Package ‘GgAM’

October 30, 2019

Type Package
Title Generalized geoAdditive Models
Version 0.1.0
Author Shan Yu, Xinyi Li, Jue Wang, Guannan Wang, Lily Wang
Maintainer <juew@iastate.edu>, <syu@iastate.edu>
Imports Triangulation, caret, geometry, ggplot2, magic, BPST, directlabels, stats, grpreg, MASS, psych, plotly, mgcv, Matrix, methods, colorRamps
Depends R (>= 2.10)
Description Variable selection in partial linear bivariate penalized spline models by Generalized Cross Validation or Cross Validation.
License GPL (>= 2)
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
Suggests R.rsp
VignetteBuilder R.rsp

R topics documented:

b .......................... 2
### Description

Function used in definition of smooth terms within `plbpsm` model formulae. The function does not evaluate a (spline) smooth - it exists purely to help set up a model using spline based smooths.

### Usage

```r
b(..., d = NULL, r = NULL, V = NULL, Tr = NULL, b = NULL, nt = NULL, Holes = NULL, B = NULL, Q2 = NULL, K = NULL, ind = NULL, lambda = NULL, fx = FALSE, id = NULL)
```

### Arguments

- `...`: a list of variables that are the covariates that this smooth is a function of.
- `d`: degree of polynomials.
- `r`: smoothness and $r \leq d$.
- `V`: an $N$ by two matrix that lists vertices with the $i$th row storing in Cartesian coordinates for the $i$th vertex. $N$ is the number of vertices.
- `Tr`: a $K$ by three matrix that each row represents one triangle. All the elements are the integers that stand for the indices of vertices in $V$. $K$ is the number of triangles.
- `b`: Boundary of the domain of sample points.
- `nt`: A parameter controls the number of total triangles.
- `Holes`: Information of holes of polygon.
B Bernstei basis matrix

Q2 The Q matrix from QR decomposition of the constraint matrix.

K Energy matrix for constructing penalty matrix

ind An ordering indices of observation points, in which the cnt[j]+1th to cnt[j+1]th elements are indices of points in the jth triangle.

lambda The default set of smoothing penalty parameter to be chosen from

fx indicates whether the term is a fixed d.f. regression spline (TRUE) or a penalized regression spline (FALSE).

id An identifying label or number for the smooth, linking it to other smooths. Defaults to NULL for no linkage.

Value

These smooth.spec objects define bivariate smooths and are turned into bases and penalties by BasisCon functions. The returned object contains the following items:

d degree of polynomials.

r smoothness and \( r \leq d \).

V an N by two matrix that lists vertices with the i\text{th} row storing in Cartesian coordinates for the i\text{th} vertex. \( N \) is the number of vertices.

Tr a K by three matrix that each row represents one triangle. All the elements are the integers that stand for the indices of vertices in V. \( K \) is the number of triangles.

ind An ordering indices of observation points corresponding to index of triangles.

B Bernstein basis matrix

K Energy matrix for constructing penalty matrix

lambda The default set of smoothing penalty parameter to be chosen from

term An array of text strings giving the names of the covariates that the term is a function of.

fixed TRUE if the term is to be treated as a pure regression spline (with fixed degrees of freedom); FALSE if it is to be treated as a penalized regression spline

dim The dimension of the smoother - i.e. the number of covariates that it is a function of.

label A suitable text label for this smooth term.

id An identifying label or number for the smooth, linking it to other smooths. Defaults to NULL for no linkage.

---

BasisCon

Basis Construction Function

Description

Smooth terms in a plpsm formula are turned into smooth specification objects of class xx.smooth.spec during processing of the formula. Each of these objects is converted to a smooth object using an appropriate BasisCon function.
BasisCon.bivariate.smooth

Usage

BasisCon(object, data)

Arguments

object is a smooth specification object or a smooth object.
data A data frame, model frame or list containing the values of the (named) covariates at which the smooth term is to be evaluated. If it’s a list then n must be supplied.

Details

It is the wrapper function which calls basis constructing method.

Value

a list of smooth objects are returned. Many of the outputs are from b function. Other outputs include all the information related to berstein basis.

Examples

library(BPST)
library(Triangulation)
Tri.hs <- TriMesh(horseshoe, n = 4) # create triangulation mesh
V <- Tri.hs$V # vetices of triangulation
Tr <- Tri.hs$Tr # triangles of triangulation
xx <- seq(-1, 4, 0.1)
yy <- seq(-1, 1, 0.1)
S0 <- expand.grid(xx, yy)
# bivariate spline
BI <- BasisCon(b(Var1, Var2, d = 2, r = 1, V = V, Tr = Tr), S0)
# univariate spline
U <- BasisCon(u(Var1),S0)
Details

It is the wrapper function which calls basis constructing method.

