Package 'FunctanSNP'

October 18, 2022

Type Package

Title Functional Analysis (with Interactions) for Dense SNP Data

Version 0.1.0

Maintainer Rui Ren <xmurr@stu.xmu.edu.cn>

Description An implementation of revised functional regression models for multiple genetic variation data, such as single nucleotide polymorphism (SNP) data, which provides revised functional linear regression models, partially functional interaction regression analysis with penalty-based techniques and corresponding drawing functions, etc.(Ruzong Fan, Yifan Wang, James L. Mills, Alexander F. Wilson, Joan E. Bailey-Wilson, and Momiao Xiong (2013) <doi:10.1002/gepi.21757>).

License GPL (>= 2)

Depends R (>= 3.5.0)

Imports caret, glmnet, lava, MASS, stats, fda, funData, graphics

Encoding UTF-8

RoxygenNote 6.1.1

NeedsCompilation no

Author Rui Ren [aut, cre],
Kuangnan Fang [aut],
Qingzhao Zhang [aut],
Shuangge Ma [aut]

Repository CRAN

Date/Publication 2022-10-18 12:28:03 UTC

R topics documented:

coefSNPreg 2
plotGVF 3
plotRawdata 4
plotSNPreg 5
predictSNPreg 6
simData1 8

simData2 9
simX 10
SNPcvinter 11
SNPgvf 13
SNPinter 14
SNPlm 16

Index **19**

`coefSNPreg` *Extract coefficients from an SNPlm or SNPinter object*

Description

This functions predicts coefficients and related coefficient functions from a fitted "SNPlm" or "SNPinter" object.

Usage

```
coefSNPreg(x)
```

Arguments

`x`	an "SNPlm"/"SNPinter" object obtained by `SNPlm`/`SNPinter`.

Value

An "coef.SNPreg" object that contains the list of the following items.

- alpha: estimated intercept value.
- gamma: estimated coefficients of the scalar covariates.
- b: estimated coefficients of the chosen basis functions for the genetic effect function beta(t) (and interaction items betak(t), if x is an "SNPinter" object).
- betat: an "fd" object, representing the estimated genetic effect function beta(t).

See Also

See Also as `SNPlm`, `SNPinter`.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata1 <- simData1(n, m, seed = 123)
SNPlmres <- SNPlm(y = simdata1$y, z = simdata1$z,
                  location = simdata1$location, X = simdata1$X,
                  type1 = "Bspline", type2 = "Bspline", nbasis1 = 5,
                  nbasis2 = 5, params1 = 4, params2 = 4,
```

```
                  intercept = FALSE, Plot = FALSE)
coef1 <- coefSNPreg(x = SNPlmres)
coef1$alpha ###intercept
coef1$gamma ###coefficients of scalar variables
coef1$b     ###coefficients of basis functions

simdata2 <- simData2(n, m, seed = 123)
lambda1 <- 0.05
lambda2 <- sqrt(3)*lambda1
eta <- 0
SNPinterres <- SNPinter(y = simdata2$y, z = simdata2$z,
                        location = simdata2$location, X = simdata2$X,
                        lambda1, lambda2, eta, type1 = "Bspline", nbasis1 = 5,
                        params1 = 4, Bsplines = 5, norder = 4, intercept = TRUE,
                        eps = 1e-2, maxstep = 1e2, Plot = FALSE)
coef2 <- coefSNPreg(x = SNPinterres)
coef2$alpha ###intercept
coef2$gamma ###coefficients of scalar variables
coef2$b     ###coefficients of basis functions
```

---

plotGVF *Plot the genetic variant function*

Description

Plot the estimated genetic variant function obtained by SNPgvf or a given functional data object(s).

Usage

