Package ‘FlexReg’

November 5, 2020

Title Regression Models for Bounded Responses

Version 1.0

Author Agnese M. Di Brisco [aut, cre], Roberto Ascari [aut, cre], Sonia Migliorati [aut], Andrea Ongaro [aut]

Maintainer Roberto Ascari <roberto.ascari@unimib.it>

Description Functions to fit regression models for bounded responses, such as rates and proportions. Available models are the flexible beta (Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018) <doi:10.1214/17-BA1079>), the variance-inflated beta (Di Brisco, A. M., Migliorati, S., Ongaro, A. (2020) <doi:10.1177/1471082X18821213>), and the beta one (Ferrari, S.L.P., and Cribari-Neto, F. (2004) <doi:10.1080/0266476042000214501>). Inference is dealt with a Bayesian approach based on the Hamiltonian Monte Carlo (HMC) algorithm (Gelman, A.; Carlin, J. B.; Stern, H. S. and Rubin, D. B. (2014) <doi:10.1201/b16018>). Besides, functions to compute residuals, posterior predictives, goodness-of-fit measures, convergence diagnostics, and graphical representations are provided.

License GPL (>= 2)

Encoding UTF-8

LazyData true

Biarch true

Depends R (>= 3.5.0)

Imports methods, Rcpp (>= 0.12.0), rstan (>= 2.18.1), rstantools (>= 2.0.0), loo, bayesplot, ggplot2, Formula, utils, grDevices

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make

RoxygenNote 7.1.1

Suggests testthat

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-11-05 08:20:05 UTC

R topics documented:

`FlexReg-package` *The 'FlexReg' package.*

Description

This package provides functions and methods to implement several types of regression models for bounded responses such as proportions and rates. Inferential statistical analysis is dealt with by Bayesian estimation based on the Hamiltonian Monte Carlo (HMC) algorithm through the `` `rstan` `` package.

References

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018) A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

Di Brisco, A. M., Migliorati, S., Ongaro, A. (2020) Robustness against outliers: A new variance inflated regression model for proportions. Statistical Modelling, **20**(3), 274–309. doi:10.1177/1471082X18821213

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. Journal of Applied Statistics, **31**(7), 799–815. doi:10.1080/0266476042000214501

Stan Development Team (2020). RStan: the R interface to Stan. R package version 2.19.3. https://mc-stan.org

---

`Consumption` *Italian Households Consumption data*

Description

This dataset is a subset from the 2016 Survey on Household Income and Wealth data, a statistical survey conducted by Bank of Italy. The statistical units are the households and the head of the household is conventionally selected as the major income earner.

Format

A data frame containing 568 observations on the following 8 variables.

`NComp` the number of household members.

`Sex` the sex of the head of household.

`Age` the age of the head of household.

`NEarners` the number of household income earners.

`Area` a factor indicating the geographical area where the household is located.

`Citizenship` a factor indicating the citizenship of the head of household.

`Income` the net disposable income.

`Consumption` the propensity to consume, defined as the percentage of `Income` that is spent rather than saved.

Details

Full data are available on the website of the Bank of Italy. `Consumption` has been created by dividing the variable 'consumption' over the 'net disposable income'.

Source

Bank of Italy, Survey on Household Income and Wealth, 2016.

Survey description.

`convergence.diag` *Convergence diagnostics*

Description

The function returns some diagnostics to check for convergence to the equilibrium distribution of the Markov Chain(s). Moreover, it prints the number (and percentage) of iterations that ended with a divergence and that saturated the max treedepth, and the E-BFMI values for each chain for which E-BFMI is less than 0.2.

Usage

```
convergence.diag(
  model,
  diagnostics = "all",
  pars = NULL,
  additional.args = list()
)
```

Arguments

`model`	an object of class flexreg.
`diagnostics`	an optional character vector of diagnostics names. The default is to compute `all` diagnostics, otherwise one can specify a selection of diagnostics among `Rhat`, `geweke`, `gaftery`, `heidel`, and `gelman`.
`pars`	an optional character vector of parameter names. If `pars` is not specified, all parameters in the regression model are evaluated.
`additional.args`	a list containing additional arguments (see details)