Value

A list of smooth objects are returned. Many of the outputs are from \( f \) function. Other outputs include all the information related to Bernstein basis.

---

**Beta_se**

*Standard Error Estimation for Linear Coefficients*

Description

This is an internal function of package ggam.

Usage

```r
beta_se(UB, B, Q2, K, lam1, lam2, Z, Y, ind.c, VS = FALSE, ...)
```
Arguments

UB  The univariate basis function matrix constructed.
B   The bernstein basis matrix.
Q2  The Q matrix from QR decomposition of the constraint matrix.
K   The energy matrix.
lam1 The smoothing penalty parameter.
lam2 The variable selection penalty parameter.
Z   The parametric matrix.
Y   Response variable.
ind.c The indexed for parametric coefficient for which standard error want to be calculated.
VS  variable section is conducted or not.
... other arguments.

Details

A sandwich formula is developed to find the standard error for $\beta$. The detailed algorithm is in the paper Wang et al. (2018).

Value

se_beta The standard error of linear coefficients.
Ve   The estimated covariance matrix.

---

cv.plbpsm Cross-validation for plbpsm

---

Description

Performs k-fold cross validation for GGAMs

Usage

cv.plbpsm(formula, data, kfold = 10, repeat.times = 1, family,
          group = NULL, ...)

Arguments

formula a PLBPSM formula. These are exactly like the formula for a GLM except that univariable and bivariate smooth terms.
data   A data frame or list containing the model response variable and covariates required by the formula. By default the variables are taken from environment(formula).
kfold number of folds splitted
repeat.times repeated times for cross validation
family This is a family object specifying the distribution and link to use in fitting etc. See glm and family for more details.
group  whether model identification is conducted for each covariates
... other arguments.
**Data_Generator**

**Data Generating function**

**Description**

Data Generating function

**Usage**

Data_Generator(F1, F2, F3, n, p.x, mu0, sig, scale, family)

**Arguments**

- **n**: Sample size.
- **p.x**: Dimension of nonlinear components.
- **sig**: Standard deviation of random error.
- **family**: Distribution family.
- **mu0**: Intercept.
- **scale**: Scale parameter in gamma distribution.
- **family**: Distribution family.

Output Arguments: (1) y: Generated response vector. (2) z: Generated design matrix of linear part; centralized for continuous z. (3) x: Generated design matrix of nonlinear part. (4) fxTrue: Generated nonlinear functions values at generated x.

---

**formula.plbpsm**

**PLBPSM formula**

**Description**

Description of plbpsm formula (see Details), and how to extract it from a fitted plbpsm object.

**Usage**

## S3 method for class 'plbpsm'
formula(x, ...)

**Arguments**

- **x**: Fitted model objects of class gam (see plbpsmObject) as produced by plbpsm().
- **...**: Un-used in this case.
Details

The formulae supplied to \texttt{plbpsm} are exactly like those supplied to \texttt{glm} except that univatiable and bivariate smooth terms, \texttt{u} and \texttt{b} can be added to the right hand side (and \texttt{.} is not supported in \texttt{plbpsm} formulae). Smooth terms are specified by expressions of the form: \( f(x_1, x_2, \ldots, r=1, d=2, fx=\text{FALSE}) \). If \( d \) is not specified then basis specific defaults are used. Note that these defaults are essentially arbitrary, and it is important to check that they are not so big that they cause oversmoothing. \( fx \) is used to indicate whether or not this term should be unpenalized. Formulae can involve nested or “overlapping” terms such as \( y=u(x)+u(z)+f(x,z) \).

Value

Returns the model formula, \texttt{x$formula}. Provided so that anova methods print an appropriate description of the model.

See Also

\texttt{plbpsm}

---

**ggrplsfilt**

Variable selection in Partial Linear Bivariate penalized Spline fit in generalized spatial model

Description

This is an internal function of package GgAM. Bivariate penalized least squares problem is solved with penalty parameter chosen by GCV or CV. Variable selection by using adaptive LASSO or group SCAD is applied in parametric coefficients.

Usage

\[
ggrplsfilt(G, criterion, method, family, ind_c, VS, control = \texttt{plbpsm.control()}, MI, \ldots)
\]

Arguments

- \texttt{G} An object of the type returned by \texttt{plbpsm} when \texttt{fit=FALSE}.
- \texttt{criterion} The criterion to choose the penalty parameter lambda. "GCV" to use generalized cross validation method and "CV" for cross validation
- \texttt{method} 'ALASSO' or 'SCAD' to penalize the coefficients for parametric part.
- \texttt{family} The family object, specifying the distribution and link to use.
- \texttt{ind_c} The given index of covariates that are selected.
- \texttt{VS} 'TRUE' for using ALASSO/SCAD to select linear variables.
- \texttt{control} A list of fit control parameters to replace defaults returned by \texttt{plbpsm.control}. Any control parameters not supplied stay at their default values.
- \texttt{MI} whether model identification is conducted or not.
- \texttt{...} other arguments.
Details

This is an internal function of package GgAM. We propose Iteratively Reweighted Least square based algorithm to get the pilot estimation and then use it to get a a spline-backfitted local polynomial estimation. The smoothing penalty parameter could be chosen by GCV or CV using the routines: gplsfitGCV.

