Package 'pblm'

July 23, 2025

Type Package

Title Bivariate Additive Marginal Regression for Categorical Responses

Version 0.1-12

Date 2025-06-18

Author Marco Enea [aut, cre, cph],
Mikis Stasinopoulos [ctb],
Robert Rigby [ctb]

Maintainer Marco Enea <marco.enea@unipa.it>

Depends R (>= 4.4.0), Matrix, lattice, splines, MASS

Imports methods

Description Bivariate additive categorical regression via penalized maximum likelihood. Under a multinomial framework, the method fits bivariate models where both responses are nominal, ordinal, or a mix of the two. Partial proportional odds models are supported, with flexible (non-)uniform association structures. Various logit types and parametrizations can be specified for both marginals and the association, including Dale's model. The association structure can be regularized using polynomial-type penalty terms. Additive effects are modeled using P-splines. Standard methods such as summary(), residuals(), and predict() are available.

License GPL (>= 2)

LazyData TRUE

Encoding UTF-8

NeedsCompilation no

URL https://github.com/MarcoEnea/pblm

BugReports https://github.com/MarcoEnea/pblm/issues

Repository CRAN

Date/Publication 2025-06-19 15:20:06 UTC

Contents

`pblm-package` *Bivariate Additive Marginal Regression for Categorical Responses*

Description

Fitting bivariate additive marginal regression models for categorical responses via penalized maximum likelihood estimation.

Details

Package:	pblm
Type:	Package
Version:	0.1-12
Date:	2025-06-18
License:	GPL (>= 2)

It is possible to fit partial proportional odds models and to specify several parametrizations for the marginals and the association, including the Dale's model. P-splines can be used to smooth covariates. The association structure can also be smoothed by polynomial surfaces by using penalty terms.

Author(s)

Marco Enea, with contributions by Mikis Stasinopoulos and Robert Rigby

Maintainer: Marco Enea <marco.enea@unipa.it>

bms *A British male sample on occupational status.*

Description

These data were analyzed by Goodman (1979) and concern the cross-classification of a sample of fathers and their sons according to the occupational status.

Usage

```
data(bms)
```

Format

A data frame with 49 observations and 3 variables.

`fathers` fathers'occupational status. A factor with levels from 1 to 7.

`sons` sons'occupational status. A factor with levels from 1 to 7.

`freq` a vector of integers representing the number of people cross-classified according to the occupational status of `fathers` and `sons`

Source

Goodman, L. A. (1979). Simple models for the analysis of cross-classications having ordered categories. *Journal of the American Statistical Association*, **74**(367), 537-552.

Examples

```
data(bms)
xtabs(freq ~ fathers + sons, data=bms)
```

multicolumn *transforming bivariate data in a multi-column format*

Description

This function transforms a grouped two-column response data frame into a multi-column one, another data format accepted by `pblm`

Usage

```
multicolumn(formula,data)
```

Arguments

formula	a two-side formula with counts in the left side.
data	a data frame with two categorical responses, covariates (if any) and a count variable.

Value

A data frame with as many responses as the number cells from the underlying response table and covariates (if any).

Author(s)

Marco Enea

Examples

```
#NOT RUN
data(ulcer)
multicolumn(freq~medication+pain+operation,data=ulcer)
```

---

pb *Specify a Penalised B-Spline Fit in a pblm Formula*

Description

Both pb and pbs are adaptations of function pb and ps from the gamlss package, respectively, to specify penalized B-spline.

Usage

```
pb(x, df = NULL, lambda = NULL, control = pb.control(...), ...)
pb.control(inter = 20, degree = 3, order = 2, quantiles = FALSE, ...)
pbs(x, df = 3, lambda = NULL, ps.intervals = 20, degree = 3, order = 3)
```

Arguments

x	the univariate predictor.
df	the desidered equivalent number of degrees of freedom (trace of the smoother matrix minus two for the constant and linear fit.
lambda	the smoothing parameter.
control	setting the control parameters
ps.intervals	the number of break points in the x-axis.
inter	the number of break points (knots) in the x-axis.
degree	the degree of the piecewise polynomials.
order	the required difference in the vector of coefficients.
quantiles	if TRUE the quantile values of x are used to determine the knots.
...	for extra arguments.

Details

Basically, pb is a reduced-functionality version of the original one specified in gamlss with no performance iteration methods (i.e. by method specification) implemented. The only method implemented minimizes the GAIC by internal optim calls.

Value

The function returns the vector x, which includes several attached attributes. While x is directly used in building the model matrix, its attributes are crucial for the backfitting procedure implemented by additive.fit()

Author(s)

Marco Enea, based on the original versions of the corresponding functions contained in the gamlss package by Mikis Stasinopoulos and Bob Rigby.