Details

- `R-hat` returns the potential scale reduction factor on split chains. An R-hat greater than 1 is indicative of a bad mix of the chains. At convergence R-hat has to be less than 1.05. See `rstan::Rhat` for further details.
- `geweke` returns the z-scores, one for each parameter, for a test of equality between the means of the first 10% and last 50% of the chain. The fraction to use from the first and last part of the chain can be edited through the additional arguments `frac1` and `frac2`. The sum of `frac1` and `frac2` has to be strictly less than 1. See `coda::geweke.diag` for further details.
- `raftery` returns the length of "burn in" (M), the required sample size (N), the minimum sample size for a chain with zero autocorrelation ($Nmin$), and the estimate of the "dependence factor" ($I = (M + N)/Nmin$). Values of I greater than 5 may be indicative of a strong autocorrelation. Additional parameters such as the quantile to be estimated (q), the desired margin of error of the estimate (r), and the probability of obtaining an estimate between $q - r$ and $q + r$ (s) can be passed as list in the `additional.args` argument. See `coda::raftery.diag` for further details.

- `heidel` returns a p-value referred to a convergence test where the null hypothesis is that the sampled values come from a stationary distribution. It is possible to set the target value for ratio of halfwidth to sample mean (eps) and the significance level of the test ($pvalue$) into the `additional.args` argument. See `coda::heidel.diag` for further details.
- `gelman` returns the estimate of the potential scale reduction factor and the upper confidence limit. At least two chains are needed to compute the Gelman and Rubin's convergence diagnostics. Additional parameters such as the confidence level ($confidence$), a logical flag indicating whether variables should be transformed ($transform$), a logical flag indicating whether only the second half of the series should be used in the computation ($autoburnin$), and a logical flag indicating whether the multivariate potential scale reduction factor should be calculated for multivariate chains ($multivariate$) can be passed as list in the `additional.args` argument. See `coda::gelman.diag` for further details.

Value

A print from `check_hmc_diagnostics` function and a list of convergence diagnostics.

References

Brooks, SP. and Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. Journal of Computational and Graphical Statistics, **7**, 434-455.

Geweke, J. (1992). Evaluating the accuracy of sampling-based approaches to calculating posterior moments. In Bayesian Statistics 4 (ed JM Bernado, JO Berger, AP Dawid and AFM Smith). Clarendon Press, Oxford, UK.

Raftery, A.E. and Lewis, S.M. (1992). One long run with diagnostics: Implementation strategies for Markov chain Monte Carlo. Statistical Science, **7**, 493-497.

Heidelberger P. and Welch P.D. (1981). A spectral method for confidence interval generation and run length control in simulations. Comm. ACM. **24**, 233-245.

The Stan Development Team Stan Modeling Language User's Guide and Reference Manual. http://mc-stan.org/.

`convergence.plot` *Convergence plots*

Description

The function produces a .pdf file containing some convergence plots for the Monte Carlo draws.

Usage

```
convergence.plot(
  model,
  file = "convergence-output.pdf",
```

```
  plotfun = "all",
  pars = NULL,
  point_est = "median",
  prob = 0.5,
  prob_outer = 0.9,
  lags = 10,
  warmup = F,
  width = 7,
  height = 7
)
```

Arguments

`model`	an object of class `flexreg`.
`file`	a character string giving the name of the file (with extension .pdf).
`plotfun`	an optional character vector of diagnostics plots. The default is to compute `all` plots, otherwise one can specify a selection of plots among `density`, `trace`, `intervals`, `rate`, `rhat`, and `acf`.
`pars`	an optional character vector of parameter names. If pars is not specified, all parameters in the regression model are evaluated.
`point_est`	an optional character to specify the point estimate to be shown between `median` (the default), `mean`, or `none`.
`prob`	the probability mass to be included in the inner interval (`intervals` plot) or in the shaded region (for `density` plot). The default is 0.5.
`prob_outer`	the probability mass to be included in the outer interval of the `intervals` plot. The default is 0.9.
`lags`	the number of lags to be shown in the `acf` plot. The default is 10.
`warmup`	a logical scalar indicating whether to include the warmup draws or not (default).
`width, height`	the width and height of the graphics region of each plot in inches. The default values are 7.

Details

The plots can be further customized using the `ggplot2` package.

- `density` returns a density plot for each parameter in `pars` computed from the posterior draws. See `bayesplot::mcmc_areas` for further details.
- `trace` returns a trace plot for each parameter in `pars` computed from the posterior draws. See `bayesplot::mcmc_trace` for further details.
- `intervals` returns a plot of uncertainty interval for each parameter in `pars` computed from the posterior draws. See `bayesplot::mcmc_intervals` for further details.
- `rate` returns a plot for each parameter in `pars` with the number of iterations on the x-axis and the monte carlo mean until iteration i-th on the y-axis.
- `rhat` returns a plot with the Rhat values for each parameter in `pars`. See `bayesplot::mcmc_rhat` for further details.
- `acf` returns the autocorrelation plots (one for each parameter in `pars`). See `bayesplot::mcmc_acf` for further details.

