Package ‘rmsb’

August 3, 2020

Title Bayesian Regression Modeling Strategies

Version 0.0.1

Date 2020-07-25

Description A Bayesian companion to the 'rms' package, 'rmsb' provides Bayesian model fitting, post-fit estimation, and graphics. It implements Bayesian regression models whose fit objects can be processed by 'rms' functions such as 'contrast()', 'summary()', 'Predict()', 'nomogram()', and 'latex()'. The fitting function currently implemented in the package is 'blrm()' for Bayesian logistic binary and ordinal regression with optional clustering, censoring, and departures from the proportional odds assumption using the partial proportional odds model of Peterson and Harrell (1990) <doi:10.2307/2347760>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

URL https://hbiostat.org/R/rmsb

RoxygenNote 7.1.1

Biarch true

Depends R (>= 3.4.0), rms (>= 6.0.1)

Imports methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.1.1), Hmisc (>= 4.3-0), survival (>= 3.1-12), ggplot2, MASS, cluster, digest, knitr, loo

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

Suggests mice

SystemRequirements GNU make

NeedsCompilation yes

Author Frank Harrell [aut, cre] (<https://orcid.org/0000-0002-8271-5493>),
Ben Goodrich [ctb] (contributed Stan code),
Ben Bolker [ctb] (wrote original code that is folded into the pdensityContour function),
Doug Bates [ctb] (write original code for highest posterior density interval that is folded into the HPDint function)

Maintainer Frank Harrell `<fh@fharrell.com>`

Repository CRAN

Date/Publication 2020-08-03 10:20:07 UTC

R topics documented:

rmsb-package 2
as.data.frame.Ocens 3
blrm 4
blrmStats 8
coef.rmsb 10
compareBmods 11
distSym 11
ExProb.blrm 12
getParamCoef 13
HPDint 14
Mean.blrm 14
Ocens 15
pdensityContour 16
plot.PostF 17
plot.rmsb 18
PostF 19
predict.blrm 21
print.blrm 22
print.blrmStats 23
print.predict.blrm 24
print.rmsb 25
Quantile.blrm 26
selectedQr 27
stackMI 28
stanDx 29
stanDxplot 30
stanGet 31
tauFetch 32
vcov.rmsb 32

Index **34**

`rmsb-package` *The 'rmsb' package.*

Description

Regression Modeling Strategies Bayesian

The **rmsb** package is an appendage to the **rms** package that implements Bayesian regression models whose fit objects can be processed by **rms** functions such as `contrast`, `summary`, `Predict`, `nomogram`, and `latex`. The fitting function currently implemented in the package is `blrm` for Bayesian logistic binary and ordinal regression with optional clustering, censoring, and departures from the proportional odds assumption using the partial proportional odds model of Peterson and Harrell (1990).

References

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

See Also

- `http://hbiostat.org/R/rmsb` for the package's main web page
- `http://hbiostat.org/R/rmsb/blrm.html` for a vignette with many examples of using the `blrm` function

---

`as.data.frame.Ocens` *as.data.frame.Ocens*

Description

Convert `Ocens` Object to Data Frame to Facilitate Subset

Usage

```
## S3 method for class 'Ocens'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	an `Ocens` object
`row.names`	optional vector of row names
`optional`	set to `TRUE` if needed
`...`	ignored

Details

Converts an `Ocens` object to a data frame so that subsetting will preserve all needed attributes

Value

data frame containing a 2-column integer matrix with attributes

Author(s)

Frank Harrell

`blrm` *blrm*

Description

Bayesian Binary and Ordinal Logistic Regression

Usage