Examples

```
#NOT RUN
#Example 1. The Dale's model
data(ulcer)
m1 <- pblm(fo1=cbind(pain,medication)~1, fo12=~I(operation=="vh"), RC.fo=~Col,
           data=ulcer, weights=freq,  contrasts=list(Col="contr.SAS"))
summary(m1)

# Example 2. An artificial data set:
set.seed(1234)
da <- expand.grid("Y1"=1:3,"Y2"=1:3,"fat1"=0:9,"fat2"=0:1)
da$x1 <- seq(-5,5,l=180)
da$x2 <- rnorm(180)

da$Freq <- sample(5:30,180,replace=TRUE)

m1 <- pblm(fo1=cbind(Y1,Y2) ~ pbs(x1) + fat2,
           fo2=~pb(x1) + x2,
           fo12=~pb(x1) + x2, data=da, weights=Freq)

plot(m1)
```

---

pblm *Bivariate Additive Regression for Categorical Responses*

Description

This function allows to fit bivariate additive marginal logistic regression models for nominal, ordinal or mixed nominal/ordinal responses.

Usage

```
pblm(fo1=NULL, fo2=NULL, fo12=NULL, RC.fo=NULL, data, weights=NULL,
     type="gg", verbose=FALSE, contrasts=NULL, start=NULL, x=FALSE,
     center=FALSE, scale=FALSE, plackett=NULL, ncat1=NULL, ncat2=NULL,
     fit=TRUE, proportional=pblm.prop(...), penalty=pblm.penalty(...),
     control=pblm.control(...), ...)
```

Arguments

fo1	a two hand-side formula for the logit(s) of the first response. Depending on the data structure, the left-hand side can be a two-column or a multi-column response vector. In the latter case, argument ncat1 must be specified. See Examples.
fo2	a one hand-side formula for the logit(s) of the second response. If omitted, it will be assumed to be equal to fo1.
fo12	a one hand-side formula for the log-odds ratio(s) of the association between the two responses. If omitted, it will be assumed to be equal to fo1.
RC.fo	a Row/Column type formula specifying the association structure. See Details.
data	a data frame.
weights	An optional vector containing the observed frequencies.
type	a two-length string specifying the type of logits to use in the model fit. See Details.
verbose	logical. Should information about convergence be printed during model estimation?
contrasts	the Row/Column contrasts to be used in RC.fo. See argument contrasts.arg in model.matrix.default.
x	logical. Should the model matrix used in the fitting process be returned as component of the fitted model?
center	logical. Should the covariates be centered with respect their mean before the fit?
scale	logical. Should the covariates be scaled with respect their standard deviation before the fit?
start	an optional vector of starting values for the coefficients of the non-additive part.
plackett	logical. Should a Plackett-based formula be used for the inversion η->π. Actually, this is allowed (and it is the default) for binary or ordered responses only. The more general method uses the Newton-Raphson inversion algorithm described in Glonek and McCullagh (1995).
ncat1	an integer indicating the number of levels of the first response. Mandatory the data is in multicolumn format. See example below.
ncat2	an integer indicating the number of levels of the second response.
fit	logical. If TRUE (default), the model will be estimated, otherwise only some objects created prior to estimation will be returned.
control	this sets the control parameters for the Fisher-scoring and the inner Newton-Raphson and backfitting algorithms. The default setting is specified by the pblm.control function.

`proportional`	this sets a list of logical vectors specifying which explanatory variables depend on the categories of the responses. The default setting is specified by the `pblm.prop` function
`penalty`	this sets the penalty terms and smoothing parameters for a non-parametric "vertical" smoothing across the categories of the responses. The default setting is specified by the `pblm.penalty` function.
`...`	further arguments.

Details

It is possible to fit partial proportional odds models and specify several association structures like the Goodman's (1979) model (interactions are allowed though these are of linear type), the Dale's (1986) model and their additive version (Bustami *et al.*, 2001), Enea *et al.*(2014). Furthermore, the association structure can also be smoothed by using penalty terms of polynomial type as those considered in Enea and Attanasio (2015). That allows to enlarge the range of possible parametrizations of the association structure as an alternative to the Dale's Row-Column parametrization. Further details on the penalty terms specified through `pblm.penalty` can also found in Enea and Lovison (2014).

The algorithm is based on the bivariate version of the model by Glonek and McCullagh (1995), that is by using the *multivariate logit transform*

$$\boldsymbol{C}' \log(\boldsymbol{M}\boldsymbol{\pi}) = \boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta}.$$

Once `fo1` has been specified, if `fo2` and `fo12` are left unspecified, these are assumed to be equal to `fo1`. By default, the function fits a proportional odds model for ordered responses, in which only the marginal and the association intercepts are assumed to be category-dependent (Glonek and McCullagh, 1995).

Model formulae using a Row-Column type parametrization, like in the Goodman or the Dale models, need to be specified by using `RC.fo`. The right-hand side of such formula only recognizes `Row` and `Col` to specify row and/or column effects. Covariates must be specified separately by using `fo12`. See Examples.
The logits implemented are *local*; *global*; *continuation*; *reverse continuation*; (Colombi and Forcina, 2001) and *basic*. By using argument `type`, several log-odds ratios can be specified, among the permutations of the local-local type (`type="ll"`), local-global (`"lg"`),..., basic-basic (`type="bb"`). Furthermore, if the responses are binary, setting `type="ss"` will correspond to classical logit $\pi = \log P(Y = 1)/P(Y = 0)$ for both responses, while other specifications will produce $\log P(Y = 0)/P(Y = 1)$.