Value

A list of fit information.

Description

This is an internal function of package ggam.

Usage

gplsfitGCV(Y, B, Q2, P, UB = NULL, lambda, family, offset, theta = 0, fx, control, start = NULL, etastart = NULL, mustart = NULL, X = NULL, ind_c = 1:ncol(X), fixedSteps = (control$maxstep + 1), ...)

Arguments

Y Response variable.
B The bernstein basis matrix.
Q2 The Q2 matrix from QR decomposition of the transpose of the constraint matrix.
P The penalty matrix.
UB The univariate basis function matrix constructed.
lambda The smoothing penalty parameter.
family The family object, specifying the distribution and link to use.
offset Can be used to supply a model offset for use in fitting. Note that this offset will always be completely ignored when predicting.
theta The given theta values in negative binomial family.
fx indicates whether the term is a fixed d.f. regression
control A list of fit control parameters to replace defaults returned by plbpsm.control. Any control parameters not supplied stay at their default values. spline (TRUE) or a penalized regression spline (FALSE).
start Initial values for model coefficients
etastart Initial values for linear predictor.
mustart Initial values for the expected response.
X The parametric model matrix. set to 'NULL' if it is not provided.
ind_c The vector of index to indicate the parametric part.
fixedSteps How many steps to take: useful when only using this routine to get rough starting values for other methods.
... other arguments.
Details


Value

A list of fit information.

gplsfitGCV_nb

Generalized Penalized Least Square Fit under GCV for negative binomial family

Description

This is an internal function of package ggam.

Usage

gplsfitGCV_nb(Y, B, Q2, P, UB, lambda, family, offset, r.theta = c(2, 8),
fx, control, X = NULL, ind_c = 1:ncol(X),
fixedSteps = (control$maxstep + 1), ...)

Arguments

Y Response variable.
B The bernstein basis matrix.
Q2 The Q2 matrix from QR decomposition of the transpose of the constraint matrix.
P The penalty matrix.
UB The univariate basis function matrix constructed.
lambda The smoothing penalty parameter.
family The family object, specifying the distribution and link to use.
offset Can be used to supply a model offset for use in fitting. Note that this offset will always be completely ignored when predicting.
r.theta All the theta values given.
fx indicates whether the term is a fixed d.f. regression spline (TRUE) or a penalized regression spline (FALSE).
control A list of fit control parameters to replace defaults returned by plbpsm.control. Any control parameters not supplied stay at their default values.
X The parametric model matrix. set to ’NULL’ if it is not provided.
ind_c The vector of index to indicate the parametric part.
fixedSteps How many steps to take: useful when only using this routine to get rough starting values for other methods.
... other arguments passed onto gplsfitGCV.

Details

In this function, the estimator of \( \theta \) is chosen to ensure that the Pearson estimate of the scale parameter is as close as possible to 1. The other parts follow from the routine of gplsfitGCV.
grplsf

Value

A list of fit information.

---

grplsfit  

Variable selection with Bivariate penalized Spline (GCV/CV) fit

Description

This is an internal function of package ggam. Bivariate penalized least squares problem is solved with penalty parameter chosen by GCV or CV. Variable selection by using adaptive LASSO or group SCAD is applied in parametric coefficients.

Usage

```
grplsfit(G, criterion, method, family, ind_c, VS, MI, ...)```

Arguments

- `G`: An object of the type returned by `plbpsm` when `fit=FALSE`.
- `criterion`: The criterion to choose the penalty parameter lambda. "GCV" to use generalized cross validation method and "CV" for cross validation.
- `method`: 'ALASSO' or 'SCAD' to penalize the coefficients for parametric part.
- `family`: The family object, specifying the distribution and link to use.
- `ind_c`: The given index of covariates that are selected.
- `VS`: 'TRUE' for using ALASSO/SCAD to select linear variables.
- `MI`: Whether model identification is conducted or not.
- `...`: other arguments.

Details

This is an internal function of package ggam. We propose a coordinate descent based algorithm to perform the variable selection efficiently. The smoothing penalty parameter could be chosen by GCV or CV using the routines: `plsfitGCV` and `plsfitCV`. In this function, the user can also choose whether to do variable selection or not.

Value

A list of fit information.
### Description

Prints 'Hello, World!'.

### Usage

