Package ‘DrBats’

October 12, 2022

Type Package
Title Data Representation: Bayesian Approach That's Sparse
Version 0.1.6
Maintainer Benedicte Fontez <benedicte.fontez@supagro.fr>
Description Feed longitudinal data into a Bayesian Latent Factor Model to obtain a low-rank representation. Parameters are estimated using a Hamiltonian Monte Carlo algorithm with STAN. See G. Weinrott, B. Fontez, N. Hilgert and S. Holmes, "Bayesian Latent Factor Model for Functional Data Analysis", Actes des JdS 2016.
Depends R (>= 3.1.0), rstan
Imports ade4, coda, MASS, Matrix, sde
License GPL-3
Encoding UTF-8
LazyData TRUE
Suggests fda, ggplot2, knitr, parallel, rmarkdown, testthat
VignetteBuilder knitr
NeedsCompilation no
Author Gabrielle Weinrott [aut], Brigitte Charnomordic [ctr], Benedicte Fontez [cre, aut], Nadine Hilgert [ctr], Susan Holmes [ctr], Isabelle Sanchez [ctr]
RoxygenNote 7.1.2
Repository CRAN
Date/Publication 2022-02-13 19:00:12 UTC

R topics documented:
coda.obj ................................................................. 2
coda.obj

Convert a STAN obj to MCMC list

description

Convert a STAN obj to MCMC list

Usage

coda.obj(stanfit)

Arguments

stanfit a STAN object

Value

codafit an mcmc.list

Author(s)

Gabrielle Weinrott

Examples

data(stanfit) # output of modelFit or main.modelFit
coda.fit <- coda.obj(stanfit)
head(coda.fit)
Perform Coinertia Analysis on the PCA of the Weighted PCA and Deville’s PCA

Usage

coinertia.drbats(
  X.histo = NULL,
  Qp = NULL,
  X = NULL,
  t = NULL,
  t.range = c(0, 1000),
  breaks
)

Arguments

- **X.histo**: the data matrix projected onto the histogram basis
- **Qp**: a matrix of weights, if Qp = NULL the function specifies a diagonal weight matrix
- **X**: a data matrix, if X.histo is NULL and needs to be built
- **t**: a matrix of observation times, if X.histo is NULL and needs to be built
- **t.range**: the range of observation times in vector form, if X.histo is NULL and needs to be built (default: t.range = c(0, 1000))
- **breaks**: integer number of histogram windows

Value

- **co_weight**: the co-inertia object

Author(s)

Gabrielle Weinrott

Examples

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
res.coinertia <- coinertia.drbats(X = res$X, t = res$t.simul, t.range = c(5, 100), breaks = 8)
res.coinertia
```
drbats.simul  Main simulation function

Description

Main simulation function

Usage

drbats.simul(
    N = 10,
    P = 150,
    t.range = c(0, 1000),
    b.range = c(0.2, 0.4),
    c.range = c(0.6, 0.8),
    b.sd = 2,
    c.sd = 2,
    a.range = c(-0.4, 0.4),
    y.range = c(0, 10),
    amp = 10,
    per = 12,
    data.type = "sparse",
    breaks = 15,
    sigma2 = 0.2,
    seed = NULL
)

Arguments

N  integer number of functions to simulate (default = 10)
P  a number of observation times (default = 150)
t.range  a range of times in which to place the P observations (default = c(1, 1000))
b.range  a vector giving the range of values for the mean of the first mode (default b.range = c(0.2, 0.4))
c.range  a vector giving the range of values for the mean of the second mode (default c.range = c(0.6, 0.8))
b.sd  the standard deviation for the first mode (default b.sd = 2)
c.sd  the standard deviation for the second mode (default c.sd = 2)
a.range  a vector giving the range of values for the slope (default a.range = c(-0.4, 0.4))
y.range  a vector giving the range of values for the intercept (default y.range = c(0, 10))
amp  the amplitude of the cosine function (default = 10)
per  the periodicity of the cosine function (default = 12)
data.type  string indicating type of functions (options :sparse, sparse.tend, sparse.tend.cos)
histoProj

Project a set of curves onto a histogram basis

**Description**

Project a set of curves onto a histogram basis

**Usage**

```r
histoProj(X, t, t.range, breaks)
```

**Arguments**

- `X` a matrix
- `t` a matrix of observation times
- `t.range` a range of times in which to place the P projections (default = c(0, 1000))
- `breaks` the number of intervals in the histogram basis

**Value**

- `Y.simul` a list containing a matrix Y, a matrix beta, and a matrix epsilon
- `t.simul` a matrix of simulated observation times
- `X` the underlying signal to build the data, see DataSimulationandProjection vignette
- `proj.pca` the outputs of the function pca.proj.Xt
- `wlu` the outputs of the function W.QR

**Examples**

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
X <- res$X
t <- res$t.simul
# To plot the observations, ie the rows
matplot(t(t), t(X), type = "l", xlab = "Time", ylab = "X")
```
Value

