Package ‘bbreg’

October 12, 2022

Type Package

Title Bessel and Beta Regressions via Expectation-Maximization Algorithm for Continuous Bounded Data

Version 2.0.2

Maintainer Vinicius Mayrink <vdinizm@gmail.com>

Description Functions to fit, via Expectation-Maximization (EM) algorithm, the Bessel and Beta regressions to a data set with a bounded continuous response variable. The Bessel regression is a new and robust approach proposed in the literature. The EM version for the well known Beta regression is another major contribution of this package. See details in the references Barreto-Souza, Mayrink and Simas (2022) <doi:10.1111/anzs.12354> and Barreto-Souza, Mayrink and Simas (2020) <arXiv:2003.05157>.

Depends R (>= 3.5.0)

License GPL-2

Encoding UTF-8

LazyData true

Imports pbapply, Formula, expint, statmod

RoxygenNote 7.1.2

NeedsCompilation no

Suggests rmarkdown, knitr

VignetteBuilder knitr

Author Vinicius Mayrink [cre, aut] (<https://orcid.org/0000-0002-5683-8326>), Alexandre B. Simas [aut] (<https://orcid.org/0000-0003-2562-2829>), Wagner Barreto-Souza [aut] (<https://orcid.org/0000-0003-0831-7881>)

Repository CRAN

Date/Publication 2022-02-14 08:00:05 UTC

R topics documented:

bbreg *bbreg*

Description

Function to fit, via Expectation-Maximization (EM) algorithm, the bessel or the beta regression to a given data set with a bounded continuous response variable.

Usage

```
bbreg(
  formula,
  data,
  link.mean = c("logit", "probit", "cauchit", "cloglog"),
  link.precision = c("identity", "log", "sqrt", "inverse"),
  model = NULL,
  residual = NULL,
  envelope = 0,
  prob = 0.95,
  predict = 0,
  ptest = 0.25,
  epsilon = 10^(-5)
)
```

Arguments

`formula` symbolic description of the model (examples: `z ~ x1 + x2` and `z ~ x1 + x2 | v1 + v2`); see details below.

`data` elements expressed in formula. This is usually a data frame composed by: (i) the bounded continuous observations in `z` (0 < z_i < 1), (ii) covariates for the mean submodel (columns `x1` and `x2`) and (iii) covariates for the precision submodel (columns `v1` and `v2`).

`link.mean` optionally, a string containing the link function for the mean. If omitted, the ’logit’ link function will be used. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` optionally, a string containing the link function the precision parameter. If omitted and the only precision covariate is the intercept, the identity link function will be used, if omitted and there is a precision covariate other than the intercept, the ’log’ link function will be used. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

`model` character ("bessel" or "beta") indicating the type of model to be fitted. The default is NULL, meaning that a discrimination test must be applied to select the model.

`residual` character indicating the type of residual to be evaluated ("pearson", "score" or "quantile"). The default is "pearson".

`envelope` number of simulations (synthetic data sets) to build envelopes for residuals (with `100*prob%` confidence level). The default `envelope = 0` dismisses the envelope analysis.

`prob` probability indicating the confidence level for the envelopes (default: `prob =` 0.95). If `envelope` = 0, `prob` is ignored.

`predict` number of partitions (training set to fit the model and a test set to calculate residuals) to be evaluated in a predictive accuracy study related to the `RSS_pred` (residual sum of squares for the partition "test set"). The default `predict =` 0 dismisses the `RSS_pred` analysis. The partitions are randomly defined when predict is set as a positive integer.

`ptest` proportion of the sample size to be considered in the test set for the `RSS_pred` analysis (default = 0.25 = 25% of the sample size). If predict = 0, ptest is ignored.

`epsilon` tolerance value to control the convergence criterion in the EM-algorithm (default = 10^(-5)).

Details

The bessel regression originates from a class of normalized inverse-Gaussian (N-IG) process introduced in *Lijoi et al. (2005)* as an alternative to the widely used Dirichlet process in the Bayesian context. These authors consider a ratio of inverse-Gaussian random variables to define the new process. In the particular univariate case, the N-IG is obtained from the representation "Z = Y1/(Y1+Y2)", with "Y1" and "Y2" being independent inverse-Gaussian random variables having scale = 1 and shape parameters "a1 > 0" and "a2 > 0", respectively. Denote "Y1 ~ IG(a1)" and "Y2 ~ IG(a2)". The density of "Z" has support in the interval (0,1) and it depends on the modified Bessel function of third kind with order 1, named here as "*K1(-)*". The presence of "*K1(-)*" in the structure of the p.d.f. establishes the name of the new distribution; consider Z ~ Bessel(a1,a2). Note that the name "beta distribution" is also an analogy to the presence of a function (the beta function) in its p.d.f. structure. The bessel regression model is defined by assuming "Z_1,...,Z_n" as a random sample of continuous bounded responses with "Z_i ~ Bessel(mu_i,phi_i)" for "i = 1,...,n". Using this parameterization, one can write: "E(Z_i) = mu_i" and "Var(Z_i) = mu_i(1-mu_i) g(phi_i)", where "*g(-)*" is a function depending on the exponential integral of "phi_i". The following link functions are assumed "logit(mu_i) = x_i^T kappa" and "log(phi_i) = v_i^T lambda", where "kappa' = (kappa_1,...,kappa_p)" and "lambda' = (lambda_1,...,lambda[q])" are real valued vectors. The terms "x_i^T" and "v_i^T" represent, respectively, the i-th row of the matrices "x" (*nxp*) and "v" (*nxq*) containing covariates in their columns ("x_i,1" and "v_i,1" may be 1 to handle intercepts). As it can be seen, this regression model has two levels with covariates explaining the mean "mu_i" and the parameter "phi_i". For more details about the bessel regression see *Barreto-Souza, Mayrink and Simas (2022)* and *Barreto-Souza, Mayrink and Simas (2020).*