The vector of the starting values must be set in the following order: itercepts of the first logit; covariates of the first logit; intercepts of the second logit; covariates of the second logit; association intercepts; association covariates.

For what concerning the additive part, p-slines can be fitted by using `pb` and/or `pbs`, which are adaptations (reduced versions) of `pb` and `pbs` from the `gamlss` package, respectively.

Value

A list of components from the model fit. Some of these could be redundant and removed in a future version of the package. When `fit=TRUE` the returned components are:

`coef`	a named column vectors of coefficients.
`n`	the total number of observations.
`m`	the number of observed configurations of the responses given covariates (if any). It corresponds to the number of rows of the dataset.
`p`	fitted probability matrix given the observations.
`Y`	the weighted matrix of the responses in multinomial format.
`x`	if requested, the model matrix.
`xx1`, `xx2`, `xx12`	vectors, for internal use.
`ynames`	vector with the names of the responses.
`tol`	the accuracy reached at the convergence.
`llp`	the penalized log-likelihood value at the convergence.
`ll`	the log-likelihood value at the convergence.
`devp`	the penalized deviance value at the convergence.
`dev`	the deviance value at the convergence.
`IM`	the estimated Information Matrix at the convergence.
`IMp`	the estimated penalized Information Matrix at the convergence. Its inverse is used to calculate the standard error of estimates.
`convergence`	logical indicating whether convergence criteria were satisfied.
`iter`	the number of iterations performed in the model fitting.
`maxB`	the number of smoothers present in the equation with the maximum number of smoothers.This also represents the number of outer iterations of the backfitting algorithm.
`ncat1`, `ncat2`	the number of levels for the two responses.
`ncat`	this is simply `ncat1*ncat2`.
`weights`	the prior weights, that is the weights initially supplied, a vector of 1s if none were.
`P`	a $p \times p$ penalty matrix, where p is the number of parameters estimated (including all the intercepts), concerning the penalty term specified (if any), but excluding the additive part (if any). If no penalty terms are specified, this will be a matrix of zeros.
`gaic.m`	the *penalty* per parameter of the generalized AIC initially provided. It is 2 by default.
`lam1`, `lam2`, `\cr lam3`, `lam4`	vectors of smoothing parameters initially supplied with the specification of a penalty term. If not, these are `NULL`.
`opt`	the object returned by `optim` if automatic selection of smoothing parameters has been set. `NULL` otherwise.

etasmooth	the matrix of predicted values for the additive part, NULL otherwise.
eta	the n×(ncat-1) matrix of predicted values on the observed data.
fsmooth	a list of objects initially created by the smoothers (if any), NULL otherwise.
one.smooth	for internal use.
df.fix	the degrees of freedom of the parametric part of the model. Note that if a penalty term is used by specifying penalty, the corresponding degrees of freedoms will be counted in the parametric part of the model.
df.fix.vect	for internal use.
df.smooth	the degrees of freedom of the additive part of the model (if any), otherwise zero.
w.res	a n×(ncat-1) matrix of working residuals.
W2	a m-length list of matrices of working weights.
z	a (m*ncat-1)-length working vector of responses.
any.smoother	logical indicating whether smoothers have been used in the model fit.
Bmat	list of bases of smoothers used in the model fit (if any), NULL otherwise.
wh.eq	list of logical vectors indicating which terms in the linear predictor are smoothers.
wh.eq2	list of logical vectors indicating which terms in the linear predictor are smoothers. For internal use.
PBWB	list of numerical matrices if smoothers are used, NULL otherwise. For internal use.
BWB	list of numerical matrices if smoothers are used, NULL otherwise. For internal use.
etalist	list of numerical vectors if smoothers are used. For internal use.
spec.name	list of numerical vectors if smoothers are used. For internal use.
beta.smooth	list of estimated coefficients for the smoothers (if any), NULL otherwise.
n.smoothers	the number of smoothers used.
PPmat	list of penalty matrices for the smoothers multiplied by the smoothing parameters. NULL if smoothers are not used.
pnl.type	the type of penalty used.
xnames	list of vectors with the names of the covariates.
prop.smooth	for internal use.
GAIC	the value of the generalized AIC from the fitted model for the specified value of gaic.m in pblm.control.
ta	the underlying observed contingency table of the responses, marginally to the covariates.
set0	the parameter setting from pblm.control as specified by the user.
fo.list	list of formulas used.
center	logical indicating whether argument if the covariates were centered
scale	logical indicating whether argument if the covariates were scaled
type	the type of logit used.

plackett	logical indicating whether the plackett inversion formula was used.
RC.fo	the Row-Column formula as specified by the user.
contrasts	the Row-Column formula contrasts as specified by the user.
acc2	tolerance to be used for the estimation as specified by the user.
maxit	maximum number of Fisher-scoring iterations as specified by the user.
label1, label2	the names of the two response variables, respectively.
call	the matched call from object.