- X.proj the matrix X after projection
- X.count a matrix containing the number of observations used to build the projection onto the histogram basis
- windows a vector containing the first time of each window of the histogram intervals
- X.max the matrix of minimum values in each window
- X.min the matrix of maximum values in each window

Author(s)

Gabrielle Weinrott

Examples

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
res.proj <- histoProj(res$X, res$t.simul, t.range = c(5, 100), breaks = 8)
res.proj
```

modelFit

*Fit a Bayesian Latent Factor to a data set using STAN*

Description

Fit a Bayesian Latent Factor to a data set using STAN

Usage

```r
modelFit(
  model = "PLT",
  var.prior = "IG",
  prog = "stan",
  parallel = TRUE,
  Xhisto = NULL,
  nchains = 4,
  nthin = 10,
  niter = 10000,
  R = NULL
)
```

Arguments

- **model** a string indicating the type of model ("PLT", or sparse", default = "PLT")
- **var.prior** the family of priors to use for the variance parameters ("IG" for inverse gamma, or "cauchy")
pca.Deville

prog a string indicating the MCMC program to use (default = "stan")
parallel true or false, whether or not to parallelize (done using the package "parallel"
Xhisto matrix of simulated data (projected onto the histogram basis)
nchains number of chains (default = 2)
nthin the number of thinned iterations (default = 1)
niter number of iterations (default = 1e4)
R rotation matrix of the same dimension as the number of desired latent factors

Value
stanfit, a STAN object

Author(s)
Gabrielle Weinrott

References
stan.org/

pca.Deville

Perform a PCA using Deville’s method

Description
Perform a PCA using Deville’s method

Usage
pca.Deville(X, t, t.range, breaks)

Arguments
X a data matrix
t a matrix of observation times corresponding to X
t.range the range of observation times in vector form (ex. t.range = c(0, 1000))
breaks integer number of histogram windows

Value
X.histo the matrix projected onto the histogram basis
U.histo a matrix of eigenvectors in the histogram basis
Cp a matrix of principal components
lambda a vector of eigenvalues
perc.lambda a vector of the percentage of total inertia explained by each principal component
Author(s)

Gabrielle Weinrott

References


Examples

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
res.pca <- pca.Deville(res$X, res$t.simul, t.range = c(5, 100), breaks = 8)
res.pca
```

pca.proj.Xt

PCA data projected onto a histogram basis

Description

PCA data projected onto a histogram basis

Usage

```r
pca.proj.Xt(X, t, t.range = c(0, 1000), breaks = 15)
```

Arguments

- `X`: the data matrix
- `t`: the matrix of observation times
- `t.range`: a vector specifying the observation time range (default: `c(0, 1000)`)
- `breaks`: the number of breaks in the histogram basis (default: `breaks = 15`)

Value

- `Xt.proj`: a matrix of projected observations
- `U`: a matrix of eigenvectors
- `lambda`: a vector of eigenvalues
- `lambda.perc`: the percentage of inertia captured by each axis

Author(s)

Gabrielle Weinrott
Examples

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
pca.proj.Xt(res$X, res$t.simul, t.range = c(0, 100), breaks = 8)
```

---

**postdens**

*Calculate the unnormalized posterior density of the model*

**Description**

Calculate the unnormalized posterior density of the model

**Usage**

```r
postdens(mcmc.output, Y, D, chain = 1)
```

**Arguments**

- `mcmc.output`: an mcmc list as produced by clean.mcmc
- `Y`: the data matrix
- `D`: the number of latent factors
- `chain`: the chain to plot (default = 1)

**Value**

`post` a vector containing the posterior density at each iteration

**Author(s)**

Gabrielle Weinrott

**Examples**