```
plotGVF(x, y = NULL, Lfdobj = 0, href = TRUE, titles = NULL,
  xlim = NULL, ylim = NULL, xlab = NULL, ylab = NULL,
  ask = FALSE, nx = NULL, axes = NULL)
```

Arguments

x	functional data object(s) to be plotted, such as the estimated genetic variant function obtained by SNPgvf.
y	sequence of points at which to evaluate the functions 'x' and plot on the horizontal axis. Defaults to seq(rangex[1], rangex[2], length = nx).
Lfdobj	either a nonnegative integer or a linear differential operator object. If present, the derivative or the value of applying the operator is plotted rather than the functions themselves.
href	a logical variable: If TRUE, add a horizontal reference line at 0.
titles	a vector of strings for identifying curves
xlim	a vector of length 2 containing axis limits for the horizontal axis.
ylim	a vector of length 2 containing axis limits for the vertical axis.

xlab	a label for the horizontal axis.
ylab	a label for the vertical axis.
ask	a logical value: If TRUE, each curve is shown separately, and the plot advances with a mouse click
nx	the number of points to use to define the plot. The default is usually enough, but for a highly variable function more may be required.
axes	Either a logical or a list or NULL. • logical whether axes should be drawn on the plot • list a list used to create custom axes used to create axes via x$axes[[1]] and x$axes[-1]. The primary example of this uses list("axesIntervals", ...)

Value

plot the estimated genetic variant function.

See Also

See Also as SNPgvf.

Examples

```
library(FunctanSNP)
n <- 20
m <- 50
simdata <- simX(n, m, seed = 1, d.ratio = 0)
X <- simdata$X
location <- simdata$location
SNPgvfres <- SNPgvf(location, X, type = "Bspline", nbasis = 5, params = 4, Plot = FALSE)
plotGVF(SNPgvfres)
```

---

plotRawdata *Visualization of the sequence (genotypes) data*

Description

Display the sequence data, such as the data generated by simX. The horizontal axis represents the sampling sites, and the vertical axis represents the sequence values containing only 0, 1 and 2.

Usage

```
plotRawdata(location, X, color = NULL, pch = 16, cex = 0.9)
```

Arguments

location	a numeric vector defining the sampling sites of the sequence data.
X	a matrix specifying the sequence data, with the number of rows equal to the number of samples.
color	A specification for the default plotting color. See graphical parameters (par) for more details.
pch	Either an integer specifying a symbol or a single character to be used as the default in plotting points. See graphical parameters (par) for more details.
cex	A numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. See graphical parameters (par) for more details.

Value

show the sequence data.

See Also

See Also as simX, SNPgvf.

Examples

```
library(FunctanSNP)
n <- 20
m <- 50
simdata <- simX(n, m, seed = 1, d.ratio = 0)
X <- simdata$X
location <- simdata$location
plotRawdata(location, X = X[1:2, ])
```

---

plotSNPreg *Plot coefficient functions and residuals from an SNPlm or SNPinter object*

Description

Produces a residual diagram or a plot of the genetic effect function beta(t) (and interaction items betak(t), if the entry x is an "SNPinter" object) for a fitted "SNPlm" or "SNPinter" object.

Usage

```
plotSNPreg(x, type)
```

Arguments

x	an "SNPlm"/"SNPinter" object obtained by SNPlm/SNPinter.
type	character, "beta" or "residuals" type, if "beta", plot the genetic effect function beta0(t) (and interaction items betak(t), if the entry x is an "SNPinter" object), if "residuals", show the residual plot.

Value

show the residual plot or coefficient functions.

See Also

See Also as SNPlm, SNPinter.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata1 <- simData1(n, m, seed = 123)
SNPlmres <- SNPlm(y = simdata1$y, z = simdata1$z,
                  location = simdata1$location, X = simdata1$X,
                  type1 = "Bspline", type2 = "Bspline", nbasis1 = 5,
                  nbasis2 = 5, params1 = 4, params2 = 4,
                  intercept = FALSE, Plot = FALSE)
plotSNPreg(x = SNPlmres, type = "beta")
plotSNPreg(x = SNPlmres, type = "residuals")

simdata2 <- simData2(n, m, seed = 123)
lambda1 <- 0.05
lambda2 <- sqrt(3)*lambda1
eta <- 0
SNPinterres <- SNPinter(y = simdata2$y, z = simdata2$z,
                        location = simdata2$location, X = simdata2$X,
                        lambda1, lambda2, eta, type1 = "Bspline", nbasis1 = 5,
                        params1 = 4, Bsplines = 5, norder = 4, intercept = TRUE,
                        eps = 1e-2, maxstep = 1e2, Plot = FALSE)
plotSNPreg(x = SNPinterres, type = "beta")
plotSNPreg(x = SNPinterres, type = "residuals")
```