Warning

Please be sure that the results you get are really what you are "expecting" when using uncommon specifications of argument type, mainly those involving an s logit type and its combinations with other logits. A part from the binary case, many of those have been not checked yet in the current version of the package.

The estimation of category-dependent p-splines, as outlined in Enea et al. (2014), is not allowed (work in progress).

Note

Please note that specifying a formula with interaction terms in RC.fo corresponds to a model with association structure of the type $\alpha + \beta_r + \gamma_c + \delta_{rc}$. In the current version of the package, non linear interaction terms of the type $\alpha + \beta_r + \gamma_c + \delta_{1r}\delta_{2c}$, as considered for example in Lapp et al. (1998), are not implemented here.
Furthermore, unlikely from the Dale's paramaterization, pblm does not put a minus sign to covariates.

Author(s)

Marco Enea <marco.enea@unipa.it> with contribution by Mikis Stasinopoulos and Bob Rigby

References

Bustami, R., Lesaffre, E., Molenberghs, G., Loos, R., Danckaerts, M. and Vlietinck, R. 2001. Modelling bivariate ordinal responses smoothly with examples from ophtalmology and genetics. *Statistics in Medicine*, **20**, 1825-1842.

Colombi, R. and Forcina, A. 2001. Marginal regression models for the analysis of positive association of ordinal response variables. *Biometrika*, **88**(4), 1007-1019.

Dale, J. R. (1986) Global Cross-Ratio Models for Bivariate, Discrete, Ordered Responses. *Biometrics*, **42**(4), 909-917.

Enea, M. and Attanasio, M. 2015. A model for bivariate data with application to the analysis of university students success. *Journal of Applied Statistics*, http://dx.doi.org/10.1080/02664763.2014.998407

Enea, M. and Lovison, G. 2019. A penalized approach for the bivariate logistic regression model with applications to social and medical data. *Statistical Modelling*, **19(5)**, 467-500.

Enea, M., Stasinopoulos, M., Rigby, R., and Plaia, A. 2014. The pblm package: semiparametric regression for bivariate categorical responses in R. In Thomas Kneib, Fabian Sobotka, Jan Fahrenholz, Henriette Irmer (Eds.) *Proceeding of the 29th International Workshop of Statistical Modelling*, Volume 2, 47-50.

Glonek, G. F. V. and McCullagh, P. (1995) Multivariate logistic models. *Journal of the Royal Statistical Society, Series B*, **57**, 533-546.

Goodman, L. A. (1979). Simple models for the analysis of cross-classications having ordered categories. *Journal of the American Statistical Association*, **74**(367), 537-552.

Lapp, K., Molenberghs, G., and Lesaffre, E. (1998) Models for the association between ordinal variables. *Computational Statistics and Data Analysis*, **27**, 387-411.

See Also

summary.pblm

Examples

```
#NOT RUN
# an artificial example:
set.seed(123)
da <- expand.grid("Y1"=1:3,"Y2"=1:3,"fat1"=0:4,"fat2"=0:1)
da$Freq <- sample(0:20,3*3*5*2,replace=TRUE)
da$x1 <- rnorm(90)

#the bivariate additive proportional-odds model
m2 <- pblm(fo1=cbind(Y1,Y2) ~ fat1 + pb(x1), data=da, weights=Freq)
plot(m2)
```

---

pblm.control *Auxiliary for controlling the algorithm in a* pblm *model*

Description

This is an auxiliary function for controlling the algorithm in a pblm model.

Usage

```
pblm.control(maxit = 30, maxit2 = 200, acc = 1e-07, acc2 = 1e-06,
             zero.adj = 1e-06, l = NULL, restore.l = FALSE,
             min.step.l = 1e-04, auto.select = FALSE, gaic.m = 2,
             rss.tol = 1e-06, max.backfitting = 10, pgtol.df = 0.01,
             factr.df = 1e+07, lmm.df = 5, parscale.df = 1,
             max.gaic.iter = 500, pgtol.gaic = 1e-05, grad.tol = 1e-07,
             factr.gaic = 1e+07, lmm.gaic = 5, parscale = 1,
             conv.crit = c("dev", "pdev"))
```

Arguments

`maxit`	maximum number of Fisher-scoring iterations.
`maxit2`	maximum number of Newton-Raphson iterations for the inversion η->π.
`acc`	tolerance to be used for the estimation.
`acc2`	tolerance to be used for the inversion η->π.
`zero.adj`	adjustment factor for zeros in the probability vector π.
`l`	numerical, ranged in (0,1], representing the initial value of step lenght. By default `l=1`.
`restore.l`	logical, should the step length be restored to its initial value after each iteration? This is an experimental option and may be changed in the future.
`min.step.l`	numerical, minimum value fixed for the step length.
`auto.select`	logical, should the smoothing parameters be estimated by GAIC minimization? If `TRUE` The optimization will be performed numerically by using `optim`.
`gaic.m`	the "penalty" per parameter of the generalized AIC. By default it is 2, corresponding to the classical AIC.
`rss.tol`	tolerance for the residual sum of squares used in the backfitting algorithm.
`max.backfitting`	maximum number of backfitting iterations.
`pgtol.df`	tolerance to be used in order to get an amount of smoothing corresponding to the fixed degrees of freedom for the additive part. See argument `pgtol` from `optim`.
`factr.df`	numerical. For degrees-of-freedom optimization in the additive part. See argument `factr` from `optim`.
`lmm.df`	integer. For degrees-of-freedom optimization in the additive part. See argument `lmm` from `optim`.
`parscale.df`	A vector of scaling parameters for vector lambda when optimizing lambda for fixed degrees of freedom. See argument `parscale` from `optim`.
`max.gaic.iter`	integer. Maximum number of iterations for automatic model optimization. See argument `maxit` from `optim`.
`pgtol.gaic`	numerical. Tolerance to be used for automatic selection of smoothing parameters. See argument `pgtol` from `optim`.
`grad.tol`	numerical. Tolerance to be used when inverting the gradient matrix.