Value

A .pdf file with one plot per page.

`curve.density` *Draw density plots*

Description

The function draws a curve corresponding to the probability density function of the specified distribution (beta, flexible beta, or variance-inflated beta).

Usage

```
curve.density(
  type = NULL,
  mu = NULL,
  phi = NULL,
  p = NULL,
  w = NULL,
  k = NULL,
  ...
)
```

Arguments

`type`	a character specifying the distribution type to be plotted (`"Beta"`, `"FB"`, or `"VIB"`).
`mu`	the mean parameter of the distribution. It must lie in (0, 1).
`phi`	the precision parameter of the distribution. It must be a positive real value.
`p`	the mixing weight (to be specified if `type = "FB"` or `type = "VIB"`). It must lie in (0, 1).
`w`	the normalized distance among clusters of the FB distribution (to be specified if `type = "FB"`). It must lie in (0, 1).
`k`	the extent of the variance inflation (to be specified if `type = "VIB"`). It must lie in (0, 1).
`...`	additional arguments of `stat_function()`.

References

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018) A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

Di Brisco, A. M., Migliorati, S., Ongaro, A. (2020) Robustness against outliers: A new variance inflated regression model for proportions. Statistical Modelling, **20**(3), 274–309. doi:10.1177/1471082X18821213

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. Journal of Applied Statistics, **31**(7), 799–815. doi:10.1080/0266476042000214501

Examples

```
curve.density("Beta", mu=0.5, phi=20)
curve.density("FB", mu=0.5, phi=20, p=0.4, w=.8)
curve.density("VIB", mu=0.5, phi=20, p=0.9, k=.8, col=3)
```

`dBeta_mu` *Beta probability density function*

Description

The function computes the probability density function of the beta distribution with a mean-precision parameterization.

Usage

```
dBeta_mu(x, mu, phi)
```

Arguments

`x`	a vector of quantiles.
`mu`	the mean parameter of the beta distribution. It must lie in (0, 1).
`phi`	the precision parameter of the Beta distribution. It must be a positive real value.

Details

The beta distribution has density

$$\frac{\Gamma(\phi)}{\Gamma(\mu\phi)\Gamma((1-\mu)\phi)}x^{\mu\phi-1}(1-x)^{(1-\mu)\phi-1}$$

for $0 < x < 1$, where $0 < \mu < 1$ identifies the mean and $\phi > 0$ is the precision parameter.

Value

A vector with the same length as `x`.

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. Journal of Applied Statistics, **31**(7), 799–815. doi:10.1080/0266476042000214501

Examples

```
dBeta_mu(x = c(.5,.7,.8), mu = 0.3, phi = 20)
```

dFB *Flexible beta probability density function*

Description

The function computes the probability density function of the flexible beta distribution.

Usage

```
dFB(x, mu, phi, p, w)
```

Arguments

x	a vector of quantiles.
mu	the mean parameter of the flexible beta distribution. It must lie in (0, 1).
phi	the precision parameter of the flexible beta distribution. It must be a positive real value.
p	the mixing weight. It must lie in (0, 1).
w	the normalized distance among clusters. It must lie in (0, 1).

Details

The FB distribution is a special mixture of two beta distributions

$$pBeta(x|\lambda_1, \phi) + (1 - p)Beta(x|\lambda_2, \phi)$$

for $0 < x < 1$ where $Beta(x|\cdot, \cdot)$ is the beta distribution with a mean-precision parameterization. Moreover, $0 < p < 1$ is the mixing weight, $\phi > 0$ is a precision parameter, $\lambda_1 = \mu + (1 - p)w$ and $\lambda_2 = \mu - pw$ are the component means of the first and second component of the mixture, $0 < \mu = p\lambda_1 + (1 - p)\lambda_2 < 1$ is the overall mean, and $0 < w < 1$ is the normalized distance between clusters.

Value

A vector with the same length as x.

References

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018). A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

Examples

```
dFB(x = c(.5,.7,.8), mu = 0.3, phi = 20, p = .5, w = .5)
```

dVIB *Variance-inflated beta probability density function*

Description

The function computes the probability density function of the variance-inflated beta distribution.