---

predictSNPreg *Make predictions from an SNPlm or SNPinter object*

Description

This functions predicts fitted values for newly entered data from a fitted "SNPlm" or "SNPinter" object.

Usage

```
predictSNPreg(x, newz, newX)
```

Arguments

x	an "SNPlm"/"SNPinter" object obtained by SNPlm/SNPinter.
newz	matrix of new values for z at which predictions are to be made.
newX	matrix of new values for X at which predictions are to be made.

Value

- predicted.values: the predicted mean values.

See Also

See Also as SNPlm, SNPinter.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata1 <- simData1(n, m, seed = 123)
SNPlmres <- SNPlm(y = simdata1$y[1:200], z = simdata1$z[1:200, ],
                  location = simdata1$location, X = simdata1$X[1:200, ],
                  type1 = "Bspline", type2 = "Bspline", nbasis1 = 5,
                  nbasis2 = 5, params1 = 4, params2 = 4, intercept = FALSE,
                  Plot = FALSE)
predict.values1 <- predictSNPreg(x = SNPlmres, newz = simdata1$z[-(1:200), ],
                                 newX = simdata1$X[-(1:200), ])

simdata2 <- simData2(n, m, seed = 123)
lambda1 <- 0.05
lambda2 <- sqrt(3)*lambda1
eta <- 0
SNPinterres <- SNPinter(y = simdata2$y[1:200], z = simdata2$z[1:200, ],
                        location = simdata2$location, X = simdata2$X[1:200, ],
                        lambda1, lambda2, eta, type1 = "Bspline",
                        nbasis1 = 5, params1 = 4, Bsplines = 5, norder = 4,
                        intercept = TRUE, eps = 1e-2, maxstep = 1e2, Plot = FALSE)
predict.values2 <- predictSNPreg(x = SNPinterres, newz = simdata2$z[-(1:200), ],
                                 newX = simdata2$X[-(1:200), ])
```

simData1 *Generate simulated data for* SNPlm

Description

Generate simulated data for users to apply the method SNPlm, including response variable y, scalar variable Z and sequence (genotypes) data X.

Usage

```
simData1(n, m, seed)
```

Arguments

n	an interger variable specifying the number of samples to be generated.
m	an interger variable specifying the sequence length of each sample.
seed	an integer variable specifying the random seed used for random sequence generation.

Value

An "simData1" object that contains the list of the following items.

- y: a numeric vector representing the response variables.
- z: a matrix representing the scalar covariates, with the number of rows equal to the number of samples.
- location: a numeric vector defining the sampling sites of the sequence (genotypes) data.
- X: a matrix representing the sequence data, with the number of rows equal to the number of samples.
- beta: an "fd" object, representing the genetic effect function.

See Also

See Also as simX, SNPlm.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata1 <- simData1(n, m, seed = 123)
```

simData2 *Generate simulated data for* SNPinter *and* SNPcvinter

Description

Generate simulated data for users to apply the method SNPinter and SNPcvinter, including response variable y, scalar variable Z and sequence (genotypes) data X.

Usage

```
simData2(n, m, seed)
```

Arguments

n	an interger variable specifying the number of samples to be generated.
m	an interger variable specifying the sequence length of each sample.
seed	an integer variable specifyinging the random seed used for random sequence generation.

Value

An "simData2" object that contains the list of the following items.