```r
hello()
```

### Examples

```r
hello()
```

---

### plbpsm

**Partial Linear Bivariate Penalized Spline Model**

### Description

Fits a partial linear bivariate penalized spline model

### Usage

```r
plbpsm(formula, family = gaussian(), data = list(), ind_c = NULL, 
group = NULL, weights = NULL, na.action, offset = NULL, 
criterion = "GCV", method = "SCAD", control = list(), scale = 1, 
VS = FALSE, fit = TRUE, G = NULL, drop.unused.levels = TRUE, 
drop.intercept = NULL, ecdf = FALSE, backfitting = TRUE, ...)
```

### Arguments

- **formula**
  - a PLBPSM formula. These are exactly like the formula for a GLM except that
  univariable and bivariate smooth terms.
- **family**
  - This is a family object specifying the distribution and link to use in fitting etc (see `glm` and `family`).
- **data**
  - A data frame or list containing the model response variable and covariates re-
  quired by the formula. By default the variables are taken from environment(formula).
- **ind_c**
  - Defualt set to 'NULL', if not, it is the arbitrary chosen variables from the para-
  metric part
- **group**
  - A vector describing the grouping of the coefficients. For greatest efficiency and
  least ambiguity (see details), it is best if group is a factor or vector of consecutive
  integers, although unordered groups and character vectors are also allowed. If
  there are coefficients to be included in the model without being penalized, assign
  them to group 0 (or "0").
weights

weights on the contribution of the data to the log likelihood. Note that a weight of 2, for example, is equivalent to having made exactly the same observation twice. If you want to reweight the contributions of each datum without changing the overall magnitude of the log likelihood, then you should normalize the weights (e.g. weights <- weights/mean(weights)).

na.action

a function which indicates what should happen when the data contain ‘NA’s. The default is set by the ‘na.action’ setting of ’options’, and is ‘na.fail’ if that is unset. The ‘factory-fresh’ default is ‘na.omit’.

offset

Can be used to supply a model offset for use in fitting. Note that this offset will always be completely ignored when predicting, unlike an offset included in formula (this used to conform to the behaviour of lm and glm).

criterion

The criterion to choose the penalty parameter lambda. "GCV" to use generalized cross validation method and "CV" for cross validation.

method

The variable selection method for linear covariates. "SCAD" is the SCAD method in penalizing the coefficients for linear covariates. "ALASSO" is the adaptive LASSO method in penalizing the coefficient for linear covariates.

control

A list of fit control parameters to replace defaults returned by plbpsm.control. Any control parameters not supplied stay at their default values.

scale

scale parameter in exponential family

VS

'TRUE' for using ALASSO/SCAD to select linear variables

fit

If this argument is TRUE then gam sets up the model and fits it, but if it is FALSE then the model is set up and an object G containing what would be required to fit is returned is returned. See argument G.

G

Usually NULL, but may contain the object returned by a previous call to gam with fit=FALSE, in which case all other arguments are ignored except for scale, control, method and fit.

drop.unused.levels

by default unused levels are dropped from factors before fitting. For some smooths involving factor variables you might want to turn this off. Only do so if you know what you are doing.

drop.intercept

Set to TRUE to force the model to really not have the a constant in the parametric model part, even with factor variables present. Can be vector when formula is a list.

ecdf

the choice of whether empirical conditional density function is used.

backfitting

whether spline backfitted local polynomial estimation is applied.

...

further arguments for passing on e.g. to grplspit.

Details

A generalized geoadditive model (GgAM) is a generalized linear model (GLM) in which the linear predictor is given by user specified univariate functions plus additive functions and a bivariate function of the location covariates of the linear predictor. A simple example is:

\[
\log(E(y_i)) = u(z_1) + u(z_2) + f(x_{1i}, x_{2i})
\]

where the (independent) response variables \(y_i \sim \text{Poi} \), \(z_1\) and \(z_2\) are explanatory variables, \(f\) is the bivariate smooth function of covariates \(x_{1i}\) and \(x_{2i}\). The log is an example of a link function.
A Generalized Geoadditive Model (GgAM) is a generalized linear model (GLM) in which the linear predictor is given by a user specified bivariate functions of the location covariates plus a conventional parametric component of the linear predictor. A simple example is:

\[
\log(E(y_i)) = z_1 + z_2 + u_{x_1} + u_{x_2} + b(s_1, s_2)
\]

where the (independent) response variables \( y_i \sim \text{Poi} \), \( z_1 \) and \( z_2 \) are explantary variables, \( f \) is the bivariate smooth function of covariates \( x_{1i} \) and \( x_{2i} \). The log is an example of a link function.

Model structure identification process is contained in \texttt{plbpsm} before model fitting to identify the linear and nonlinear continuous variables by using group adaptive LASSO.

We incorporate a variable selection mechanism into the Partial linear Spatial Regression Model (PLSM) and propose a double penalized least squares approach based on bivariate spline approximation over the spatial domain when the link is gaussian.

\texttt{plbpsm} in GgAM chooses the smoothing penalty parameter by using the Generalized Cross Validation (GCV) criterion.

In terms of shrinkage penalty, Adaptive LASSO and SCAD penalty could be used to do variable selection under PLSM and GPLSM. Broadly \texttt{plbpsm} works by first constructing basis functions and penalty coefficient matrices for bivariate smooth term in the model formula, obtaining a model matrix for the parametric part of the model formula. The algorithm chooses penalty parameter based on GCV/CV criterion for the bivariate smoothing part and chooses significant linear covariates based on adaptive LASSO and SCAD method using BIC criterion. And then, the refit is applied to the choosen model to get the final fit.

\textbf{Value}

If \texttt{fit=FALSE} the function returns a list \texttt{G} of items needed to fit a PLBPSM, but doesn’t actually fit it. Otherwise the function returns an object of class "plbpsm" as described in \texttt{plbpsmObject}.

\textbf{References}


\textbf{Examples}

