Package ‘robsfplsr’

February 6, 2024

Type Package

Title Robust Sparse Functional Partial Least Squares Regression

Version 1.0

Depends R (>= 3.5.0), fda, MASS

Imports expm, fda.usc, goffda, pcaPP, cvTools

LazyLoad yes

ByteCompile TRUE

Encoding UTF-8

Maintainer Sude Gurer <sudegurer4@gmail.com>

Description Functions for implementing robust methods for sparse functional linear regression.

License GPL-3

R topics documented:

`dgp` *Generate functional data for the scalar-on-function regression model (sparse or non-sparse)*

Description

This function is used to simulate data for the (sparse and non-sparse) scalar-on-function regression model

$$Y = \beta_0 + \int X(t)\beta(t)dt + \epsilon,$$

where Y denotes the scalar response, $X(t)$ denotes the functional predictor, $\beta(t)$ denotes the regression coefficient function, and ϵ is the error process.

Usage

```
dgp(n, nknots, norder, domain = c(0, 1), snr, simind, out.p = 0)
```

Arguments

`n`	An integer, specifying the number of observations for each variable to be generated.
`nknots`	An integer, denoting the number of cubic B-spline basis functions defined on the domain.
`norder`	An integer, denoting the degree of B-spline basis.
`domain`	A vector with two elements, denoting the starting and end points of the fine grid. Default is [0,1].
`snr`	An integer, denoting the signal-to-ratio. Default value is 5.
`simind`	An integer. If 1, then the non-sparse functional dataset is generated. If 2, then, sparse functional dataset is generated.
`out.p`	An integer between 0 and 1, denoting the outlier percentage in the generated data.

Details

In the data generation process, first, the functional predictor is simulated at 501 equally spaced points within the interval [0,1] based on the following process:

$$X_i(t) = \sum a_{i,j} B_j(t),$$

where $B_j(t)$ represents cubic B-spline basis functions defined on 50 equally spaced knots over [0, 1], and the coefficients a_{ij} are sampled from a standard normal distribution. If simind = 1, then, the regression coefficient function is generated as follows:

$$\beta(t) = 3te^{t^2}\cos(3\pi t) + 1$$

. If simind = 2, then, the regression coefficient function is generated as follows:

$$\beta(t) = \sin(4\pi t)e^{-10t^2}$$

with a zero sub-region occurring after the discrete point 368 within the 501 equally spaced points in the interval [0, 1].

Value

X	A matricex containing the observations of simulated functional predictor variable.
Y	A vector containing the observations of simulated scalar response variable.
b	A vector containing the generated regression coefficient function.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

Examples

```
library(fda.usc)

nknots <- 50
norder <- 4
snr <- 5
n <- 100
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1), snr=snr, simind=simind, out.p = 0)
y  = data$Y
x = data$X
b <- data$b

fx <- fdata(x, argvals = seq(0, 1, length.out = 501))
plot(fx, lty = 1, ylab = "", xlab = "Grid point",
     main = expression(X(t)), mgp = c(2, 0.5, 0), ylim = range(fx))
fb <- fdata(b, argvals = seq(0, 1, length.out = 501))
plot(fb, lty = 1, ylab = "", xlab = "Grid point",
     main = expression(beta(t)), mgp = c(2, 0.5, 0), ylim = range(fb))
```

`fpcr` *Functional principal component regression*

Description

This function is used to perform classical scalar-on-function regression model

$$Y = \beta_0 + \int X(t)\beta(t)dt + \epsilon,$$

based on functional principal component decomposition of the functional predictor.