- y: a numeric vector representing the response variables.
- z: a matrix representing the scalar covariates, with the number of rows equal to the number of samples.
- location: a numeric vector defining the sampling sites of the sequence (genotypes) data.
- X: a matrix representing the sequence data, with the number of rows equal to the number of samples.
- beta: an "fd" object specifying the genetic effect function.

See Also

See Also as simData1, SNPinter, SNPcvinter.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata2 <- simData2(n, m, seed = 123)
```

simX *Generate the sequence (genotypes) data containing only 0, 1 and 2*

Description

This function provides a method for generating sequence data containing only 0, 1 and 2, which can be used to simulate the generation of sequence genotypes.

Usage

```
simX(n, m, seed = 1, d.ratio = 0)
```

Arguments

n	an interger variable specifying the number of samples to be generated.
m	an interger variable specifying the sequence length of each sample.
seed	an integer variable specifying the random seed used for random sequence generation.
d.ratio	a numeric variable between 0 and 1 indicating the deletion ratio of sample sequences, default value is 0.

Value

An "simX" object that contains the list of the following items.

- location: a numeric vector defining the sampling sites of the sequence data.
- X: a matrix with n rows and m columns representing the sequence data.

See Also

See Also as plotRawdata, SNPgvf.

Examples

```
library(FunctanSNP)
n <- 2
m <- 50
simdata1 <- simX(n, m, seed = 1, d.ratio = 0)
simdata2 <- simX(n, m, seed = 1, d.ratio = 0.3)
plotRawdata(location = simdata1$location, X = simdata1$X)
plotRawdata(location = simdata2$location, X = simdata2$X)
```

SNPcvinter *Cross-validation for* SNPinter

Description

Performs K-fold cross validation for the revised partially functional interaction regression analysis over a grid of values for the regularization parameter lambda1 and lambda2.

Usage