This package implements an Expectation Maximization (EM) algorithm to fit the bessel regression. The full EM approach proposed in *Barreto-Souza and Simas (2017)* for the beta regression is also available here. Fitting the beta regression via EM-algorithm is a major difference between the present package **bbreg** and the well known `betareg` created by Alexandre B. Simas and currently maintained by Achim Zeileis. The estimation procedure on the betareg packages is given by maximizing the beta model likelihood via `optim`. In terms of initial values, **bbreg** uses quasi-likelihood estimates as the starting points for the EM-algorithms. The formulation of the target model also has the same structure as in the standard functions `lm`, `glm` and **betareg**, with also the same structure as the latter when precision covariates are being used. The user is supposed to write a formula object describing elements of the regression (response, covariates for the mean submodel, covariates for the precision submodel, presence of intercepts, and interactions). As an example, the description "z ~ x" indicates: "response = z" (continuous and bounded by 0 and 1), "covariates = columns of x" (mean submodel) and precision submodel having only an intercept. On the other hand, the configuration "z ~ x | v" establishes that the covariates given in the columns of "v" must be used in the precision submodel. Intercepts may be removed by setting "z ~ 0 + x | 0 + v" or "z ~ x - 1|v - 1". Absence of intercept and covariates is not allowed in any submodel. The type of model to be fitted ("bessel" or "beta") can be specified through the argument "model" of **bbreg**. If the user does not specify the model, the package will automatically apply a discrimination test (DBB - Discrimination between Bessel and Beta), developed in *Barreto-Souza, Mayrink and Simas (2022)* and *Barreto-Souza, Mayrink and Simas (2020)*, to select the most appropriate model for the given data set. In this case, some quantities related to the DBB are included in the final output; they are: "sum(Z2/n)" = mean of z_i^2, "sum(quasi_mu)" = sum (for i = 1,...,n) of muq_i + muq_i(1-muq_i)/2, with muq_i being the quasi-likelihood estimator of mu_i and, finally, the quantities "|D_bessel|" and "|D_beta|" depending on muq_i and the EM-estimates of phi_i under bessel or beta.

In the current version, three types of residuals are available for analysis ("Pearson", "Score" and "Quantile"). The user may choose one of them via the argument "residual". The score residual is computed empirically, based on 100 artificial data sets generated from the fitted model. The sample size is the same of the original data and the simulations are used to estimate the mean and s.d. required in the score residual formulation. The user may also choose to build envelopes for the residuals with confidence level in "prob". This feature also requires simulations of synthetic data ("envelope" is the number of replications). Residuals are obtained for each data set and confronted against the quantiles of the N(0,1). Predictive accuracy of the fitted model is also explored by setting "predict" as a positive integer (this value represents the number of random partitions to be evaluated). In this case, the full data set is separated in a training (partition to fit the model) and a test set (to evaluate residuals) for which the RSS (Residual Sum of Squares) is computed. The default partition is 75% (training) and 25% (test); this can be modified by choosing the proportion `ptest` for the test set (large `ptest` is not recommended).

Value

Object of class **bbreg** containing the outputs from the model fit (bessel or beta regression).

References

DOI:10.1111/anzs.12354 (Barreto-Souza, Mayrink and Simas; 2022)

arXiv:2003.05157 (Barreto-Souza, Mayrink and Simas; 2020)

DOI:10.1080/00949655.2017.1350679 (Barreto-Souza and Simas; 2017)

DOI:10.18637/jss.v034.i02 (Cribari-Neto and Zeileis; 2010)

DOI:10.1198/016214505000000132 (Lijoi et al.; 2005)

See Also

`summary.bbreg`, `plot.bbreg`, `simdata_bes`, `dbessel`, `dbbtest`, `simdata_bet`, `Formula`

Examples

```
# Example with artificial data.
n = 100; x = cbind(rbinom(n, 1, 0.5), runif(n, -1, 1)); v = runif(n, -1, 1);
z = simdata_bes(kap = c(1, -1, 0.5), lam = c(0.5, -0.5), x, v,
repetition = 1, link.mean = "logit", link.precision = "log")
z = unlist(z)
fit1 = bbreg(z ~ x | v)
summary(fit1)
plot(fit1)

# Examples using the Weather Task (WT) data available in bbreg.

fit2 = bbreg(agreement ~ priming + eliciting, data = WT)
summary(fit2)

fit3 = bbreg(agreement ~ priming + eliciting, envelope = 30, predict = 10, data = WT)
summary(fit3)
# Example with precision covariates

fit4 = bbreg(agreement ~ priming + eliciting|eliciting, data = WT)
summary(fit4)
# Example with different link functions:

fit5 = bbreg(agreement ~ priming + eliciting|eliciting, data = WT,
link.mean = "cloglog", link.precision = "sqrt")
summary(fit5)
```

BF *Body Fat data set*

Description

Penrose body fat data set. Response variable is the percentage of body fat and covariates represent several physiologic measurements related to 252 men. All covariates were rescaled dividing their original value by 100.