factr.gaic	numerical. For automatic selection of smoothing parameters. See argument factr from optim.
lmm.gaic	integer. For automatic selection of smoothing parameters. See argument lmm from optim.
parscale	A vector of scaling parameters for vector lambda for automatic model optimization. See argument parscale from optim.
conv.crit	Convergence criterion for model estimation. The default is "dev", corresponding to log-likelihood maximization. Alternatively, "pdev" is concerned with maximum penalized log-likelihood.

Value

A list with the same arguments of the function, unless unlikely specified by the user.

Author(s)

Marco Enea

See Also

pblm

---

pblm.penalty *Auxiliary for specifying penalty terms in a* pblm *model*

Description

This is an auxiliary function for specifying penalty terms in a pblm model.

Usage

```
pblm.penalty(pnl.type=c("none","ARC1","ARC2","ridge","lasso","lassoV", "equal"),
             lam1=NULL, lam2=NULL, lam3=NULL, lam4=NULL,
             s1=NULL, s2=NULL, s3=NULL, s4=NULL, min.lam.fix=0.1,
             constraints=FALSE, lamv1=1e10, lamv2=1e10)
```

Arguments

pnl.type	The type of penalty term to be used. By default "none" of them is used. See Details.
lam1, lam2	vectors of smoothing parameters for the marginals. By default they are zero. It is common to all the penalty terms implemented.
lam3	vector of smoothing parameters for the association. It is common to all the penalty terms implemented. If "ARC2" is selected, it will act on differences of by-row adjacent parameters.

lam4	vector smoothing parameters for the association. Specific for the "ARC2" penalty, for which the differences of by-column adjacent parameters are penalized
s1, s2, s3, s4	the orders of the difference operator. Specific for the "ARC2" penalty.
min.lam.fix	the minimum value for any penalty parameters in order to consider any lamdas smaller than this value as fixed. See details.
constraints	Should inequality constraints be applied to the marginal probabilities? This is done through a penalty term and intended for ordered resposes.
lamv1, lamv2	penalty parameters to be applied to both the marginal probabilities in order to mimicking inequality constraints.

Details

Some penalty terms implemented in pblm are described in Enea and Lovison (2014) and Enea and Attanasio (2015).

Just one penalty per model is allowed.

Penalty "ARC1" acts on first order differences of category-adjacent parameters. By increasing the smoothing parameters, the resulting marginal and/or association parameters, will tend to be equal among the categories. When the underlying contingency table cross-classifying the responses contains zero cells, This penalty may be useful to stabilize the estimates, for example to get a more "regular" association structure.

Penalty "ARC2" generalizes in a certain sense "ARC1", since it acts on high order differences of Ajacent Row and/or Column parameters, but it is maily used for ordered responses. For high smoothing values it constraints the marginal parameters to lie onto a polynomial curve, and/or the association structure to lie onto a polynomial surface. The degrees of the marginal polynomials are determined by s1-1, for the first marginal, and s2-1 for the second. The degree of the association polynomial surface is determined by s3+s4-2, in which s3-1 is the by-row polynomial degree and s4-1 the by-column one.

Penalty "ridge" constraints the regression parameters towards zero (horizontal penalty), so providing equal estimates for high penalty values.

The current implementation of the "lasso" and "lassoV" penalty terms is to be considerer provisional and needs to be better checked.

Penalty "lasso" is acts similarly to "ridge" and it is based on absolute values of the regression parameters.

Penalty "lassoV" vertically penalizes the absolute value of differences of adjacent row and column parameters. It is similar to ARC1. By increasing the smoothing parameter, the resulting marginal and/or association regression parameters, will tend to be equal among the response categories. This

Penalty "equal" constraints the marginal equations to be equal, so providing equal estimates. This could be useful, for example, in eyes or twins studies. The tuning parameter to be specified for this penalty is `lam1`.

Furthermore, if global logits are specified, inequality constraints on marginal predictors are mimicked by using penalty terms. Argument `lamv1` is the penalty parameter for the marginal predictor of the first response, `lamv2` is that for the second one.

Argument `min.lam.fix` is useful when an automatic selection of penalty parameters is desidered for certain lambdas only. The remaining lambdas can be either 0 or less than `min.lam.fix`, and excluded from the automatic selection. Fixing some lambdas to assume values in $[0$,`min.lam.fix`$)$ may be useful, for example, for parameter space regularization.