```r
data("toydata")
data("stanfit")
dens <- postdens(coda.obj(stanfit), Y = toydata$Y.simul$Y, D = 2, chain = 1)
hist(dens)
```
stanfit

A stanfit object fitted to the toydata

Description

A stanfit object fitted to the toydata

Usage

stanfit

Format

A large stanfit object

toydata

A toy longitudinal data set

Description

A toy longitudinal data set

Usage

toydata

Format

A list with 5 elements:

Y.simul  a list of simulated data with 3 elements
t.simul  a matrix with 5 rows and 150 columns giving the observation times of the original dataX  the original data matrix with 5 rows and 150 columnsproj.pca  a list with 4 elements: results of the function histoProj(X, t, t.range = c(0, 1000), breaks = 8)wlu  a list with 4 elements: results of the function W.QR(U, lambda) where U and lambda are the results of the PCA of X
Description

Format scores output for visualization

Usage

visbeta(mcmc.output, Y, D, chain = 1, axes = c(1, 2), quant = NULL)

Arguments

mcmc.output  an mcmc list as produced by clean.mcmc
Y           the matrix of data
D           the number of latent factors
chain       the chain to use (default = 1)
axes        the axes to use (default = c(1, 2))
quant       a vector of quantiles to retain (default = NULL)

Value

mean.df are the MCMC estimates for the parameters
points.df contains all of the estimates of the chain
contour.df contains the exterior points of the convex hull of the cloud of estimates

Author(s)

Gabrielle Weinrott

Examples

data("toydata")
data("stanfit")
codafit <- coda.obj(stanfit) ## convert to mcmc.list
beta.res <- visbeta(codafit, Y = toydata$Y.simul$Y, D = toydata$wlud$D, chain = 1,
axes = c(1, 2), quant = c(0.05, 0.95))

ggplot2::ggplot() +
  ggplot2::geom_path(data = beta.res$contour.df, ggplot2::aes(x = x, y = y, colour = ind)) +
  ggplot2::geom_point(data = beta.res$mean.df, ggplot2::aes(x = x, y = y, colour = ind))
Description
Plot the estimates for the latent factors

Usage
visW(mcmc.output, Y, D, chain = 1, factors = c(1, 2))

Arguments
mcmc.output  an mcmc list as produced by clean.mcmc
Y     the matrix of data
D     the number of latent factors
chain  the chain to plot (default = 1)
factors a vector indicating the factors to plot (default = c(1, 2))

Value
res.W a data frame containing the estimates for the factors, and their lower and upper bounds
Inertia the percentage of total inertia captured by each of the factors

Author(s)
Gabrielle Weinrott

Examples
data("toydata")
data("stanfit")
codafit <- coda.obj(stanfit) ## convert to mcmc.list
W.res <- visW(codafit, Y = toydata$Y.simul$Y, D = toydata$wlu$D,
chain = 1, factors = c(1, 2))

## plot the results

data <- data.frame(time = rep(1:9, 2), W.res$res.W)
ggplot2::ggplot() +
ggplot2::geom_step(data = data, ggplot2::aes(x = time, y = Estimation, colour = Factor)) +
ggplot2::geom_step(data = data, ggplot2::aes(x = time, y = Lower.est, colour = Factor),
linetype = "longdash") +
ggplot2::geom_step(data = data, ggplot2::aes(x = time, y = Upper.est, colour = Factor),
linetype = "longdash")
**W.QR**

*Build and decompose a low-rank matrix W*

---

**Description**

Build and decompose a low-rank matrix from a matrix of eigenvectors and eigenvalues from principal component analysis

**Usage**

\[
W.QR(U, lambda)
\]

**Arguments**

- **U** a matrix of eigenvectors
- **lambda** a vector of corresponding eigenvalues

**Value**

- **W** a low-rank matrix
- **D** the number of latent factors
- **Q** the orthogonal matrix of the \(W = QR\) matrix decomposition
- **R** the upper triangular matrix of the \(W = QR\) matrix decomposition

**Author(s)**

Gabrielle Weinrott

**Examples**

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
res.pca <- pca.Deville(res$X, res$t.simul, t.range = c(5, 100), breaks = 8)
Wres.pca <- W.QR(res.pca$U, res.pca$lambda)
W.res.pca
```
weighted.Deville

Perform a weighted PCA using Deville’s method on a data matrix \( X \) that we project onto a histogram basis and weighted

Description

Perform a weighted PCA using Deville’s method on a data matrix \( X \) that we project onto a histogram basis and weighted

Usage

weighted.Deville(X, t, t.range, breaks, Qp = NULL)

Arguments

- \( X \) a data matrix
- \( t \) a matrix of observation times corresponding to \( X \)
- \( t\.range \) the range of observation times in vector form (ex. \( t\.range = c(a, b) \))
- \( \text{breaks} \) integer number of histogram windows
- \( Qp \) a matrix of weights, if \( Qp = \text{NULL} \) the function specifies a diagonal weight matrix

Value

- \( X\.h histo \) the matrix projected onto the histogram basis
- \( U\.h histo \) a matrix of eigenvectors in the histogram basis
- \( Cp \) a matrix of principal components
- \( lambda \) a vector of eigenvalues
- \( \text{perc.lambda} \) a vector of the percentage of total inertia explained by each principal component

Author(s)

Gabrielle Weinrott

Examples

```r
res <- drbats.simul(N = 5, P = 100, t.range = c(5, 100), breaks = 8)
res.weighted <- weighted.Deville(res$X, res$t.simul, t.range = c(5, 100), breaks = 8, Qp = NULL)
res.weighted
```
Index

* datasets
  stanfit, 10
  toydata, 10

coda.obj, 2
coinertia.drbats, 3
drbats.simul, 4
histoProj, 5
modelFit, 6
pca.Deville, 7
pca.proj.Xt, 8
postdens, 9
stanfit, 10
toydata, 10
visbeta, 11
visW, 12
W.QR, 13
weighted.Deville, 14