Usage

```
data(BF)
```

Format

Data frame containing 252 observations on 14 variables.

bodyfat percentage of body fat obtained through underwater weighting.

age age in years/100.

weight weight in lbs/100.

height height in inches/100.

neck neck circumference in cm/100.

chest chest circumference in cm/100.

abdomen abdomen circumference in cm/100.

hip hip circumference in cm/100.

thigh thigh circumference in cm/100.

knee knee circumference in cm/100.

ankle ankle circumference in cm/100.

biceps biceps circumference in cm/100.

forearm forearm circumference in cm/100.

wrist wrist circumference in cm/100.

Source

Data is freely available from *Penrose et al. (1985)*. See also *Brimacombe (2016)* and *Barreto-Souza, Mayrink and Simas (2020)* for details.

References

arXiv:2003.05157 (Barreto-Souza, Mayrink and Simas; 2020)

DOI:10.1249/00005768-198504000-00037 (Penrose et al.; 1985)

DOI:10.4236/ojs.2016.61010 (Brimacombe; 2016)

Examples

```
data(BF)
```

`coef.bbreg` *coef.bbreg*

Description

Function to extract the coefficients of a fitted regression model (bessel or beta).

Usage

```
## S3 method for class 'bbreg'
coef(object, parameters = c("all", "mean", "precision"), ...)
```

Arguments

`object`	object of class "bbreg" containing results from the fitted model.
`parameters`	a string to determine which coefficients should be extracted: 'all' extracts all coefficients, 'mean' extracts the coefficients of the mean parameters and 'precision' extracts coefficients of the precision parameters.
`...`	further arguments passed to or from other methods.

See Also

fitted.bbreg, summary.bbreg, vcov.bbreg, plot.bbreg, predict.bbreg

Examples

```
fit = bbreg(agreement ~ priming + eliciting, data = WT)
coef(fit)
coef(fit, parameters = "precision")
```

d2mudeta2 *d2mudeta2*

Description

Function to obtain the second derivatives of the mean parameter with respect to the linear predictor.

Usage

```
d2mudeta2(link.mean, mu)
```

Arguments

link.mean	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
mu	mean parameter.

d2phideta2 *d2phideta2*

Description

Function to obtain the second derivatives of the precision parameter with respect to the linear predictor.

Usage

```
d2phideta2(link.precision, phi)
```

Arguments

link.precision	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".
phi	precision parameter.

D2Q_Obs_Fisher_bes *D2Q_Obs_Fisher_bes*

Description

Auxiliary function to compute the observed Fisher information matrix for the bessel regression.

Usage

```
D2Q_Obs_Fisher_bes(theta, z, x, v, link.mean, link.precision)
```

Arguments

theta	vector of parameters (all coefficients: kappa and lambda).
z	response vector with 0 < z_i < 1.
x	matrix containing the covariates for the mean submodel. Each column is a different covariate.
v	matrix containing the covariates for the precision submodel. Each column is a different covariate.
link.mean	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
link.precision	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Hessian of the Q-function.

D2Q_Obs_Fisher_bet *D2Q_Obs_Fisher_bet*

Description

Auxiliary function to compute the observed Fisher information matrix for the beta regression.

Usage

```
D2Q_Obs_Fisher_bet(theta, z, x, v, link.mean, link.precision)
```

Arguments

`theta` vector of parameters (all coefficients: kappa and lambda).

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Hessian of the Q-function.

`dbbtest` *dbbtest*

Description

Function to run the discrimination test between beta and bessel regressions (DBB).

Usage

```
dbbtest(formula, data, epsilon = 10^(-5), link.mean, link.precision)
```

Arguments

`formula` symbolic description of the model (set: z ~ x or z ~ x | v); see details below.

`data` arguments considered in the formula description. This is usually a data frame composed by: (i) the response with bounded continuous observations (0 < z_i < 1), (ii) covariates for the mean submodel (columns of matrix x) and (iii) covariates for the precision submodel (columns of matrix v).

`epsilon` tolerance value to control the convergence criterion in the Expectation-Maximization algorithm (default = 10^(-5)).

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Object of class dbbtest, which is a list containing two elements. The 1st one is a table of terms considered in the decision rule of the test; they are sum(z2/n) = sum_i=1^n(z_i^2)/n, sum(quasi_mu) = sum_i=1^n(tildemu_i^2 + tildemu_i(1-tildemu_i)/2) |D_bessel| and |D_beta| as indicated in the main reference. The 2nd term of the list is the name of the selected model (bessel or beta).

See Also

simdata_bes, dbessel, simdata_bet

Examples

```
# Illustration using the Weather task data set available in the bbreg package.
dbbtest(agreement ~ priming + eliciting, data = WT,
link.mean = "logit", link.precision = "identity")
```

---

dbessel *dbessel*

Description

Function to calculate the probability density of the bessel distribution.