Usage

```
dVIB(x, mu, phi, p, k)
```

Arguments

x	a vector of quantiles.
mu	the mean parameter of the variance-inflated beta distribution. It must lie in (0, 1).
phi	the precision parameter of the Variance-Inflated distribution. It must be a positive real value.
p	the mixing weight. It must lie in (0, 1).
k	the extent of the variance inflation. It must lie in (0, 1).

Details

The VIB distribution is a special mixture of two beta distributions

$$pBeta(x|\mu, \phi k) + (1 - p)Beta(x|\mu, \phi)$$

for $0 < x < 1$ where $Beta(x|\cdot, \cdot)$ is the beta distribution with a mean-precision parameterization. Moreover, $0 < p < 1$ is the mixing weight, $0 < \mu < 1$ represents the overall (as well as mixture component) mean, $\phi > 0$ is a precision parameter, and $0 < k < 1$ determines the extent of the variance inflation.

Value

A vector with the same length as x.

References

Di Brisco, A. M., Migliorati, S., Ongaro, A. (2020) Robustness against outliers: A new variance inflated regression model for proportions. Statistical Modelling, **20**(3), 274–309. doi:10.1177/1471082X18821213

Examples

```
dVIB(x = c(.5,.7,.8), mu = 0.3, phi = 20, p = .5, k= .5)
```

Election *Italian Election Results*

Description

Results of the Italian general election held on 4 March 2018 for six parties.

Format

A data frame containing 232 observations on 13 variables.

`NVotes` the number of valid votes.

`FI` the percentage of votes got by 'Forza Italia' party.

`FDI` the percentage of votes got by 'Fratelli d'Italia' party.

`LEGA` the percentage of votes got by 'Lega' party.

`LEU` the percentage of votes got by 'Liberi e Uguali' party.

`M5S` the percentage of votes got by 'Movimento 5 Stelle' party.

`PD` the percentage of votes got by 'Partito Democratico' party.

`Other` the percentage of votes got by other parties, including blank ballots.

`AgeInd` the age index, defined as the ratio of the number of elderly persons (aged 65 and over) to the number of young persons (from 0 to 14), multiplied by 100.

`PopDens` the number of inhabitants per km2.

`ER` the employment rate, defined as the ratio of the number of employed persons (aged 15-64) to the number of persons (aged 15-64).

`Illiteracy` the illiteracy rate, defined as the ratio of the number of persons without a qualification (aged 15 and over) to the total number of persons aged 15 and over.

`Foreign` the number of foreigners per 1000 inhabitants.

Details

Data are collected on the 232 electoral districts into which the Italian territory is organized. Distribution of votes for Aosta constituency is not available. Distributions of votes are available on the Italian Ministry of Interior's webpage whereas constituencies information have been obtained from 2011 Italian Census.

Source

Italian Ministry of Interior's webpage.

flexreg *Flexible Regression Models for Proportions*

Description

The function fits some flexible regression models for proportions via a Bayesian approach to inference based on Hamiltonian Monte Carlo algorithm. Available regression models are the flexible beta regression model (`type="FB"`, default), the variance inflated beta (`type="VIB"`), and the beta one (`type="Beta"`).

Usage

```
flexreg(
  formula,
  data,
  type = "FB",
  link.mu = "logit",
  prior.beta = "normal",
  hyperparam.beta = 100,
  link.phi = NULL,
  prior.phi = NULL,
  hyperparam.phi = NULL,
  prior.psi = NULL,
  hyperparam.psi = NULL,
  n.iter = 5000,
  burnin.perc = 0.5,
  n.chain = 1,
  thin = 1,
  verbose = TRUE,
  ...
)
```

Arguments

`formula`	an object of class `` `formula` ``: a symbolic description of the model to be fitted (of type `y ~ x` or `y ~ x | z`).
`data`	an optional data frame, list, or object coercible to a data frame through `base::as.data.frame` containing the variables in the model. If not found in data, the variables in formula are taken from the environment from which the function flexreg is called.
`type`	a character specifying the type of regression model. Current options are the flexible beta regression model `"FB"` (default), the variance inflated beta `"VIB"`, and the beta one `"Beta"`.
`link.mu`	a character specifying the link function for the mean model (mu). Currently, `"logit"` (default), `"probit"`, `"cloglog"`, and `"loglog"` are supported.
`prior.beta`	a character specifying the prior distribution for the `beta` regression coefficients of the mean model. Currently, `"normal"` (default) and `"cauchy"` are supported.