```
# Horseshoe Domain Example
rm(list = ls())
# library
library(mgcv)
library(splines)
library(bindata)
library(parallel)
library(BPST)
library(GgAM)
library(Triangulation)
```
library(caret)

# true univariate component functions, F1 is a linear function.
# F2 & F3 are nonlinear functions.
# beta_1 = 1
F1 <- function(x) {
  f1x <- x
  return(f1x)
}

# nonlinear univariate components: u_1
F2 <- function(x) {
  f2x <- -0.25 * cos(pi * x) ^ 2 + 0.5 * sin(pi * x) ^ 2 - 0.125
  return(f2x)
}

# nonlinear univariate components: u_2
F3 <- function(x) {
  f3x <- x + 3 * x ^ 2 - 0.25
  return(f3x)
}

# generating grid points and true functions
# univariate
X0 <- matrix(rep(seq(-0.5, 0.5, 0.01), each = 2), ncol = 2, byrow = TRUE)
m1 <- F2(X0[, 1])
m2 <- F3(X0[, 2])

# bivariate
# m0 are the true bivariate function at grid points within the
# horsedomain.
fsb <- list(fs.boundary())[[1]]
# generate grid points: S0
xx <- seq(-1, 4, 0.05)
yy <- seq(-1, 1, 0.05)
S0 <- expand.grid(xx, yy)
# m is the true value of bivariate function
v <- S0[, 1]
w <- S0[, 2]
m <- fs.test(v, w, b = 1)
# remove points out of the boundary
names(fsb) <- c("v", "w")
ind <- inSide(fsb, x = v, y = w)
S0 <- S0[ind, ]
m0 <- m[ind]
# m0 is the value of bivariate function at S0
m0 <- m0 - mean(m0)
m0 <- m0 / 2

# generate triangulation using TriMesh within R package: Triangulation.
Tri.hs <- TriMesh(horseshoe, n = 4) # create triangulation mesh
V <- Tri.hs$V # vertices of triangulation
Tr <- Tri.hs$Tr # triangles of triangulation

# settings in Simulation Study 1
p.x <- 3L # dimension for explanatory variable x
mu0 <- 0 # intercept
sig <- 1
family <- 'gaussian'
n <- 1000
set.seed(2)
data <- Data_Generator(F1, F2, F3, n, p.x, mu0, sig, scale = 1, family = family)
y.sam <- data$y # response variable
x.sam <- data$x # explanatory variable x
s.sam <- data$s # locations
# combine response variable, covariates and locations.
dat.poi <- as.data.frame(cbind(y.sam, x.sam, s.sam))
# change the column names in the data: dat.poi
colnames(dat.poi) <- c('y', 'x1', 'x2', 'x3', 's1', 's2')
# estimation
formula <- y ~ x1 + u(x2, N = 3) + u(x3, N = 3) + b(s1, s2, V = V, Tr = Tr, d = 2, r = 1) # indicate the formula
fit <- plbpsm(formula = formula, data = as.data.frame(dat.poi), family = family, group = c(0, 0)) # fit the model

plbpsm.control

Setting PLBPSM fitting defaults

Description
This is an internal function of package ggam which allows control of the numerical options for fitting a PLBPSM. Typically users will want to modify the defaults if model fitting fails to converge, or if the warnings are generated which suggest a loss of numerical stability during fitting.

Usage
plbpsm.control(delta1 = 1, delta2 = 1, trace = FALSE, maxstep = 10, epsilon = 1e-07)

Arguments

delta1 The convergence criterion in gplsfitGCV.
delta2 The convergence criterion in gplsfitGCV.
trace Set this to TRUE to turn on diagnostic output.
maxstep Maximum number of iterations to perform.
epsilon This is used for judging conversion of the loop in gplsfitGCV.

plot.plbpsm

Default PLBPSM plotting

Description
Takes a fitted plbpsm object produced by plbpsm() and plots the triangulation of location data points, predicted surface of bivariate smooth function and optionally produces histogram of residuals for the model.
## Usage