Usage

```
dbessel(z, mu, phi)
```

Arguments

z	scalar (0 < z < 1) for which the p.d.f. is to be evaluated.
mu	scalar representing the mean parameter.
phi	scalar representing the precision parameter.

Value

scalar expressing the value of the density at z.

See Also

simdata_bes, dbbtest, simdata_bet

Examples

```
z = seq(0.01, 0.99, 0.01); np = length(z);
density = rep(0, np)
for(i in 1:np){ density[i] = dbessel(z[i], 0.5, 1) }
plot(z, density, type = "l", lwd = 2, cex.lab = 2, cex.axis = 2)
```

DQ2_Obs_Fisher_bes *DQ2_Obs_Fisher_bes*

Description

Auxiliary function to compute the observed Fisher information matrix for the bessel regression.

Usage

```
DQ2_Obs_Fisher_bes(theta, z, x, v, link.mean, link.precision)
```

Arguments

`theta`	vector of parameters (all coefficients: kappa and lambda).
`z`	response vector with 0 < z_i < 1.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

matrix given by the conditional expectation of the gradient of the Q-function and its tranpose.

DQ2_Obs_Fisher_bet *DQ2_Obs_Fisher_bet*

Description

Auxiliary function to compute the observed Fisher information matrix for the beta regression.

Usage

```
DQ2_Obs_Fisher_bet(theta, z, x, v, link.mean, link.precision)
```

Arguments

`theta` vector of parameters (all coefficients: kappa and lambda).

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

matrix given by the conditional expectation of the gradient of the Q-function and its tranpose.

`EMrun_bes` *EMrun_bes*

Description

Function to run the Expectation-Maximization algorithm for the bessel regression.

Usage

```
EMrun_bes(kap, lam, z, x, v, epsilon, link.mean, link.precision)
```

Arguments

`kap` initial values for the coefficients in kappa related to the mean parameter.

`lam` initial values for the coefficients in lambda related to the precision parameter.

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`epsilon` tolerance to control the convergence criterion.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the estimates for kappa and lambda in the bessel regression.

EMrun_bes_dbb *EMrun_bes_dbb*

Description

Function (adapted for the discrimination test between bessel and beta - DBB) to run the Expectation-Maximization algorithm for the bessel regression.

Usage

```
EMrun_bes_dbb(lam, z, v, mu, epsilon, link.precision)
```

Arguments

`lam`	initial values for the coefficients in lambda related to the precision parameter.
`z`	response vector with 0 < z_i < 1.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`mu`	mean parameter (vector having the same size of z).
`epsilon`	tolerance to controll convergence criterion.
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the estimates for lam in the bessel regression.

EMrun_bet *EMrun_bet*

Description

Function to run the Expectation-Maximization algorithm for the beta regression.

Usage

```
EMrun_bet(kap, lam, z, x, v, epsilon, link.mean, link.precision)
```

Arguments

`kap` initial values for the coefficients in kappa related to the mean parameter.

`lam` initial values for the coefficients in lambda related to the precision parameter.

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`epsilon` tolerance to control the convergence criterion.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the estimates for kappa and lambda in the beta regression.

`EMrun_bet_dbb` *EMrun_bet_dbb*

Description

Function (adapted for the discrimination test between bessel and beta - DBB) to run the Expectation-Maximization algorithm for the beta regression.

Usage

```
EMrun_bet_dbb(lam, z, v, mu, epsilon, link.precision)
```

Arguments

`lam` initial values for the coefficients in lambda related to the precision parameter.

`z` response vector with 0 < z_i < 1.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`mu` mean parameter (vector having the same size of z).

`epsilon` tolerance to controll convergence criterion.

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the estimates for lam in the beta regression.

envelope_bes *envelope_bes*

Description

Function to calculate envelopes based on residuals for the bessel regression.

Usage

```
envelope_bes(
  residual,
  kap,
  lam,
  x,
  v,
  nsim_env,
  prob,
  n,
  epsilon,
  link.mean,
  link.precision
)
```

Arguments

`residual`	character indicating the type of residual ("pearson", "score" or "quantile").
`kap`	coefficients in kappa related to the mean parameter.
`lam`	coefficients in lambda related to the precision parameter.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`nsim_env`	number of synthetic data sets to be generated.
`prob`	confidence level of the envelope (number between 0 and 1).
`n`	sample size.
`epsilon`	tolerance parameter used in the Expectation-Maximization algorithm applied to the synthetic data.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Matrix with dimension 2 x n (1st row = upper bound, second row = lower bound).

envelope_bet *envelope_bet*

Description

Function to calculate envelopes based on residuals for the beta regression.

Usage

```
envelope_bet(
  residual,
  kap,
  lam,
  x,
  v,
  nsim_env,
  prob,
  n,
  epsilon,
  link.mean,
  link.precision
)
```

Arguments

`residual`	character indicating the type of residual ("pearson", "score" or "quantile").
`kap`	coefficients in kappa related to the mean parameter.
`lam`	coefficients in lambda related to the precision parameter.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`nsim_env`	number of synthetic data sets to be generated.
`prob`	confidence level of the envelope (number between 0 and 1).
`n`	sample size.
`epsilon`	tolerance parameter used in the Expectation-Maximization algorithm applied to the synthetic data.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Matrix with dimension 2 x n (1st row = upper bound, second row = lower bound).