Value

A list with the same arguments of the function, unless unlikely specified by the user.

Author(s)

Marco Enea <marco.enea@unipa.it>

References

Enea, M. and Attanasio, M. 2015. A model for bivariate data with application to the analysis of university students success. Journal of Applied Statistics, http://dx.doi.org/10.1080/02664763.2014.998407

Enea, M. and Lovison, G. 2019. A penalized approach for the bivariate logistic regression model with applications to social and medical data. *Statistical Modelling*, **19(5)**, 467-500.

See Also

`pblm`

Examples

```
#Example 1
# A British male sample on occupational status.
data(bms)

# A third degree polynomial surface with equally-spaced integer scores
m1 <- pblm(fo1=cbind(fathers,sons)~1, data=bms, weights=bms$freq,
             penalty=pblm.penalty(pnl.type="ARC2",lam3=c(1e7), lam4=c(1e7),
                                  s3=c(4), s4=c(4)))

require(lattice)
g <- expand.grid("sons"=1:6,"fathers"=1:6)
g$logGOR <- m1$coef[13:48]

oldpar <- par(no.readonly = TRUE)
```

```
wireframe(logGOR ~ sons*fathers, data = g, zlim=c(min(g$logGOR-1),max(g$logGOR+1)),
          scales = list(arrows = FALSE), screen = list(z = -130, x = -70),
          col.regions="magenta")
par(oldpar)


#Example 2

# an artificial data frame with two binary responses and two factors
set.seed(12)
da <- expand.grid("Y1"=0:1,"Y2"=0:1,"fat1"=0:1,"fat2"=0:1)
da$Freq <- sample(0:20,2*2*2*2,replace=TRUE)

# A quasi-independence model obtained by strongly penalizing the association intercept
# through a ridge-type penalty term
m3 <- pblm(fo1=cbind(Y1,Y2) ~ fat1 + fat2,
           fo12=~ 1,
           data=da, weights=da$Freq, type="ss",
           proportional=pblm.prop(prop12=c(TRUE)),
           penalty=pblm.penalty(pnl.type="ridge",lam3=1e12))
summary(m3)

# notice that the last coefficient is not exactly zero
coef(m3)

m3.1 <- glm(Y1 ~ fat1 + fat2, data=da, weights=Freq, family=binomial)
m3.2 <- glm(Y2 ~ fat1 + fat2, data=da, weights=Freq, family=binomial)

all.equal(logLik(m3), logLik(m3.1)[1]+logLik(m3.2)[1])
```

`pblm.prop` *Auxiliary for specyfing category-dependent covariates in a* `pblm` *model*

Description

This is an auxiliary function which allows to specify partially proportional odds for one (or both) the marginals and with the association parameters which can depend (or not) on the categories of the responses. It simply returns a list with its arguments.

Usage

```
pblm.prop(prop1=NULL, prop2=NULL, prop12=NULL)
```

Arguments

prop1	a TRUE/FALSE logical vector specifying which explanatory variables are category-dependent, including the intercepts. Each element of this vector must exactly match the order of the covariates appearing in fo1, leaving out any additive term. If a factor is present, the corresponding elements in this vector will have as many elements as the number of levels of the factor minus 1.
prop2	a logical vector like prop1, but now it refers to the covariates present in fo2.
prop12	a logical vector like prop1, but now it refers to the covariates present in fo12

Details

The default specification will result in a model with category-dependent intercepts for both the marginal and the association, while all the covariates are assumed independent of the categories. Note that, for ordered responses, setting category-independent intercepts for the marginals is not a good idea.

Value

A list with the same arguments of the function, unless unlikely specified by the user.

Author(s)

Marco Enea <marco.enea@unipa.it>

Examples

```
# an artificial data frame with two five-category responses and two factors
set.seed(10)
da <- expand.grid("Y1"=1:5,"Y2"=1:5,"fat1"=letters[1:3],"fat2"=letters[1:3])
da$Freq <- sample(1:20,5*5*3*3,replace=TRUE)

#A partial proportional-odds model with uniform association
m2 <- pblm(fo1=cbind(Y1,Y2) ~ fat1 + fat2,
           fo2=~fat1,
           fo12=~1,
           data=da, weights=da$Freq,
           proportional=pblm.prop(prop1=c(FALSE,TRUE,TRUE,FALSE,FALSE),
           prop2=c(FALSE,TRUE,TRUE),
           prop12=c(TRUE)))
summary(m2)
```

---

plot.pblm *Plotting terms for a* pblm *object*

Description

Plotting fixed or smooth terms for a pblm object

Usage

```
## S3 method for class 'pblm'
plot(x, which.eq=1:3, which.var=1:x$maxNpred, add.bands=TRUE,
                    type="l",col.line=list("blue"), col.bands=NULL,
                    dashed.bands=FALSE, pause = FALSE, ylim, xlim, ylab, xlab,
                    main, overlaid_pvc=TRUE,...)
```