```r
## S3 method for class 'plbpsm'
plot(x, residuals = FALSE, pages = 0, select = NULL,
     xlab = NULL, ylab = NULL, main = NULL, ylim = NULL,
     xlim = NULL, n1 = 40, n2 = 40, ...)
```

### Arguments

- **x**: a fitted `plbpsm` object as produced by `plbpsm`.
- **residuals**: If `TRUE` then a histogram of standardized residuals will be added.
- **pages**: (default 0) the number of pages over which to spread the output. For example, if `pages=1` then all terms will be plotted on one page with the layout performed automatically. Set to 0 to have the routine leave all graphics settings as they are.
- **select**: Allows the plot for a single model term to be selected for printing. e.g. if you just want the plot for the second smooth term set `select=2`.
- **xlab**: If supplied then this will be used as the x label for all plots.
- **ylab**: If supplied then this will be used as the y label for all plots.
- **main**: Used as title for plots if supplied.
- **ylim**: If supplied then this pair of numbers are used as the y limits for each plot.
- **xlim**: If supplied then this pair of numbers are used as the x limits for each plot.
- **n1**: number of points used in x axis in each plot.
- **n2**: number of points used in y axis in each plot.
- **...**: other graphics parameters to pass on to plotting commands. See details for smooth plot specific options.

### Details

Used R package `fdaPDE` and `plotly` to draw triangulation plot and predicted surfaces. See `plbpsm:::plot.plbpsm.smooth`.

---

### plsfncv

#### Penalized Least Square Fit under CV

#### Description

This is an internal function of package `ggam`.

#### Usage

```r
plsfncv(B, Q2, K, lambda, Y, fx, fold = 5, Z = NULL)
```
**Arguments**

- **B**  
  The bernstein basis matrix.
- **Q2**  
  The Q2 matrix from QR decomposition of the transpose of the constraint matrix.
- **K**  
  The energy matrix to construct penalty matrix.
- **lambda**  
  The smoothing penalty parameter.
- **Y**  
  Response variable.
- **fx**  
  Indicates whether the term is a fixed d.f. regression spline (TRUE) or a penalized regression spline (FALSE).
- **fold**  
  Number of folders to do cross validation.
- **Z**  
  The parametric model matrix. Set to `"NULL"` if it is not provided.

**Details**

The method is a computationally efficient means of applying cross validation to the problem of smoothing parameter selection:

\[
\min_{\beta, \gamma} \frac{1}{2} \left\{ \| Y - Z\beta - B\gamma \|^2 + \lambda \gamma^T P \gamma \right\}
\]

subject to constraints \( H\gamma = 0 \). \( Z \) is a parametrix design matrix, \( \beta \) a parameter vector, \( Y \) a data vector, \( \gamma \) is the bernstein coefficients, \( B \) is the Bernstein basis matrix, \( H \) is constraint matrix.

**Value**

A list of fit information.

---

**Description**

This is an internal function of package `ggam`.

**Usage**