See Also

score_residual_bet, quantile_residual_bet, pred_accuracy_bet

Ew1z *Ew1z*

Description

Auxiliary function required in the Expectation-Maximization algorithm (E-step) and in the calculation of the Fisher information matrix. It represents the conditional expected value E(W_i^s|Z_i), with s = -1; i.e., latent W_i^(-1) given the observed Z_i.

Usage

```
Ew1z(z, mu, phi)
```

Arguments

z	response vector with 0 < z_i < 1.
mu	mean parameter (vector having the same size of z).
phi	precision parameter (vector having the same size of z).

Value

Vector of expected values.

Ew2z *Ew2z*

Description

Auxiliary function required in the calculation of the Fisher information matrix. It represents the conditional expected value E(W_i^s|Z_i), with s = -2; i.e., latent W_i^(-2) given the observed Z_i.

Usage

```
Ew2z(z, mu, phi)
```

Arguments

z	response vector with 0 < z_i < 1.
mu	mean parameter (vector having the same size of z).
phi	precision parameter (vector having the same size of z).

Value

vector of expected values.

`fitted.bbreg` *fitted.bbreg*

Description

Function providing the fitted means for the model (bessel or beta).

Usage

```
## S3 method for class 'bbreg'
fitted(object, type = c("response", "link", "precision", "variance"), ...)
```

Arguments

`object`	object of class "bbreg" containing results from the fitted model.
`type`	the type of variable to get the fitted values. The default is the "response" type, which provided the estimated values for the means. The type "link" provides the estimates for the linear predictor of the mean. The type "precision" provides estimates for the precision parameters whereas the type "variance" provides estimates for the variances.
`...`	further arguments passed to or from other methods.

See Also

`predict.bbreg`, `summary.bbreg`, `coef.bbreg`, `vcov.bbreg`, `plot.bbreg`

Examples

```
fit = bbreg(agreement ~ priming + eliciting, data = WT)
fitted(fit)
fitted(fit, type = "precision")
```

`gradlam_bes_dbb` *gradlam_bes_dbb*

Description

Gradient of the Q-function (adapted for the discrimination test between bessel and beta - DBB) to calculate the gradient required for optimization via `optim`. This option is related to the bessel regression.

Usage

```
gradlam_bes_dbb(lam, wz, z, v, mu, link.precision)
```

Arguments

`lam` coefficients in lambda related to the covariates in v.

`wz` parameter wz representing E(1/W_i|Z_i = z_i, theta).

`z` response vector with 0 < z_i < 1.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`mu` mean parameter (vector having the same size of z).

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary gradient function for the bessel case.

`gradlam_bet_dbb` *gradlam_bet*

Description

Gradient of the Q-function (adapted for the discrimination test between bessel and beta - DBB) to calculate the gradient required for optimization via `optim`. This option is related to the beta regression.

Usage

```
gradlam_bet_dbb(lam, phiold, z, v, mu, link.precision)
```

Arguments

`lam` coefficients in lambda related to the covariates in v.

`phiold` previous value of the precision parameter (phi).

`z` response vector with 0 < z_i < 1.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`mu` mean parameter (vector having the same size of z).

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary gradient function for the beta case.

gradtheta_bes *gradtheta_bes*

Description

Function to calculate the gradient of the Q-function, which is required for optimization via `optim`. This option is related to the bessel regression.

Usage

```
gradtheta_bes(theta, wz, z, x, v, link.mean, link.precision)
```

Arguments

`theta` vector of parameters (all coefficients: kappa and lambda).

`wz` parameter representing E(1/W_i|Z_i = z_i, theta).

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

vector representing the output of this auxiliary gradient function for the bessel case.

gradtheta_bet *gradtheta_bet*

Description

Function to calculate the gradient of the Q-function, which is required for optimization via `optim`. This option is related to the beta regression.

Usage

```
gradtheta_bet(theta, phiold, z, x, v, link.mean, link.precision)
```

Arguments

`theta`	vector of parameters (all coefficients).
`phiold`	previous value of the precision parameter (phi).
`z`	response vector with 0 < z_i < 1.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary gradient function for the beta case.

`infmat_bes` *infmat_bes*

Description

Function to compute standard errors based on the Fisher information matrix for the bessel regression. This function can also provide the Fisher's information matrix.

Usage

```
infmat_bes(theta, z, x, v, link.mean, link.precision, information = FALSE)
```

Arguments

`theta`	vector of parameters (all coefficients: kappa and lambda).
`z`	response vector with 0 < z_i < 1.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".
`information`	optionally, a logical parameter indicating whether the Fisher's information matrix should be returned

Value

Vector of standard errors or Fisher's information matrix if the parameter 'information' is set to TRUE.

`infmat_bet` *infmat_bet*

Description

Function to compute standard errors based on the Fisher information matrix for the beta regression. This function can also provide the Fisher's information matrix.

Usage

```
infmat_bet(theta, z, x, v, link.mean, link.precision, information = FALSE)
```

Arguments

`theta` vector of parameters (all coefficients: kappa and lambda).

