uniform distribution, no regularisation is performed and \( y \) is returned. If \( \alpha \cdot \text{diff}(\text{range}(x)) > 1 \), the regularisation is not possible and an error is thrown. Otherwise, the regularised density is computed by adding an appropriate constant \( \gamma \), followed by renormalisation to have integral 1.

Value

dens density values on \( x \)

See Also

DeregulariseByAlpha, normaliseDensities

Examples

\[
\begin{align*}
x &= \text{seq}(0,1,\text{length.out}=122) \\
y &= \text{seq}(0,2,\text{length.out}=122) \\
z &= \text{RegulariseByAlpha}(x=x, y=y, \alpha = 0.1)
\end{align*}
\]

Top50BabyNames

Baby name popularity densities for 50 male and 50 female names in the USA

Description

Baby name popularity densities, obtained by smoothing year-to-year popularity indices from 1950 to 2016, after normalization to have integral equal to 1. The top 50 names, in absolute popularity, are included for each gender.

Format

A list with two variables

\( x \) grid of years between 1950 and 2016, of length 67.

\( \text{dens} \) list of length two, corresponding to male (\( \text{dens$male} \)) and female (\( \text{dens$female} \)) names. Each is a 67-by-50 matrix of density estimates, where each column corresponds to a unique baby name given by the corresponding column name.

References

Data from the R package babynames, originally from the US Social Security Administration
Index

BacteriaPI, 2
CreateModeOfVarPlotLQ2D, 3
dens2lqd, 4, 16
dens2qd, 5
dens2quantile, 6
DeregulariseByAlpha, 4, 7, 14, 19
FPCA, 9
FPCAdens, 8
GetFVE, 9
getWFmean, 10
lqd2dens, 9, 12, 14
lqd2quantile, 10, 13
MakeDENsample, 14
MakeLQDsamp;le, 9, 15
normaliseDensities, 5–7, 16, 19
qd2dens, 17
RegulariseByAlpha, 7, 9, 10, 16, 18
Top50BabyNames, 19