```
plsfitGCV(B, Q2, P, lambda, Y, fx, Z = NULL, ...)
```

**Arguments**

- **B**  
  The bernstein basis matrix.
- **Q2**  
  The Q2 matrix from QR decomposition of the transpose of the constraint matrix.
- **P**  
  The penalty matrix.
- **lambda**  
  The smoothing penalty parameter.
- **Y**  
  Response variable.
- **fx**  
  Indicates whether the term is a fixed d.f. regression spline (TRUE) or a penalized regression spline (FALSE).
- **Z**  
  The parametric model matrix. Set to `"NULL"` if it is not provided.
- **...**  
  Other arguments.
The method is a computationally efficient means of applying GCV to the problem of smoothing parameter selection:

$$\min_{\beta, \gamma} \frac{1}{2} \left\{ \| Y - Z\beta - B\gamma \|_2^2 + \lambda \gamma^T P\gamma \right\}$$

subject to constraints $H\gamma = 0$. $Z$ is a parametrix design matrix, $\beta$ a parameter vector, $Y$ a data vector, $\gamma$ is the Bernstein coefficients, $B$ is the Bernstein basis matrix, $H$ is constraint matrix.

Value

A list of fit information.

Arguments

- `object`: a fitted `plbpsm` object as produced by `plbpsm()`
- `newdata`: A data frame or list containing the values of the model covariates at which predictions are required. If this is not provided then predictions corresponding to the original data are returned. If `newdata` is provided then it should contain all the variables needed for prediction: a warning is generated if not.
- `type`: the type of prediction required. The default is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable. Thus for a default binomial model the default predictions are of log-odds (probabilities on logit scale) and `type = "response"` gives the predicted probabilities. The "terms" option returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale. When this has the value "link" the linear predictor (possibly with associated standard errors) is returned.
- `se.fit`: when this is `TRUE` (not default) standard error estimates are returned for each prediction.
- `terms`: if `type="terms"` then only results for the terms given in this array will be returned.
exclude

if type="terms" or type= "iterms" then terms (smooth or parametric) named in this array will not be returned. Otherwise any smooth terms named in this array will be set to zero. If NULL then no terms are excluded. Note that this is the term names as it appears in the model summary, see example.

block.size

maximum number of predictions to process per call to underlying code: larger is quicker, but more memory intensive. Set to < 1 to use total number of predictions as this. If NULL then block size is 1000 if new data supplied, and the number of rows in the model frame otherwise.

ewdata.guaranteed

Set to TRUE to turn off all checking of newdata: this can speed things up for large prediction tasks, but newdata must be complete, with no NA values for predictors required in the model.

na.action

what to do about NA values in newdata. With the default na.pass, any row of newdata containing NA values for required predictors, gives rise to NA predictions (even if the term concerned has no NA predictors). na.exclude or na.omit result in the dropping of newdata rows, if they contain any NA values for required predictors. If newdata is missing then NA handling is determined from object$na.action.

unconditional

if TRUE then the smoothing parameter uncertainty corrected covariance matrix is used to compute uncertainty bands, if available. Otherwise the bands treat the smoothing parameters as fixed.

newB

the given matrix of bivariate spline basis functions.

newind00

the given index of the data points in the triangulation.

backfitting

whether SBL estimation is obtained.

... other arguments.

Details

See examples to see usages of different types.

Value

if se.fit is TRUE then a 2 item list is returned with items (both arrays) fit and se.fit containing predictions and associated standard error estimates, otherwise an array of predictions is returned. The dimensions of the returned arrays depends on whether type is "terms" or not: if it is then the array is 2 dimensional with each term in the linear predictor separate, otherwise the array is 1 dimensional and contains the linear predictor/predicted values (or corresponding s.e.s). The linear predictor returned termwise will not include the offset or the intercept.

---

print.plbpsm

Print a Bivariate Penalized Spline based on Triangulation object.

Description

The default print method for a plbpsm object.

Usage

## S3 method for class 'plbpsm'
print(x, ...)
Arguments

x, ...  
  fitted model objects of class plbpsm as produced by plbpsm().

Details

Prints out the family, model formula, and etc. (need to be decided) See plbpsmObject (or names(x)) for a listing of what the object contains. summary.plbpsm provides more detail. Note that the optimized smoothing penalty parameter selection criterion reported is one of GCV, CV.

summary.plbpsm  
  Summary for a PLBPSM fit

Description

Takes a fitted plbpsm object produced by plbpsm() and produces various useful summaries from it.

Usage

## S3 method for class 'plbpsm'
summary(object, h_opt = NULL, X0 = NULL, 
dispersion = NULL, ...)

## S3 method for class 'summary.plbpsm'
print(x, digits = max(3, getOption("digits") - 
  3), signif.stars = getOption("show.signif.stars"), ...)

Arguments

object  
  a fitted plbpsm object as produced by plbpsm().

h_opt  
  the bandwidth given for Spline-backfitting local estimator, default is NULL.

X0  
  the new predict matrix for obtaining simultaneous confidence band.

dispersion  
  a value for the dispersion parameter: not normally used.

...  
  other arguments.

x  
  a summary.plbpsm object produced by summary.plbpsm().

digits  
  controls number of digits printed in output.

signif.stars  
  Should significance stars be printed alongside output.

Value

summary.plbpsm produces a list of summary information for a fitted plbpsm object.

p.coeff  
  is an array of estimates of the strictly parametric model coefficients.

p.t  
  is an array of the p.coeff’s divided by their standard errors.

p.pv  
  is an array of p-values for the null hypothesis that the corresponding parameter is zero. Calculated with reference to the t distribution with the estimated residual degrees of freedom for the model fit if the dispersion parameter has been estimated, and the standard normal if not.
The number of smooth terms in the model.

`se` array of standard error estimates for all parameter estimates.

`r.sq` The adjusted r-squared for the model. Defined as the proportion of variance explained, where original variance and residual variance are both estimated using unbiased estimators. This quantity can be negative if your model is worse than a one parameter constant model, and can be higher for the smaller of two nested models! The proportion null deviance explained is probably more appropriate for non-normal errors. Note that `r.sq` does not include any offset in the one parameter model.

`dev.expl` The proportion of the null deviance explained by the model. The null deviance is computed taking account of any offset, so `dev.expl` can be substantially lower than `r.sq` when an offset is present.

`edf` array of estimated degrees of freedom for the model terms.

`residual.df` estimated residual degrees of freedom.

`n` number of data.

`np` number of model coefficients (regression coefficients, not smoothing parameters or other parameters of likelihood).

`criterion` The criterion to choose the penalty parameter lambda. "GCV" to use generalized cross validation method and "CV" for cross validation

`family` The family object, specifying the distribution and link to use.

`method` 'ALASSO' or 'SCAD' to penalize the coefficients for parametric part.

`formula` the original PLBPSM formula.

`dispersion` the scale parameter.

`pTerms.df` the degrees of freedom associated with each parametric term (excluding the constant).

`pTerms.chi.sq` a Wald statistic for testing the null hypothesis that the each parametric term is zero.

`pTerms.pv` p-values associated with the tests that each term is zero. For penalized fits these are approximate. The reference distribution is an appropriate chi-squared when the scale parameter is known, and is based on an F when it is not.

`cov.scaled` The estimated covariance matrix of the parameters.

`p.table` significance table for parameters

`p.Terms` significance table for parametric model terms

`gcv_opt` The optimized gcv score.

`cv_opt` The optimized cv score.

`bands` A list of confidence bands for univaratie functions estimates.

`mhat` The estimated values for each linear or nonlinear term.
Defining smooths in PLBPSM formulae

Description

Function used in definition of smooth terms within plbpsm model formulae. The function does not evaluate a (spline) smooth - it exists purely to help set up a model using spline based smooths.

Usage