```
SNPcvinter(y, z, location, X, K, lambda1, lambda2 = NULL, eta = 0,
  type1, nbasis1, params1, Bsplines = 20, norder = 4,
  intercept = FALSE, eps = 1e-05, maxstep = 1e+05, Plot = FALSE)
```

Arguments

y	a numeric vector defining the response variables.
z	a matrix defining the scalar covariates, with the number of rows equal to the number of samples.
location	a numeric vector defining the sampling sites of the sequence data.
X	a matrix specifying the sequence (genotypes) data, with the number of rows equal to the number of samples.
K	an integer specifying the number of cross-validation folds, default is 5.
lambda1	a numeric vector specifying the sparsity penalty parameter to be determined.
lambda2	a numeric vector specifying the group sparsity penalty parameter to be determined.
eta	a numeric vector specifying the penalty parameter for smoothing analysis.
type1	a character specifying the type of the basis functions that constitutes the genetic variation function. The options are "Bspline", "Exponential", "Fourier", "Monomial", and "Power".
nbasis1	an integer specifying the number of basis functions that constitutes the genetic variation function.
params1	in addition to rangeval1 (a vector of length 2 giving the lower and upper limits of the range of permissible values for the genetic variation function) and nbasis1, all bases have one or two parameters unique to that basis type or shared with one other; • bspline: Argument norder = the order of the spline, which is one more than the degree of the polynomials used. This defaults to 4, which gives cubic splines. • exponential: Argument ratevec. In fda_2.0.2, this defaulted to 1. In fda_2.0.3, it will default to 0:1. • fourier: Argument period defaults to diff(rangeval).

- monomial/power: Argument exponents. Default = 0:(nbasis-1). For monomial bases, exponents must be distinct nonnegative integers. For power bases, they must be distinct real numbers.

`Bsplines`	an integer specifying the number of basis functions that constitutes the genetic effect function.
`norder`	an integer specifying the order of bsplines that constitutes the genetic effect function, which is one higher than their degree. The default of 4 gives cubic splines.
`intercept`	should intercept(s) be fitted (TRUE) or set to zero (default = FALSE).
`eps`	a numeric variable specifying the threshold at which the algorithm terminates, default is 1e-5.
`maxstep`	a numeric variable specifying the maximum iteration steps, default is 1e5.
`Plot`	should the estimated genetic effect function beta0(t) and interaction items betak(t) be plotted (TRUE) or not (default = FALSE).

Value

An "SNPcvinter" object that contains the list of the following items.

- lambda1Select: a numeric value of the sparsity penalty parameter selected by cross validation.
- lambda2Select: a numeric value of the group sparsity penalty parameter selected by cross validation.
- etaSelect: a numeric value of the smoothing parameter selected by cross validation.
- alpha: estimated intercept value..
- gamma: estimated coefficients of the scalar covariates.
- b: estimated coefficients of the chosen basis functions for the genetic effect function beta0(t) and interaction items betak(t).
- betat: an "fd" object, representing the estimated genetic effect function beta(t) and interaction items betak(t).
- residuals: the residuals, that is response minus fitted values.
- fitted.values: the fitted mean values.
- lambda1: a numeric vector specifying the sparsity penalty parameter for cross validation.
- lambda2: a numeric vector specifying the group sparsity penalty parameter for cross validation.
- eta: a numeric vector specifying the smoothing parameter for cross validation.
- CVerror: a numeric vector, containing the mean square errors on testing set during cross validation.

See Also

See Also as `simData2`, `SNPinter`.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata2 <- simData2(n, m, seed = 123)
y <- simdata2$y
z <- simdata2$z
location <- simdata2$location
X <- simdata2$X
lambda1 <- c(0.01, 0.05, 0.1)
lambda2 <- sqrt(3)*lambda1
SNPcvinterres <- SNPcvinter(y, z, location, X, K = 3, lambda1, lambda2, eta = 0,
                            type1 = "Bspline", nbasis1 = 5, params1 = 4, Bsplines = 5,
                        norder = 4, intercept = TRUE, eps = 1e-2, maxstep = 1e2, Plot = TRUE)
SNPcvinterres$lambda1Select
SNPcvinterres$lambda2Select
SNPcvinterres$alpha
SNPcvinterres$gamma
SNPcvinterres$b
```

SNPgvf *Transform the sequence (genotypes) data into the genetic variant function*

Description

This function conducts the ordinary linear square smoothing analysis and models the genotypes of an individual (such as the sequence data generated by simX containing only 0, 1, 2) as the genetic variant function X(t).

Usage

```
SNPgvf(location, X, type, nbasis, params, Plot = FALSE)
```

Arguments

location	a numeric vector defining the sampling sites of the sequence data.
X	a matrix specifyinging the sequence data, with the number of rows equal to the number of samples.
type	a character specifying the type of the basis functions. The options are "Bspline", "Exponential", "Fourier", "Monomial", and "Power".
nbasis	an integer specifying the number of basis functions.
params	in addition to rangeval (a vector of length 2 giving the lower and upper limits of the range of permissible values for the function) and nbasis, all bases have one or two parameters unique to that basis type or shared with one other:

- bspline: Argument norder = the order of the spline, which is one more than the degree of the polynomials used. This defaults to 4, which gives cubic splines.
- exponential: Argument ratevec. In fda_2.0.2, this defaulted to 1. In fda_2.0.3, it will default to 0:1.
- fourier: Argument period defaults to diff(rangeval).
- monomial/power: Argument exponents. Default = 0:(nbasis-1). For monomial bases, exponents must be distinct nonnegative integers. For power bases, they must be distinct real numbers.

`Plot` should the estimated genetic variant function X(t) be plotted (TRUE) or not (default = FALSE).

Value

An 'fd' object that contains the estimated genetic variant function.

See Also

See Also as `simX`, `plotRawdata`.