`hyperparam.beta`
: a numeric (vector of length 1) specifying the hyperprior parameter for the prior distribution of `beta` regression coefficients. A value of 100 is suggested if the prior is `"normal"`, 2.5 if `"cauchy"`.

`link.phi`
: a character specifying the link function for the precision model (phi). Currently, `"identity"` (default), `"log"` and `"sqrt"` are supported.

`prior.phi`
: a character specifying the prior distribution for precision parameter `phi` if `link.phi = "identity"`. Currently, `"gamma"` (default) and `"unif"` are supported.

`hyperparam.phi`
: a numeric (vector of length 1) specifying the hyperprior parameter for the prior distribution of `phi`. A value of 0.001 is suggested if the prior is `"gamma"`. If the prior is `"uniform"` the hyperparameter must be specified to define the upper limit of the support of `phi`.

`prior.psi`
: a character specifying the prior distribution for `psi` regression coefficients of the precision model (not supported if link.phi is `"identity"`). Currently, `"normal"` (default) and `"cauchy"` are supported.

`hyperparam.psi`
: a numeric (vector of length 1) specifying the hyperprior parameter for the prior distribution of `psi` regression coefficients. A value of 100 is suggested if the prior is `"normal"`, 2.5 if `"cauchy"`.

`n.iter`
: a positive integer specifying the number of iterations for each chain (including warmup). The default is 5000.

`burnin.perc`
: the percentage of iterations (out of `n.iter`) per chain to discard.

`n.chain`
: a positive integer specifying the number of Markov chains. The default is 1.

`thin`
: a positive integer specifying the period for saving samples. The default is 1.

`verbose`
: `TRUE` (default) or `FALSE`: flag indicating whether to print intermediate output.

`...`
: additional arguments for `rstan::sampling`.

Details

Let μ be the mean of a random variable Y whose distribution can be specified in the `type` argument. Then the `flexreg` function links the parameter μ to a linear predictor through a function $g(\cdot)$ specified in `link.mu`:

$$g(\mu_i) = x_i^t \boldsymbol{\beta},$$

where $\boldsymbol{\beta}$ is the vector of regression coefficients for the mean model. By default, the precision parameter ϕ is assumed to be constant. It is possible to extend the model by linking ϕ to an additional (possibly overlapping) set of covariates through a proper link function $q(\cdot)$ specified in the `link.phi` argument:

$$q(\phi_i) = z_i^t \boldsymbol{\psi},$$

where $\boldsymbol{\psi}$ is the vector of regression coefficients for the precision model. In `flexreg`, the regression model for the mean and, where appropriate, for the precision parameter can be specified in the `formula` argument with a formula of type $y \sim x_1 + x_2 | z_1 + z_2$ where covariates on the left of ("|") are included in the regression model for the mean and covariates on the right of ("|") are included in the regression model for the precision.

If the second part is omitted, i.e., $y \sim x_1 + x_2$, the precision is assumed constant for each observation.

Value

The `flexreg` function returns an object of class `` `flexreg` ``, i.e. a list with the following elements:

`call`	the function call.
`formula`	the original formula.
`link.mu`	a character specifing the link function in the mean model.
`link.phi`	a character specifing the link function in the precision model.
`model`	an object of class `` `stanfit` `` containing the fitted model.
`response`	the response variable, assuming values in (0, 1).
`design.X`	the design matrix for the mean model.
`design.Z`	the design matrix for the precision model (if defined).

References

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018) A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

Di Brisco, A. M., Migliorati, S., Ongaro, A. (2020) Robustness against outliers: A new variance inflated regression model for proportions. Statistical Modelling, **20**(3), 274–309. doi:10.1177/1471082X18821213

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. Journal of Applied Statistics, **31**(7), 799–815. doi:10.1080/0266476042000214501

Examples

```
data("Reading")
FB <- flexreg(accuracy ~ iq, Reading, type="FB", n.iter=1000)
```

`plot.flexreg` *Plot method for flexreg Objects*

Description

Method for plotting regression curves for the mean from fitted regression model objects of class `` `flexreg` ``.

Usage

```
## S3 method for class 'flexreg'
plot(x, name.x, additional.cov.default = NA, ...)
```

Arguments

x
: an object of class flexreg, usually the result of `flexreg`.

name.x
: a character containing the name of the covariate to be plotted on the x-axis of the scatterplot.

additional.cov.default
: a list of additional covariates to be set as default.

...
: additional arguments. Currently not used.