Usage

```
fpcr(y, x, nbf = NULL, ncomp = NULL, gp = NULL, nfold=10, CV = TRUE,
as = 1:5, Bs = c(4, 5, 8, 10, 15, 20))
```

Arguments

`y`	A vector containing the observations of scalar response Y, where n denotes the sample size.
`x`	A matrix containing the observations of functional predictor $X(t)$.
`nbf`	An integer denoting the number of B-spline basis expansion functions. Default is NULL.
`ncomp`	An integer denoting the number of functional principal components to be computed for the functional predictor.
`gp`	A vector containing the grid points of the functional predictor.
`nfold`	An integer denoting the number of folds used in the k-fold cross validation. Default value is 10.
`CV`	Logical. If TRUE, then, nfold cross-validation is used to find optimum values of `nbf`, and `ncomp`. If FALSE, then the specified `nbf` and `ncomp` values are used in the model.
`as`	A vector containing the candidate elements for the `ncomp`.
`Bs`	A vector containing the candidate elements for the `nbf`.

Details

When performing a scalar-on-function regression model based on the functional principal component analysis, first, the functional predictor $X(t)$ is decomposed by the functional principal component analysis method:

$$X(t) = \bar{X}(t) + \sum_{k=1}^{K} a_k \phi_k(t),$$

where $\bar{X}(t)$ is the mean function, $\phi_k(t)$ is the weight function, and $a_k = \int (X(k) - \bar{X}(k))\phi_k(t)$ is the principal component score for the functional predictor.

If `CV = TRUE`, then, a two-dimensional grid search (`as` and `Bs` vectors) with 10-fold cross-validation is used to determine optimum values of the number of principal components (`ncomp`) and the number of B-spline basis expansion functions (`nbf`).

Value

`y`	Scalar response variable.
`x`	Functional predictor variable.
`f.coeff`	A vector containing estimated regression coefficient function by the functional principal component regression.
`fitted.values`	A vector containing the residuals.
`coeffs`	A vector containing the estimated model parameters in the finite-dimensional space.
`model.details`	A list object containing model details, such as grid points and functional principal components.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

Examples

```
nknots <- 50
norder <- 4
snr <- 5
n <- 100
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1),
snr=snr, simind=simind, out.p = 0.1)
y  = data$Y
x = data$X
b <- data$b

model.fpcr <- fpcr(y, x)
```

---

fplsR — *Functional partial least squares regression*

Description

This function is used to perform both classical and robust scalar-on-function regression model

$$Y = \beta_0 + \int X(t)\beta(t)dt + \epsilon,$$

based on functional partial least squares decomposition of the functional predictor.

Usage

```
fplsR(y, x, gp = NULL, a = NULL, B = NULL, probp1 = 0.95, hampelp2 = 0.975,
hampelp3 = 0.999, numit = 100, prec = 0.01, type = c("classical", "robust"),
nfold=10, CV = TRUE, as = 1:5, Bs = c(4, 5, 8, 10))
```

Arguments

y	A vector containing the observations of scalar response Y, where n denotes the sample size.
x	A matrix containing the observations of functional predictor $X(t)$.
gp	A vector containing the grid points of the functional predictor.
a	An integer denoting the number of functional partial least squares components to be computed for the functional predictor.
B	An integer denoting the number of B-spline basis expansion functions. Default is NULL.

`probp1`	A numeric value used to determine the first outlier cutoff point for the weights.
`hampelp2`	A numeric value used to determine the first outlier cutoff point for the weights.
`hampelp3`	A numeric value used to determine the third outlier cutoff point for the weights.
`numit`	n integer value defining the maximum iteration used to achieve convergence.
`prec`	A numeric value used for the precision of the coefficient estimate.
`type`	Method type used to estimate the scalar-on-function linear regression model. Possibilities are "classical" and "robust".
`nfold`	An integer denoting the number of folds used in the k-fold cross validation. Default value is 10.
`CV`	Logical. If TRUE, then nfold cross-validation is used to find optimum values of `a`, and `B`. If FALSE, then the specified `a` and `B` values are used in the model.
`as`	A vector containing the candidate elements for the `a`.
`Bs`	A vector containing the candidate elements for the `B`.

Details

If `type = "classical"`, then, the NIPALS algorithm is used to obtain functional partial least squares regression components.

If `type = "robust"`, then, the partial least squares regression algorithm Serneels et al. (2005) is used to obtain functional partial least squares regression components.

Value

`fitted.values`	A vector containing the residuals.
`coef`	A vector containing estimated regression coefficient function by the functional partial least squares regression.
`intercept`	intercept

A numeric value containing the estimated intercept.