Arguments

x	An object of class pblm.
which.var	Index of the smoother term, indicating its position in the model formula.
which.eq	Equation index identifying the component (marginal or association) where the smoother is applied.
add.bands	Logical. Should confidence bands for the smoother be added to the plot?
col.bands	Color to be used for the confidence bands.
col.line	Color to be used for the smoother line. Different colors are allowed when using pvc().
type	Graphical parameter specifying the plot type.
dashed.bands	Logical. If TRUE (and add.bands is also TRUE), dashed confidence bands will be drawn instead of filled color bands.
pause	Logical. If TRUE, the user will be prompted to press a key before each plot is displayed.
ylim	Graphical parameter defining the limits of the y-axis.
xlim	Graphical parameter defining the limits of the x-axis.
ylab	Graphical parameter specifying the label for the y-axis.
xlab	Graphical parameter specifying the label for the x-axis.
main	Graphical parameter specifying the main title of the plot.
overlaid_pvc	Logical. Under development, currently ignored.
...	Further graphical parameters to be passed to the plotting functions.

Details

This function works similarly to the termplot function for many statistical models, and is based on the predict method. The argument overlaid_pvc is currently ignored because, although implemented, the smoother function pvc() for fitting penalized varying coefficient models is still experimental and not included in this version of the package.

Value

Returns the plots of the partial effects for the terms specified in the model formula, or for a specific term identified by which.equation and which.term.

In addition, a variable-length list is returned, containing one object for each term included in the model formula or selected via which.equation and which.term. Each object is named by concatenating the equation type and the term name, and consists of a data frame with as many rows as

the original dataset and four columns: the x-axis values, the y-axis values, the 95% lower bounds, and the 95% upper bounds.

For example, consider a variable named v included in the model for both marginals and the association. The returned list would include:

mar1:v	A data frame containing the plotting data for variable v in the first marginal.
mar2:v	A data frame containing the plotting data for variable v in the second marginal.
ass12:v	A data frame containing the plotting data for variable v in the association component.
...	Additional data frames, depending on the number of terms involved.

Author(s)

Marco Enea

See Also

pb, pbs

Examples

```
#NOT RUN
# an artificial data set:
set.seed(123)
da <- expand.grid("Y1"=1:3,"Y2"=1:3,"v1"=0:4,"fat2"=0:1)
da$Freq <- sample(0:20,3*3*5*2,replace=TRUE)
da$x1 <- rnorm(90)
#the bivariate additive proportional-odds model
m7 <- pblm(fo1=cbind(Y1,Y2) ~ v1 + fat2 + pb(x1), data=da, weights=Freq)
plot(m7)
```

---

summary.pblm *Summarizing methods for bivariate additive logistic regression*

Description

Summarizing methods anf functions for objects of class pblm.

Usage

```
## S3 method for class 'pblm'
summary(object,...)

## S3 method for class 'pblm'
print(x,digits = max(3, getOption("digits") - 3),...)

## S3 method for class 'summary.pblm'
```

```
print(x,digits = max(3, getOption("digits") - 3),...)

## S3 method for class 'pblm'
AIC(object,...,k=2)

## S3 method for class 'pblm'
logLik(object, penalized=FALSE,...)

## S3 method for class 'pblm'
vcov(object,...)

## S3 method for class 'pblm'
coef(object, digits = max(3, getOption("digits") - 3), ...)

## S3 method for class 'pblm'
coefficients(object, digits = max(3, getOption("digits") - 3), ...)

## S3 method for class 'pblm'
residuals(object, type = c("working", "pearson"),...)

## S3 method for class 'pblm'
resid(object, type = c("working", "pearson"),...)

## S3 method for class 'pblm'
fitted(object,...)

## S3 method for class 'pblm'
predict(object, newdata, type=c("link","response","terms","count"),
                  se.fit=FALSE, digits= max(6, getOption("digits") - 3),...)

## S3 method for class 'pblm'
deviance(object, penalized=FALSE,...)

edf.pblm(object, which.eq=1, which.var=1 )
chisq.test.pblm(obj)
se.smooth.pblm(object)
Rsq.pblm(object, type = c("McFadden", "Cox Snell", "Cragg Uhler", "all"))
```

Arguments

object, obj	An object of class pblm.
digits	Integer controlling the number of digits printed in the output.
x	An object produced by pblm or summary.pblm to be printed.
k	Numeric; the penalty per parameter to be used. By default k = 2, corresponding to the classical AIC.
penalized	Logical indicating whether the value of the penalized log-likelihood is required.
which.var	Index indicating the position of the smoother as it appears in the model formula.

which.eq	Equation index for the smoothers.
newdata	Optional data frame with new values of the model covariates.
type	The type of residuals, predictions, or pseudo R-squared desired.
se.fit	Logical. Should prediction standard errors be returned? Currently available only for type = "link" or type = "terms" when newdata is not specified, and only for type = "link" when newdata is specified.
...	Further arguments passed to or from other methods.

Details

fitted.pblm is equivalent to predict.pblm specifying type="response". chisq.test.pblm performs the χ^2 and G^2 tests. Rsq.pblm calculates pseudo R-squared.