Examples

```
library(FunctanSNP)
n <- 20
m <- 50
simdata <- simX(n, m, seed = 1, d.ratio = 0)
X <- simdata$X
location <- simdata$location
SNPgvfres <- SNPgvf(location, X, type = "Bspline", nbasis = 5, params = 4, Plot = FALSE)
plotRawdata(location, X)
plotGVF(SNPgvfres)
```

SNPinter *Revised partially functional interaction regression analysis for the sequence (genotypes) data*

Description

This function conducts joint analysis, which includes all scalar covariates Z, genetic variant function X(t), and their interactions in a partially functional interaction regression model. In addition, this function identifies the relevant genetic variation sites with a local sparsity penalty-based method.

Usage

```
SNPinter(y, z, location, X, lambda1, lambda2 = NULL, eta, type1, nbasis1,
  params1, Bsplines = 20, norder = 4, intercept = FALSE,
  eps = 1e-05, maxstep = 1e+05, Plot = FALSE)
```

Arguments

y	a numeric vector defining the response variables.
z	a matrix defining the scalar covariates, with the number of rows equal to the number of samples.
location	a numeric vector defining the sampling sites of the sequence data.
X	a matrix specifying the sequence (genotypes) data, with the number of rows equal to the number of samples.
lambda1	a numeric vector specifying the sparsity penalty parameter to be determined.
lambda2	a numeric vector specifying the group sparsity penalty parameter to be determined.
eta	a numeric vector specifying the penalty parameter for smoothing analysis.
type1	a character specifying the type of the basis functions that constitutes the genetic variation function. The options are "Bspline", "Exponential", "Fourier", "Monomial", and "Power".
nbasis1	an integer specifying the number of basis functions that constitutes the genetic variation function.
params1	in addition to rangeval1 (a vector of length 2 giving the lower and upper limits of the range of permissible values for the genetic variation function) and nbasis1, all bases have one or two parameters unique to that basis type or shared with one other; • bspline: Argument norder = the order of the spline, which is one more than the degree of the polynomials used. This defaults to 4, which gives cubic splines. • exponential: Argument ratevec. In fda_2.0.2, this defaulted to 1. In fda_2.0.3, it will default to 0:1. • fourier: Argument period defaults to diff(rangeval). • monomial/power: Argument exponents. Default = 0:(nbasis-1). For monomial bases, exponents must be distinct nonnegative integers. For power bases, they must be distinct real numbers. • para: some relevant parameters of "SNPlm" model.
Bsplines	an integer specifying the number of basis functions that constitutes the genetic effect function.
norder	an integer specifying the order of bsplines that constitutes the genetic effect function, which is one higher than their degree. The default of 4 gives cubic splines.
intercept	should intercept(s) be fitted (TRUE) or set to zero (default = FALSE).
eps	a numeric variable specifying the threshold at which the algorithm terminates, default is 1e-5.
maxstep	a numeric variable specifying the maximum iteration steps, default is 1e5.
Plot	should the estimated genetic effect function beta0(t) and interaction items betak(t) be plotted (TRUE) or not (default = FALSE).

Value

An "SNPinter" object that contains the list of the following items.

- alpha: estimated intercept value..
- gamma: estimated coefficients of the scalar covariates.
- b: estimated coefficients of the chosen basis functions for the genetic effect function beta0(t) and interaction items betak(t).
- betat: an "fd" object, representing the estimated genetic effect function beta(t) and interaction items betak(t).
- residuals: the residuals, that is response minus fitted values.
- fitted.values: the fitted mean values.