```r
u(..., N = 2, q = 3, KnotsLocation = "quantile", knots = NULL,
    N_MI = 4, fx = FALSE, id = NULL)
```

Arguments

- `...`: a list of variables that are the covariates that this smooth is a function of.
- `N`: Number of interior knots in generating spline matrix.
- `q`: Degree of polynomial spline. Default is 3.
- `KnotsLocation`: A character string naming the way for knots locations. Default is "quantile". The only alternative is "uniform".
- `knots`: An optional vector specifying the knots to be used in constructing spline bases.
- `N_MI`: Number of interior knots in generating spline matrix in the model identification process.
- `fx`: indicates whether the term is a fixed d.f. regression spline (TRUE) or a penalized regression spline (FALSE).
- `id`: An identifying label or number for the smooth, linking it to other smooths. Defaults to NULL for no linkage.

Value

These smooth.spec objects define univariate smooths and are turned into bases and penalties by BasisCon functions. The returned object contains the following items:

- `term`: An array of text strings giving the names of the covariates that the term is a function of.
- `N`: Number of interior knots in generating spline matrix.
- `q`: Degree of polynomial spline. Default is 3.
- `knotsLocation`: A character string naming the way for knots locations. Default is "quantile". The only alternative is "uniform".
- `knots`: An optional vector specifying the knots to be used in constructing spline bases.
- `N_MI`: Number of interior knots in generating spline matrix in the model identification process.
- `fx`: TRUE if the term is to be treated as a pure regression spline (with fixed degrees of freedom); FALSE if it is to be treated as a penalized regression spline.
- `dim`: The dimension of the smoother - i.e. the number of covariates that it is a function of.
- `label`: A suitable text label for this smooth term.
- `id`: An identifying label or number for the smooth, linking it to other smooths. Defaults to NULL for no linkage.
Index

b, 2, 8
BasisCon, 3
BasisCon.bivariate.smooth, 4
BasisCon.univariate.smooth, 5
beta_se, 5
cv.plbpm, 6
Data_Generator, 7
family, 12
formula.plbpm, 7
ggrpllsfit, 8
glm, 8, 12
gpllsfitGCV, 9, 10, 16
gpllsfitGCV_nb, 10
grpllsfit, 11
hello, 12
plbpm, 7, 8, 12
plbpm.control, 8–10, 13, 16
plbpmObject, 7, 14, 21
plot_plbpm, 16
plssfitCV, 11, 17
plssfitGCV, 11, 18
predict.plbpm, 19
print.plbpm, 20
print.summary.plbpm(summary.plbpm), 21
summary.plbpm, 21, 21
u, 8, 23