Details

The function produces a scatterplot of the covariate specified in `name.x` and the response `y`. Any other variable involved in the formula must be set to a default through the `additional.cov.default` argument. If the regression model is of `FB` type the function returns a scatterplot with three curves, one corresponding to the overall mean and two corresponding to the component means of the FB distribution, i.e., λ_1 and λ_1.

Examples

```
data("Reading")
FB <- flexreg(accuracy ~ iq+dyslexia, Reading, n.iter=800)
plot(FB, name.x="iq", additional.cov.default = list("dyslexia"=1))
```

`plot.flexreg_postpred` *Posterior Predictives Plot*

Description

A method for plotting the simulated posterior predictive distribution. It takes an object of class `` `flexreg_postpred` `` (i.e., the output of `posterior_predictive.flexreg`) and plots the posterior predictive interval for each statistical unit. Additionally, the mean of the posterior predictives and the value of the response variable can be added.

Usage

```
## S3 method for class 'flexreg_postpred'
plot(x, prob = 0.9, p_mean = F, response = NULL, ...)
```

Arguments

x
: an object of class `` `flexreg_postpred` `` containing the simulated posterior predictives.

prob
: the interval probability for the posterior predictives (default is 0.9).

p_mean
: a logical value indicating whether the posterior predictives' mean should be plotted.

`response`	a numerical vector containing the response variable to be added to the plot. if `NULL`, response values are omitted.
`...`	additional arguments. Currently not used.

Examples

```
data("Reading")
dataset <- Reading
FB <- flexreg(accuracy ~ iq, dataset, n.iter=1000)
pp <- posterior_predict(FB)
plot(pp)
```

`posterior_predict.flexreg`
Posterior Predictive

Description

The function takes an object of class `` `flexreg` `` and generates values from the posterior predictive distribution.

Usage

```
## S3 method for class 'flexreg'
posterior_predict(model, newdata = NULL)
```

Arguments

`model`	an object of class `` `flexreg` ``
`newdata`	an optional data frame containing variables with which to predict. If omitted, the fitted values are used.

Details

The function generates values from the posterior predictive distribution, which is the distribution of a future outcome given the observed data.

Value

An object of class `` `flexreg_postpred` `` containing a matrix with the simulated posterior predictions. Each column refers to a statistical unit to predict.

References

Gelman, A.; Carlin, J. B.; Stern, H. S. and Rubin, D. B. (2014), Bayesian Data Analysis, 3th edition. Chapman and Hall/CRC. doi:10.1201/b16018

Examples

```
data("Reading")
dataset <- Reading
FB <- flexreg(accuracy ~ iq, dataset, n.iter=1000)
pp <- posterior_predict(FB)
plot(pp)
```

---

predict.flexreg *Prediction Method for flexreg Objects*

Description

Method that computes various types of prediction from objects of class `` `flexreg` ``. If the model type is FB and cluster = T, the function returns also cluster means.

Usage

```
## S3 method for class 'flexreg'
predict(
  object,
  newdata = NULL,
  cluster = F,
  type = "response",
  estimate = "mean",
  q = NULL,
  ...
)
```

Arguments

object	an object of class `` `flexreg` ``, usually the result of flexreg.
newdata	an optional data frame containing variables with which to predict. If omitted, the fitted values are used.
cluster	if the model is "FB", cluster = T returns the cluster means. By default, cluster = F.
type	a character indicating the type of predictions. Available options are the fitted means of response (response), the linear predictor (link), the fitted precision parameter phi (precision), and the fitted variances of the response (variance).
estimate	the type of estimate: mean (default), median or quantile.
q	if estimate is quantile, numeric value of probability in (0, 1).
...	additional arguments. Currently not used.

Examples

```
{
data("Reading")
FB <- flexreg(accuracy ~ iq, Reading, type="FB", n.iter=1000)
predict(FB, type="response", cluster=TRUE)
}
```

R2_bayes *Bayesian R-squared for flexreg Objects*

Description

Bayesian version of R-squared for flexible regression models for proportions

Usage

```
R2_bayes(model)
```

Arguments

`model`	an object of class `` `flexreg` ``, usually the result of `flexreg`.

Details

The function provides a Bayesian version of the R-squared measure, defined as the variance of the predicted values divided by itself plus the expected variance of the errors.