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

`information` optionally, a logical parameter indicating whether the Fisher's information matrix should be returned

Value

Vector of standard errors or Fisher's information matrix if the parameter 'information' is set to TRUE.

plot.bbreg *plot.bbreg*

Description

Function to build useful plots for bounded regression models.

Usage

```
## S3 method for class 'bbreg'
plot(x, which = c(1, 2, 3, 4), ask = TRUE, main = "", qqline = TRUE, ...)
```

Arguments

x	object of class "bbreg" containing results from the fitted model. If the model is fitted with envelope = 0, the Q-Q plot will be produced without envelopes.
which	a number of a vector of numbers between 1 and 4. Plot 1: Residuals vs. Index; Plot 2: Q-Q Plot (if the fit contains simulated envelopes, the plot will be with the simulated envelopes); Plot 3: Fitted means vs. Response; Plot 4: Residuals vs. Fitted means.
ask	logical; if TRUE, the user is aked before each plot.
main	character; title to be placed at each plot additionally (and above) all captions.
qqline	logical; if TRUE and the fit does *not* contain simulated envelopes, a qqline will be added to the normal Q-Q plot.
...	graphical parameters to be passed.

See Also

summary.bbreg, coef.bbreg, vcov.bbreg, fitted.bbreg, predict.bbreg

Examples

```
n = 100; x = cbind(rbinom(n, 1, 0.5), runif(n, -1, 1)); v = runif(n, -1, 1);
z = simdata_bes(kap = c(1, 1, -0.5), lam = c(0.5, -0.5), x, v, repetitions = 1,
link.mean = "logit", link.precision = "log")
z = unlist(z)
fit = bbreg(z ~ x | v, envelope = 10)
plot(fit)
plot(fit, which = 2)
plot(fit, which = c(1,4), ask = FALSE)
```

predict.bbreg *predict.bbreg*

Description

Function to obtain various predictions based on the fitted model (bessel or beta).

Usage

```
## S3 method for class 'bbreg'
predict(
  object,
  newdata = NULL,
  type = c("response", "link", "precision", "variance"),
  ...
)
```

Arguments

object	object of class "bbreg" containing results from the fitted model.
newdata	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted response values will be provided.
type	the type of prediction. The default is the "response" type, which provided the estimated values for the means. The type "link" provides the estimates for the linear predictor. The type "precision" provides estimates for the precision parameters whereas the type "variance" provides estimates for the variances.
...	further arguments passed to or from other methods.

See Also

fitted.bbreg, summary.bbreg, coef.bbreg, vcov.bbreg, plot.bbreg

Examples

```
fit = bbreg(agreement ~ priming + eliciting, data = WT)
predict(fit)
new_data_example = data.frame(priming = c(0,0,1), eliciting = c(0,1,1))
predict(fit, new_data = new_data_example)
predict(fit, new_data = new_data_example, type = "precision")
```

pred_accuracy_bes *pred_accuracy_bes*

Description

Function to calculate the Residual Sum of Squares for partitions (training and test sets) of the data set. Residuals are calculated here based on the bessel regression.

Usage

```
pred_accuracy_bes(
  residual,
  kap,
  lam,
  z,
  x,
  v,
  ntest,
  predict,
  epsilon,
  link.mean,
  link.precision
)
```

Arguments

`residual` Character indicating the type of residual ("pearson", "score" or "quantile").

`kap` coefficients in kappa related to the mean parameter.

`lam` coefficients in lambda related to the precision parameter.

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`ntest` number of observations in the test set for prediction.

`predict` number of partitions (training and test sets) to be evaluated.

`epsilon` tolerance parameter used in the Expectation-Maximization algorithm for the training data set.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the RSS for each partition of the full data set.

pred_accuracy_bet *pred_accuracy_bet*

Description

Function to calculate the Residual Sum of Squares for partitions (training and test sets) of the data set. Residuals are calculated here based on the beta regression.

Usage

```
pred_accuracy_bet(
  residual,
  kap,
  lam,
  z,
  x,
  v,
  ntest,
  predict,
  epsilon,
  link.mean,
  link.precision
)
```

Arguments

residual	Character indicating the type of residual ("pearson", "score" or "quantile").
kap	coefficients in kappa related to the mean parameter.
lam	coefficients in lambda related to the precision parameter.
z	response vector with 0 < z_i < 1.
x	matrix containing the covariates for the mean submodel. Each column is a different covariate.
v	matrix containing the covariates for the precision submodel. Each column is a different covariate.
ntest	number of observations in the test set for prediction.
predict	number of partitions (training and test sets) to be evaluated.
epsilon	tolerance parameter used in the Expectation-Maximization algorithm for the training data set.
link.mean	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
link.precision	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the RSS for each partition of the full data set.

See Also

score_residual_bet, quantile_residual_bet, envelope_bet

print.bbreg *print.bbreg*

Description

Function providing a brief description of results related to the regression model (bessel or beta).

Usage

```
## S3 method for class 'bbreg'
print(x, ...)
```

Arguments

x	object of class "bbreg" containing results from the fitted model.
...	further arguments passed to or from other methods.

See Also

fitted.bbreg, summary.bbreg, coef.bbreg, vcov.bbreg, plot.bbreg, predict.bbreg

Examples

```
fit = bbreg(agreement ~ priming + eliciting, data = WT)
fit
```

Qf_bes *Qf_bes*

Description

Q-function related to the bessel model. This function is required in the Expectation-Maximization algorithm.