`residuals`	A vector containing the residuals.
`gp`	A vector containing the grid points.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

References

S. Serneels and C. Croux and P. Filzmoser and P. J. V. Espen (2005), "Partial robust M-regression", *Chemometrics and Intelligent Laboratory Systems*, **79**(1-2), 55-64.

Examples

```
nknots <- 50
norder <- 4
snr <- 5
n <- 100
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1),
snr=snr, simind=simind, out.p = 0.1)
y  = data$Y
x = data$X
b <- data$b

model.fplsR <- fplsR(y, x, type = "classical")
model.RfplsR <- fplsR(y, x, type = "robust")
```

---

predict_fpcr — *Prediction for a scalar-on-function linear regression model based on functional principal component analysis*

Description

This function is used to make prediction for a new set of functional predictor based upon a fitted scalar-on-function linear regression model in the output of `fpcr`.

Usage

```
predict_fpcr(object, xnew)
```

Arguments

`object`	An output object obtained from `fpcr`.
`xnew`	A matrix consisting of the new observations of functional predictor. The argument `xnew` must have the same length and the same structure as the input `x` of `fpcr`.

Value

A vector of predicted values of the scalar response variable for the given new functional predictor `xnew`.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

Examples

```
nknots <- 50
norder <- 4
snr <- 5
n <- 100
ntest = 5000
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1),
snr=snr, simind=simind, out.p = 0.1)
y  = data$Y
x = data$X

data.test <- dgp(n=ntest, nknots=nknots, norder=norder,
domain = c(0, 1), snr=snr, simind=simind, out.p = 0)
ytest <- data.test$Y
xtest <- data.test$X

model.fpcr <- fpcr(y, x)
predictions.fpcr <- predict_fpcr(object = model.fpcr, xnew = xtest)
```

---

predict_fpls — *Prediction for a scalar-on-function linear regression model based on both classical and robust functional partial least squares regression*

Description

This function is used to make prediction for a new functional predictor based upon a fitted scalar-on-function linear regression model in the output of `fplsR`.

Usage

```
predict_fpls(object, xnew)
```

Arguments

`object`	An output object obtained from `fplsR`.
`xnew`	A matrix consisting of the new observations of functional predictor. The argument `xnew` must have the same length and the same structure as the input `x` of `fplsR`.

Value

A vector of predicted values of the scalar response variable for the given new functional predictor `xnew`.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

Examples

```
nknots <- 50
norder <- 4
snr <- 5
n <- 100
ntest = 5000
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1),
snr=snr, simind=simind, out.p = 0.1)
y  = data$Y
x = data$X

data.test <- dgp(n=ntest, nknots=nknots, norder=norder,
domain = c(0, 1), snr=snr, simind=simind, out.p = 0)
ytest <- data.test$Y
xtest <- data.test$X

model.fplsR <- fplsR(y, x, type = "classical")
model.RfplsR <- fplsR(y, x, type = "robust")

predictions.fplsR <- predict_fpls(object = model.fplsR, xnew = xtest)
predictions.RfplsR <- predict_fpls(object = model.RfplsR, xnew = xtest)
```

---

predict_sfpls — *Prediction for a scalar-on-function linear regression model based on both classical and robust sparse functional partial least squares regression*

Description

This function is used to make prediction for a new functional predictor based upon a fitted scalar-on-function linear regression model in the output of `sfplsR`.

Usage

```
predict_sfpls(object, xnew)
```

Arguments

`object`	An output object obtained from `sfplsR`.
`xnew`	A matrix consisting of the new observations of functional predictor. The argument `xnew` must have the same length and the same structure as the input `x` of `sfplsR`.