See Also

See Also as simData1, SNPcvinter.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata2 <- simData2(n, m, seed = 123)
y <- simdata2$y
z <- simdata2$z
location <- simdata2$location
X <- simdata2$X
lambda1 <- 0.05
lambda2 <- sqrt(3)*lambda1
eta <- 0
SNPinterres <- SNPinter(y, z, location, X, lambda1, lambda2, eta, type1 = "Bspline",
                     nbasis1 = 5, params1 = 4, Bsplines = 5, norder = 4, intercept = TRUE,
                        eps = 1e-2, maxstep = 1e2, Plot = TRUE)
SNPinterres$alpha
SNPinterres$gamma
SNPinterres$b
```

SNPlm *Revised functional linear regression analysis for the sequence (genotypes) data*

Description

This function models the genetic effect of genetic variants by relating the genetic variant function to the phenotype adjusting for covariates.

Usage

```
SNPlm(y, z, location, X, type1, type2, nbasis1, nbasis2, params1, params2,
  intercept = FALSE, Plot = FALSE)
```

Arguments

y	a numeric vector specifying the response variables.
z	a matrix specifying the scalar covariates, with the number of rows equal to the number of samples.
location	a numeric vector defining the sampling sites of the sequence data.
X	a matrix specifying the sequence data, with the number of rows equal to the number of samples.
type1	a character specifying the type of the basis functions that constitutes the genetic variation function. The options are "Bspline", "Exponential", "Fourier", "Monomial", and "Power".
type2	a character specifying the type of the basis functions that constitutes the genetic effect function. The options are "Bspline", "Exponential", "Fourier", "Monomial", and "Power".
nbasis1	an integer specifying the number of basis functions that constitutes the genetic variation function.
nbasis2	an integer specifying the number of basis functions that constitutes the genetic effect function.
params1	in addition to rangeval1 (a vector of length 2 giving the lower and upper limits of the range of permissible values for the genetic variation function) and nbasis1, all bases have one or two parameters unique to that basis type or shared with one other;
params2	in addition to rangeval1 (a vector of length 2 giving the lower and upper limits of the range of permissible values for the genetic effect function) and nbasis1, all bases have one or two parameters unique to that basis type or shared with one other; • bspline: Argument norder = the order of the spline, which is one more than the degree of the polynomials used. This defaults to 4, which gives cubic splines. • exponential: Argument ratevec. In fda_2.0.2, this defaulted to 1. In fda_2.0.3, it will default to 0:1. • fourier: Argument period defaults to diff(rangeval). • monomial/power: Argument exponents. Default = 0:(nbasis-1). For monomial bases, exponents must be distinct nonnegative integers. For power bases, they must be distinct real numbers.
intercept	should intercept(s) be fitted (TRUE) or set to zero (default = FALSE).
Plot	should the estimated genetic effect function beta(t) be plotted (TRUE) or not (default = FALSE).

Value

An "SNPlm" object that contains the list of the following items.

- alpha: estimated intercept value.
- gamma: estimated coefficients of the scalar covariates.
- b: estimated coefficients of the chosen basis functions for the genetic effect function beta(t).
- betat: an "fd" object, representing the estimated genetic effect function beta(t).
- residuals: the residuals, that is response minus fitted values.
- fitted.values: the fitted mean values.
- terms: the terms object used.
- model: if requested (the default), the model frame used.
- para: some relevant parameters of "SNPlm" model.

See Also

See Also as SNPgvf.

Examples

```
library(FunctanSNP)
n <- 300
m <- 30
simdata1 <- simData1(n, m, seed = 123)
y <- simdata1$y
z <- simdata1$z
location <- simdata1$location
X <- simdata1$X
SNPlmres <- SNPlm(y, z, location, X, type1 = "Bspline", type2 = "Bspline", nbasis1 = 5,
                  nbasis2 = 5, params1 = 4, params2 = 4, intercept = FALSE, Plot = TRUE)
SNPlmres$alpha
SNPlmres$gamma
SNPlmres$b
```

Index

coefSNPreg, 2

par, *5*
plotGVF, 3
plotRawdata, 4, *10*, *14*
plotSNPreg, 5
predictSNPreg, 6

simData1, 8, *9*, *16*
simData2, 9, *12*
simX, *5*, *8*, 10, *14*
SNPcvinter, *9*, 11, *16*
SNPgvf, *4, 5*, *10*, 13, *18*
SNPinter, *2*, *6, 7*, *9*, *11, 12*, 14
SNPlm, *2*, *6–8*, 16