Value

print.pblm return an object of class "pblm". print.summary.pblm return an object of class "summary.pblm". summary.pblm returns a list with the following components:

results	A $p \times 4$ data frame with columns for the estimated coefficient, its standard error, z-statistic, and corresponding (two-sided) p-value.
convergence	Logical flag indicating whether the algorithm converged.
iter	Number of iterations performed in the model fitting.
logLik	Log-likelihood value at convergence.
logLikp	Penalized log-likelihood of the fitted model.
AIC	Akaike Information Criterion of the fitted model.
gAIC	Generalized AIC of the fitted model.
BIC	Bayesian Information Criterion of the fitted model.
gaic.m	Penalty coefficient used in the gAIC.
np1, np2	Number of parameters estimated in the first and second marginal, respectively.
deviance	Model deviance.
names	A character vector of length two containing the names of the response variables.
df.res	Residual degrees of freedom.
df.tot	Total degrees of freedom of the model.
df.fix	Degrees of freedom associated with the fixed part of the model.
df.smooth	Effective degrees of freedom of the smoothers in the model, if any.
res.smooth	A data frame reporting an approximate chi-squared test for the smooth terms. If the effective degrees of freedom are far from the fixed ones, consider re-fitting the model by reducing pgtol.df in pblm.control, via the control argument.
smoother	Logical. TRUE if smooth terms were used in the model.
pnl.type	Penalty type as specified in the fitted object.
PR	A matrix with residuals summarized by row, one per model equation.
which.term	A list. For internal use.

respStr	A matrix. For internal use.
label1, label2	Names of the two response variables, respectively.
call	Matched call from the fitted object.

coef.pblm, coefficients.pblm return a numeric vector of estimated coefficient.

edf.pblm returns a scalar with the effective degrees of freedom for a specified smooth term.

vcov.pblm returns the variance-covariance matrix.

If se.fit = FALSE, the method predict.pblm returns a data frame with predicted values according to the specified prediction type. If se.fit = TRUE, a list of length two is returned with the following components:

fit	Data frame of predicted values for each linear predictor, according to the specified prediction type.
se.fit	Associated data frame of standard errors.

resid.pblm, residuals.pblm return a data frame with the specified type of residuals.

se.smooth.pblm returns a list of length three, containing the standard errors and the lower and upper bounds of the 95% confidence intervals for the smooth terms.

chisq.test.pblm returns a list of length two containing the results of the χ^2 and G^2 tests.

Rsq.pblm returns a scalar, or a list if type="both" is selected.

AIC.pblm returns a scalar with several attributes, with printed AIC and df. If more objects are passed as arguments, it returns a data frame.

logLik, deviance return a scalar.

Author(s)

Marco Enea

See Also

pblm, pb

Examples

```
#NOT RUN
## Example 1
#The Dale's model
data(ulcer)
m1 <- pblm(fo1=cbind(pain,medication)~1, fo12=~I(operation=="vh"), RC.fo=~Col,
           data=ulcer, weights=freq,  contrasts=list(Col="contr.SAS"))
summary(m1)
deviance(m1)
predict(m1,type="response")


#the same data but in another format
#compare with Dale (1986), Table 3
dat <- multicolumn(freq~medication+pain+operation,data=ulcer)
```

```
fo <- as.formula(paste(attributes(dat)$"resp","~1",sep=""))
m1bis <- pblm(fo1=fo, fo12=~I(operation=="vh"), RC.fo=~Col, verbose=TRUE,
              data=dat, ncat1=3, contrasts=list(Col="contr.SAS"))
deviance(m1bis)
chisq.test.pblm(m1bis)
Rsq.pblm(m1bis)

# Example 2. An artificial data set:
set.seed(10)
da <- expand.grid("Y1"=1:3,"Y2"=1:3,"fat1"=0:4,"fat2"=0:1)
da$Freq <- sample(1:20,3*3*5*2,replace=TRUE)
da$x1 <- rnorm(90)

#the bivariate additive proportional-odds model
m2 <- pblm(fo1=cbind(Y1,Y2) ~ fat1 + pb(x1), data=da, weights=Freq)
summary(m2)
```

---

ulcer *The ulcer data*

Description

Data analyzed by Dale (1986)

Usage

```
data(ulcer)
```

Format

A data frame with 48 observations and 4 variables.

`medication` medication requirements. A factor with levels `never`; `seldom`; `occasionally`; and `regularly`.

`pain` patients' post operative pain level. A factor with levels `none`, `slight` and `moderate`.

`operation` a factor representing the type of operation with levels: vagatomy drainage procedure (`vp`); vagatomy and distal antrectomy (`va`); vagatomy and hemigastrectomy (`vh`); and resection alone (`ra`).

`freq` a numeric vector representing the number of patients classified for the corresponding levels of `pain`, `medication` and `operation`

Source

Dale, J. R. (1986) Global Cross-Ratio Models for Bivariate, Discrete, Ordered Responses. *Biometrics*, **42**(4), 909-917.

Examples

```
data(ulcer)
```

Index