Usage

```
Qf_bes(theta, wz, z, x, v, link.mean, link.precision)
```

Arguments

`theta`	vector of parameters (all coefficients: kappa and lambda).
`wz`	parameter representing E(1/W_i|Z_i = z_i, theta).
`z`	response vector with 0 < z_i < 1.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary function for the bessel case.

`Qf_bes_dbb` *Qf_bes_dbb*

Description

Q-function related to the bessel model. This function was adapted for the discrimination test between bessel and beta (DBB) required in the Expectation-Maximization algorithm.

Usage

```
Qf_bes_dbb(lam, wz, z, v, mu, link.precision)
```

Arguments

`lam`	coefficients in lambda related to the covariates in v.
`wz`	parameter wz representing E(1/W_i|Z_i = z_i, theta).
`z`	response vector with 0 < z_i < 1.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`mu`	mean parameter (vector having the same size of z).
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary function for the bessel case.

Qf_bet *Qf_bet*

Description

Q-function related to the beta model. This function is required in the Expectation-Maximization algorithm.

Usage

```
Qf_bet(theta, phiold, z, x, v, link.mean, link.precision)
```

Arguments

theta	vector of parameters (all coefficients).
phiold	previous value of the precision parameter (phi).
z	response vector with 0 < z_i < 1.
x	matrix containing the covariates for the mean submodel. Each column is a different covariate.
v	matrix containing the covariates for the precision submodel. Each column is a different covariate.
link.mean	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
link.precision	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary function for the beta case.

Qf_bet_dbb *Qf_bet_dbb*

Description

Q-function related to the beta model. This function was adapted for the discrimination test between bessel and beta (DBB) required in the Expectation-Maximization algorithm.

Usage

```
Qf_bet_dbb(lam, phiold, z, v, mu, link.precision)
```

Arguments

lam coefficients in lambda related to the covariates in v.

phiold previous value of the precision parameter (phi).

z response vector with 0 < z_i < 1.

v matrix containing the covariates for the precision submodel. Each column is a different covariate.

mu mean parameter (vector having the same size of z).

link.precision a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Scalar representing the output of this auxiliary function for the beta case.

quantile_residual_bes *quantile_residual_bes*

Description

Function to calculate quantile residuals based on the bessel regression. Details about this type of residual can be found in *Pereira (2019)*.

Usage

```
quantile_residual_bes(kap, lam, z, x, v, link.mean, link.precision)
```

Arguments

kap coefficients in kappa related to the mean parameter.

lam coefficients in lambda related to the precision parameter.

z response vector with 0 < z_i < 1.

x matrix containing the covariates for the mean submodel. Each column is a different covariate.

v matrix containing the covariates for the precision submodel. Each column is a different covariate.

link.mean a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

link.precision a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the quantile residuals.

References

DOI:10.1080/03610918.2017.1381740 (Pereira; 2019)

See Also

`score_residual_bes`

`quantile_residual_bet` *quantile_residual_bet*

Description

Function to calculate quantile residuals based on the beta regression. Details about this type of residual can be found in *Pereira (2019)*.

Usage

```
quantile_residual_bet(kap, lam, z, x, v, link.mean, link.precision)
```

Arguments

`kap`	coefficients in kappa related to the mean parameter.
`lam`	coefficients in lambda related to the precision parameter.
`z`	response vector with 0 < z_i < 1.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the quantile residuals.

References

DOI:10.1080/03610918.2017.1381740 (Pereira; 2019)

See Also

`score_residual_bet`

SA *Stress/Axiety data set*

Description

Stress and anxiety scores among nonclinical women in Townsville - Queensland, Australia.

Usage

```
data(SA)
```

Format

Data frame containing 166 observations on 2 variables.

stress score, linearly transformed to the open unit interval.

anxiety score, linearly transformed to the open unit interval.

Source

Data can be obtained from the supplementary materials of *Smithson and Verkuilen (2006)*. See also *Barreto-Souza, Mayrink and Simas (2020)* for details.

References

arXiv:2003.05157 (Barreto-Souza, Mayrink and Simas; 2020)

DOI:10.1037/1082-989X.11.1.54 (Smithson and Verkuilen (2006))

Examples

```
data(SA)
```

score_residual_bes *score_residual_bes*

Description

Function to calculate the empirical score residuals based on the bessel regression.

Usage

```
score_residual_bes(
  kap,
  lam,
  z,
  x,
  v,
  nsim_score = 100,
  link.mean,
  link.precision
)
```

Arguments

`kap`	coefficients in kappa related to the mean parameter.
`lam`	coefficients in lambda related to the precision parameter.
`z`	response vector with 0 < z_i < 1.
`x`	matrix containing the covariates for the mean submodel. Each column is a different covariate.
`v`	matrix containing the covariates for the precision submodel. Each column is a different covariate.
`nsim_score`	number synthetic data sets (default = 100) to be generated as a support to estime mean and s.d. of log(z)-log(1-z).
`link.mean`	a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".
`link.precision`	a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the score residuals.

See Also

`quantile_residual_bes`

`score_residual_bet` *score_residual_bet*

Description

Function to calculate the empirical score residuals based on the beta regression.