References

Andrew Gelman, Ben Goodrich, Jonah Gabry & Aki Vehtari (2019) R-squared for Bayesian Regression Models, The American Statistician, 73:3, 307-309, DOI: 10.1080/00031305.2018.1549100

Examples

```
data("Reading")
FB <- flexreg(accuracy ~ iq, Reading, type="FB", n.iter=1000)
hist(R2_bayes(FB))
```

`rBeta_mu` *Random generator from the beta distribution*

Description

The function randomly generates values from the beta distribution with a mean-precision parameterization.

Usage

```
rBeta_mu(n, mu, phi)
```

Arguments

`n`	the number of observations. If length(n) > 1, the length is taken to be the number required.
`mu`	the mean parameter of the beta distribution. It must lie in (0, 1).
`phi`	the precision parameter of the Beta distribution. It must be a positive real value.

Value

A vector of length `n`.

References

Ferrari, S.L.P., and Cribari-Neto, F. (2004). Beta Regression for Modeling Rates and Proportions. Journal of Applied Statistics, **31**(7), 799–815. doi:10.1080/0266476042000214501

Examples

```
rBeta_mu(100, mu = 0.5, phi = 30)
```

`Reading` *Reading Skills data*

Description

Data for assessing the contribution of non-verbal IQ to children's reading skills in dyslexic and non-dyslexic children.

Format

A data frame containing 44 observations on 3 variables.

`accuracy` a reading score.

`dyslexia` a factor indicating wheter the child is dyslexic.

`iq` a quantitative measure of the children's non verbal abilities.

Details

The data were originally analyzed by Pammer and Kevan (2004) and successively used by Smithson and Verkuilen (2006) and by Migliorati et al. (2018).

Source

betareg.

References

Smithson, M., and Verkuilen, J. (2006). A Better Lemon Squeezer? Maximum-Likelihood Regression with Beta-Distributed Dependent Variables. Psychological Methods, **11**(7), 54–71.

Cribari-Neto, F., and Zeileis, A. (2010). Beta Regression in R. Journal of Statistical Software, 34(2), 1–24.

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018). A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

`residuals.flexreg` *Residuals Method for flexreg Objects*

Description

Method that computes various types of residuals from objects of class `` `flexreg` ``. If the model type is `FB` and `cluster = T`, the method returns also residuals with respect to cluster means.

Usage

```
## S3 method for class 'flexreg'
residuals(
  object,
  type = "raw",
  cluster = FALSE,
  estimate = "mean",
  q = NULL,
  ...
)
```

Arguments

`object`	an object of class `` `flexreg` ``, usually the result of `flexreg`.
`type`	a character indicating type of residuals (`raw` or `standardized`).
`cluster`	a logical. If the model is `"FB"`, `cluster=T` returns the cluster means. Default `cluster = F`.
`estimate`	the type of estimate: `mean` (default), `median` or `quantile`.
`q`	if estimate is `quantile`, a numeric value of probability in (0, 1).
`...`	additional arguments. Currently not used.

Details

Raw residuals are defined as $y_i - \hat{\mu}_i$ for $i = 1, \dots, n$. The values y_i for $i, \dots, n$ are referred to the observed response variable and they are specified on the left-hand side of `formula` in the `flexreg` function. $\hat{\mu}_i$ for $i = 1, \dots, n$ is the predicted mean. It can be computed separately through the `predict` function by setting `type=response`. Standardized residuals are defined as $\frac{y_i - \hat{\mu}_i}{V\hat{a}r(y_i)}$ where $V\hat{a}r(y_i)$ is the variance of the dependent variable evaluated at the posterior means (default, otherwise quantile of order q) of the parameters. If the model is `"FB"` and `cluster=T`, the cluster residuals are computed as the difference between the observed response variable and the cluster means $\hat{\lambda}_{1i}$ and $\hat{\lambda}_{2i}$ for $i = 1, \dots, n$.

References

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018) A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

Examples

```
{
data("Reading")
FB <- flexreg(accuracy ~ iq, Reading, type="FB", n.iter=1000)
residuals(FB, type="raw", cluster=TRUE)
}
```

---

rFB — *Random generator from the flexible beta distribution*

Description

The function randomly generates values from the flexible beta distribution.

Usage

```
rFB(n, mu, phi, p, w)
```

Arguments

`n`	the number of observations. If length(n) > 1, the length is taken to be the number required.
`mu`	the mean parameter of the flexible beta distribution. It must lie in (0, 1).
`phi`	the precision parameter of the Flexible Beta distribution. It must be a positive real value.
`p`	the mixing weight. It must lie in (0, 1).
`w`	the normalized distance among clusters. It must lie in (0, 1).