Value

A vector of predicted values of the scalar response variable for the given new functional predictor xnew.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

Examples

```
nknots <- 50
norder <- 4
snr <- 5
n <- 100
ntest = 5000
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1),
snr=snr, simind=simind, out.p = 0.1)
y  = data$Y
x = data$X

data.test <- dgp(n=ntest, nknots=nknots, norder=norder,
domain = c(0, 1), snr=snr, simind=simind, out.p = 0)
ytest <- data.test$Y
xtest <- data.test$X

model.sfplsR <- sfplsR(y, x, type = "classical")
model.RsfplsR <- sfplsR(y, x, type = "robust")

predictions.sfplsR <- predict_sfpls(object = model.sfplsR, xnew = xtest)
predictions.RsfplsR <- predict_sfpls(object = model.RsfplsR, xnew = xtest)
```

---

sfplsR *Sparse functional partial least squares regression*

Description

This function is used to perform both classical and robust scalar-on-function regression model

$$Y = \beta_0 + \int X(t)\beta(t)dt + \epsilon,$$

based on sparse functional partial least squares decomposition of the functional predictor.

Usage

```
sfplsR(y, x, gp = NULL, a = NULL, B = NULL, probp1 = 0.95, hampelp2 = 0.975,
hampelp3 = 0.999, numit = 100, prec = 0.01, type = c("classical", "robust"),
nfold=10, CV = TRUE, as = 1:5, etas = c(0, 0.1, 0.3, 0.5, 0.7, 0.9),
Bs = c(4, 5, 8, 10))
```

Arguments

y	A vector containing the observations of scalar response Y, where n denotes the sample size.
x	A matrix containing the observations of functional predictor $X(t)$.
gp	A vector containing the grid points of the functional predictor.
a	An integer denoting the number of functional partial least squares components to be computed for the functional predictor.
B	An integer denoting the number of B-spline basis expansion functions. Default is NULL.
probp1	A numeric value used to determine the first outlier cutoff point for the weights.
hampelp2	A numeric value used to determine the first outlier cutoff point for the weights.
hampelp3	A numeric value used to determine the third outlier cutoff point for the weights.
numit	n integer value defining the maximum iteration used to achieve convergence.
prec	A numeric value used for the precision of the coefficient estimate.
type	Method type used to estimate the scalar-on-function linear regression model. Possibilities are "classical" and "robust".
nfold	An integer denoting the number of folds used in the k-fold cross validation. Default value is 10.
CV	Logical. If TRUE, then nfold cross-validation is used to find optimum values of `a`, and `B`. If FALSE, then the specified `a` and `B` values are used in the model.
as	A vector containing the candidate elements for the `a`.
etas	A vector containing the candidate elements for the sparsity parameter.
Bs	A vector containing the candidate elements for the `B`.

Details

If `type = "classical"`, then, the sparse NIPALS algorithm of Lee et. al. (2011) is used to obtain functional partial least squares regression components.

If `type = "robust"`, then, the robust sparse partial least squares regression algorithm Hoffmann et al. (2015) is used to obtain functional partial least squares regression components.

Value

fitted.values	A vector containing the residuals.
coef	A vector containing estimated regression coefficient function by the functional partial least squares regression.
intercept	intercept

A numeric value containing the estimated intercept.

residuals	A vector containing the residuals.
gp	A vector containing the grid points.

Author(s)

Sude Gurer, Han Lin Shang, Abhijit Mandal, and Ufuk Beyaztas

References

D. Lee and W. Lee and Y. Lee and Y.Pawitan (2011), "Sparse partial least-squares regression and its applications to high-throughput data analysis", *Chemometrics and Intelligent Laboratory Systems* **109**(1), 1–8.

I. Hoffmann and S.Serneels and P. Filzmoser and C. Croux (2015), "Sparse partial robust M regression", *Chemometrics and Intelligent Laboratory Systems*, **149**, 50–59.

Examples

```
nknots <- 50
norder <- 4
snr <- 5
n <- 100
domain <- c(0,1)
simind <- 2

data <- dgp(n=n, nknots=nknots, norder=norder, domain = c(0, 1),
snr=snr, simind=simind, out.p = 0.1)
y  = data$Y
x = data$X
b <- data$b

model.sfplsR <- sfplsR(y, x, type = "classical")
model.RsfplsR <- sfplsR(y, x, type = "robust")
```

Index