Usage

```
score_residual_bet(
  kap,
  lam,
  z,
  x,
  v,
  nsim_score = 100,
  link.mean,
  link.precision
)
```

Arguments

`kap` coefficients in kappa related to the mean parameter.

`lam` coefficients in lambda related to the precision parameter.

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`nsim_score` number synthetic data sets (default = 100) to be generated as a support to estime mean and s.d. of log(z)-log(1-z).

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

Vector containing the score residuals.

See Also

`quantile_residual_bet`

`simdata_bes` *simdata_bes*

Description

Function to generate synthetic data from the bessel regression. Requires the R package "statmod" generate random numbers from the Inverse Gaussian distribution (*Giner and Smyth, 2016*).

Usage

```
simdata_bes(kap, lam, x, v, repetitions = 1, link.mean, link.precision)
```

Arguments

`kap` coefficients in kappa related to the mean parameter.

`lam` coefficients in lambda related to the precision parameter.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`repetitions` the number of random draws to be made.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

a list of response vectors z (with 0 < z_i < 1).

References

DOI:10.32614/RJ-2016-024 (Giner and Smyth; 2016)

See Also

`dbessel`, `dbbtest`, `simdata_bet`

Examples

```
n = 100; x = cbind(rbinom(n, 1, 0.5), runif(n, -1, 1)); v = runif(n, -1, 1);
z = simdata_bes(kap = c(1, -1, 0.5), lam = c(0.5, -0.5), x, v,
repetitions = 1, link.mean = "logit", link.precision = "log")
z = unlist(z)
hist(z, xlim = c(0, 1), prob = TRUE)
```

`simdata_bet` *simdata_bet*

Description

Function to generate synthetic data from the beta regression.

Usage

```
simdata_bet(kap, lam, x, v, repetitions = 1, link.mean, link.precision)
```

Arguments

`kap` coefficients kappa related to the mean parameter.

`lam` coefficients lambda related to the precision parameter.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`repetitions` the number of random draws to be made.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`link.precision` a string containing the link function the precision parameter. The possible link functions for the precision parameter are "identity", "log", "sqrt", "inverse".

Value

a list of response vectors z (with 0 < z_i < 1).

See Also

`simdata_bes`, `dbessel`, `dbbtest`

Examples

```
n = 100; x = cbind(rbinom(n, 1, 0.5), runif(n, -1, 1)); v = runif(n, -1, 1);
z = simdata_bet(kap = c(1, -1, 0.5), lam = c(0.5,- 0.5), x, v, repetitions = 1,
link.mean = "logit", link.precision = "log")
z = unlist(z)
hist(z, xlim = c(0, 1), prob = TRUE)
```

`startvalues` *startvalues*

Description

Function providing initial values for the Expectation-Maximization algorithm.

Usage

```
startvalues(z, x, v, link.mean)
```

Arguments

`z` response vector with 0 < z_i < 1.

`x` matrix containing the covariates for the mean submodel. Each column is a different covariate.

`v` matrix containing the covariates for the precision submodel. Each column is a different covariate.

`link.mean` a string containing the link function for the mean. The possible link functions for the mean are "logit","probit", "cauchit", "cloglog".

`summary.bbreg` *summary.bbreg*

Description

Function providing a summary of results related to the regression model (bessel or beta).

Usage

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

Arguments

`object` an object of class "bbreg" containing results from the fitted model.

`...` further arguments passed to or from other methods.

See Also

`fitted.bbreg`, `plot.bbreg`, `predict.bbreg`

Examples

```
fit = bbreg(agreement ~ priming + eliciting|priming, data = WT)
summary(fit)
```

vcov.bbreg *vcov.bbreg*

Description

Function to extract the variance-covariance matrix of the parameters of the fitted regression model (bessel or beta).

Usage

```
## S3 method for class 'bbreg'
vcov(object, parameters = c("all", "mean", "precision"), ...)
```

Arguments

object	an object of class "bbreg" containing results from the fitted model.
parameters	a string to determine which coefficients should be extracted: 'all' extracts all coefficients, 'mean' extracts the coefficients of the mean parameters and 'precision' extracts coefficients of the precision parameters.
...	further arguments passed to or from other methods.

See Also

infmat_bes, infmat_bet

Examples

```
fit = bbreg(agreement ~ priming + eliciting|priming, data = WT)
vcov(fit)
vcov(fit, parameters = "precision")
```

WT *Weather Task data set*

Description

Weather task data set.

Usage

```
data(WT)
```

Format

Data frame containing 345 observations on 3 variables.

agreement probability or the average between two probabilities indicated by each individual.

priming categorical covariate (0 = two-fold, 1 = seven-fold).

eliciting categorical covariate (0 = precise, 1 = imprecise).

Source

Data can be obtained from supplementary materials of *Smithson et al. (2011)*. See also *Barreto-Souza, Mayrink and Simas (2020)* for details.

References

arXiv:2003.05157 (Barreto-Souza, Mayrink and Simas; 2020)

DOI:10.1080/15598608.2009.10411918 (Smithson and Verkuilen; 2009)

DOI:10.3102/1076998610396893 (Smithson et al.; 2011)

Examples

```
data(WT)
```

Index