Value

A vector of length `n`.

References

Migliorati, S., Di Brisco, A. M., Ongaro, A. (2018). A New Regression Model for Bounded Responses. Bayesian Analysis, **13**(3), 845–872. doi:10.1214/17-BA1079

Examples

```
rFB(100,0.5,30,0.3,0.6)
```

`rVIB` *Random generation from the variance-inflated beta distribution*

Description

The function randomly generates values from the variance-inflated beta distribution.

Usage

```
rVIB(n, mu, phi, p, k)
```

Arguments

`n`	the number of observations. If length(n) > 1, the length is taken to be the number required.
`mu`	the mean parameter of the Variance-Inflated distribution. It must lie in (0, 1).
`phi`	the precision parameter of the Variance-Inflated distribution. It must be a positive real value.
`p`	the mixing weight. It must lie in (0, 1).
`k`	the extent of the variance inflation. It must lie in (0, 1).

Value

A vector of length `n`.

References

Di Brisco, A. M., Migliorati, S., Ongaro, A. (2020) Robustness against outliers: A new variance inflated regression model for proportions. Statistical Modelling, **20**(3), 274–309. doi:10.1177/1471082X18821213

Examples

```
rVIB(100,0.5,30,0.3,0.6)
```

`Stress` *Stress and anxiety data*

Description

Data for assessing the dependency between stress and anxiety in nonclinical women in Townsville, Queensland, Australia.

Format

A data frame containing 166 observations on 2 variables.

`stress` defined as rate.

`anxiety` defined as rate.

Details

Both variables are rates obtained as linear transformations from the Depression Anxiety Stress Scales which range from 0 to 42 (Lovibond & Lovibond, 1995). Additional details can be found in Example 2 from Smithson and Verkuilen (2006).

Source

Example 2 from Smithson and Verkuilen (2006).

References

Smithson, M., and Verkuilen, J. (2006). A Better Lemon Squeezer? Maximum-Likelihood Regression with Beta-Distributed Dependent Variables. Psychological Methods, 11(7), 54–71.

Lovibond, P. F., & Lovibond, S. H. (1995). The structure of negative emotional states: Comparison of the Depression Anxiety Stress Scales (DASS) with the Beck Depression and Anxiety Inventories. Behaviour research and therapy, 33(3), 335-343.

summary.flexreg *Methods for flexreg Objects*

Description

Methods for extracting information from fitted regression model objects of class `` `flexreg` ``.

Usage

```
## S3 method for class 'flexreg'
summary(object, ..., digits = 4)

## S3 method for class 'summary.flexreg'
print(x, ...)

## S3 method for class 'flexreg'
coef(object, ...)
```

Arguments

object	an object of class `` `flexreg` ``, usually the result of `flexreg`.
...	additional arguments. Currently not used.
digits	an integer indicating the number of decimal places. Default equal to 4.
x	an object of class `` `flexreg` ``, usually the result of `flexreg`.

Details

The `summary.flexreg` method summarizes the results of `flexreg`, adding also information from the functions `residuals.flexreg` and `WAIC`. The `summary.flexreg` method returns an object of class `` `summary.flexreg` `` containing the relevant summary statistics which can subsequently be printed using the associated `print` method.

Examples

```
data("Reading")
dataset <- Reading
FB <- flexreg(accuracy ~ iq, dataset, n.iter = 1000)
summary(FB)
```

WAIC *WAIC and LOO*

Description

The function computes widely applicable information criterion (WAIC) and efficient approximate leave-one-out cross-validation (LOO) from fitted regression model objects of class `` `flexreg` ``.

Usage

```
WAIC(model, ...)

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

Arguments

model	an object (or a list of objects) of class `` `flexreg` ``.
...	additional arguments.
x	an object of class `` `WAIC.flexreg` ``, usually the result of `WAIC`.

Details

This function takes advantage of the `loo` package to compute the widely applicable information criterion (WAIC) and leave-one-out cross-validation (LOO) for object of class `` `flexreg` ``. If two or more objects of class `` `flexreg` `` are provided by a list, the function compares them

References

Vehtari, A., Gelman, A., and Gabry, J. (2017a). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing. **27**(5), 1413–1432. doi:10.1007/s11222-016-9696-4

Examples

```
{
data("Reading")
FB <- flexreg(accuracy ~ iq, Reading, type="FB", n.iter=1000)
WAIC(FB)
}
```

Index