```
blrm(
  formula,
  ppo = NULL,
  cppo = NULL,
  keepsep = NULL,
  data = environment(formula),
  subset,
  na.action = na.delete,
  priorsd = rep(100, p),
  priorsdppo = rep(100, pppo),
  conc = 1/(0.8 + 0.35 * max(k, 3)),
  psigma = 1,
  rsdmean = if (psigma == 1) 0 else 1,
  rsdsd = 1,
  normcppo = TRUE,
  iter = 2000,
  chains = 4,
  refresh = 0,
  progress = if (refresh > 0) "stan-progress.txt" else "",
  x = TRUE,
  y = TRUE,
  loo = n <= 1000,
  ppairs = NULL,
  method = c("both", "sampling", "optimizing"),
  inito = if (length(ppo)) 0 else "random",
  inits = inito,
  standata = FALSE,
  file = NULL,
  debug = FALSE,
  ...
)
```

Arguments

`formula`	a R formula object that can use `rms` package enhancements such as the restricted interaction operator
`ppo`	formula specifying the model predictors for which proportional odds is not assumed
`cppo`	a function that if present causes a constrained partial PO model to be fit. The function specifies the values in the Gamma vector in Peterson and Harrell (1990) equation (6). To make posterior sampling better behaved, the function should be scaled and centered. This is done by wrapping `cppo` in a function that scales the `cppo` result before return the vector value. See the `normcco` argument for how to prevent this. The default normalization is based on the mean and standard deviation of the function values over the distribution of observed Y. For getting predicted values and estimates post-`blrm()`, `cppo` must not reference any functions that are not available at such later times.
`keepsep`	a single character string containing a regular expression applied to design matrix column names, specifying which columns are not to be QR-orthonormalized, so that priors for those columns apply to the original parameters. This is useful for treatment and treatment interaction terms. For example `keepsep='treat'` will keep separate all design matrix columns containing `'treat'` in their names. Some characters such as the caret used in polynomial regression terms will need to be escaped by a double backslash.
`data`	a data frame; defaults to using objects from the calling environment
`subset`	a logical vector or integer subscript vector specifying which subset of data whould be used
`na.action`	default is `na.delete` to remove missings and report on them
`priorsd`	vector of prior standard deviations. If the vector is shorter than the number of model parameters, it will be repeated until the length equals the number of parametertimes.
`priorsdppo`	vector of prior standard deviations for non-proportional odds parameters. As with `priorsd` the last element is the only one for which the SD corresponds to the original data scale.
`conc`	the Dirichlet distribution concentration parameter for the prior distribution of cell probabilities at covariate means. The default is the reciprocal of 0.8 + 0.35 max(k, 3) where k is the number of Y categories. The default is chosen to make the posterior mean of the intercepts more closely match the MLE. For optimizing, the concentration parameter is always 1.0 to obtain results very close to the MLE for providing the posterior mode.
`psigma`	defaults to 1 for a half-t distribution with 4 d.f., location parameter `rsdmean` and scale parameter `rsdsd`. Set `psigma=2` to use the exponential distribution.
`rsdmean`	the assumed mean of the prior distribution of the standard deviation of random effects. When `psigma=2` this is the mean of an exponential distribution and defaults to 1. When `psigma=1` this is the mean of the half-t distribution and defaults to zero.
`rsdsd`	applies only to `psigma=1` and is the scale parameter for the half t distribution for the SD of random effects, defaulting to 1.

`normcppo`	set to `FALSE` to leave the cppo function as-is without automatically centering and scaling the result
`iter`	number of posterior samples per chain for `rstan::sampling()` to run
`chains`	number of separate chains to run
`refresh`	see `rstan::sampling()`. The default is 0, indicating that no progress notes are output. If `refresh` > `0` and `progress` is not `''`, progress output will be appended to file `progress`. The default file name is `'stan-progress.txt'`.
`progress`	see `refresh`. Defaults to `''` if `refresh` = `0`. Note: If running interactively but not under RStudio, `rstan` will open a browser window for monitoring progress.
`x`	set to `FALSE` to not store the design matrix in the fit. `x=TRUE` is needed if running `blrmStats` for example.
`y`	set to `FALSE` to not store the response variable in the fit
`loo`	set to `FALSE` to not run `loo` and store its result as object `loo` in the returned object. `loo` defaults to `FALSE` if the sample size is greater than 1000, as `loo` requires the per-observation likelihood components, which creates a matrix N times the number of posterior draws.
`ppairs`	set to a file name to run `rstan pairs` and store the resulting png plot there. Set to `TRUE` instead to directly plot these diagnostics. The default is not to run `pairs`.
`method`	set to `'optimizing'` to run the Stan optimizer and not do posterior sampling, `'both'` (the default) to run both the optimizer and posterior sampling, or `'sampling'` to run only the posterior sampling and not compute posterior modes. Running `optimizing` is a way to obtain maximum likelihood estimates and allows one to quickly study the effect of changing the prior distributions. When `method='optimizing'` is used the result returned is not a standard `blrm()` object but is instead the parameter estimates, -2 log likelihood, and optionally the Hession matrix (if you specify `hessian=TRUE` in ...). When `method='both'` is used, `rstan::sampling()` and `rstan::optimizing()` are both run, and parameter estimates (posterior modes) from `optimizing` are stored in a matrix `param` in the fit object, which also contains the posterior means and medians, and other results from `optimizing` are stored in object `opt` in the `blrm()` fit object. When random effects are present, `method` is automatically set to `'sampling'` as maximum likelihood estimates without marginalizing over the random effects do not make sense.
`inito`	intial value for optimization. The default is the `rstan` default `'random'`. Frequently specifying `init=0` will benefit when the number of distinct Y categories grows or when using ppo hence 0 is the default for that.
`inits`	initial value for sampling, defaults to `inito`
`standata`	set to `TRUE` to return the Stan data list and not run the model
`file`	a file name for a `saveRDS`-created file containing or to contain the saved fit object. If `file` is specified and the file does not exist, it will be created right before the fit object is returned, less the large `rstan` object. If the file already exists, its stored `md5` hash string `datahash` fit object component is retrieved and compared to that of the current `rstan` inputs. If the data to be sent to `rstan`, the priors, and all sampling and optimization options and stan code are identical, the previously stored fit object is immediately returned and no new calculatons are done.

`debug`	set to `TRUE` to output timing and progress information to /tmp/debug.txt
`...`	passed to `rstan::optimizing()`. The `seed` parameter is a popular example.

Details

Uses `rstan` with pre-compiled Stan code to get posterior draws of parameters from a binary logistic or proportional odds semiparametric ordinal logistic model. The Stan code internally using the qr decompositon on the design matrix so that highly collinear columns of the matrix do not hinder the posterior sampling. The parameters are transformed back to the original scale before returning results to R. Design matrix columns are centered before running Stan, so Stan diagnostic output will have the intercept terms shifted but the results of `blrm()` for intercepts are for the original uncentered data. The only prior distributions for regression betas are normal with mean zero, and the vector of prior standard deviations is given in `priorsd`. These priors are for the qr-projected design matrix elements, except that the very last element is not changed. So if one has a single non-interactive linear or binary variable for which a skeptical prior is designed, put that variable last in the model.

The partial proportional odds model of Peterson and Harrell (1990) is implemented, and is invoked when the user specifies a second model formula as the `ppo` argument. This formula has no left-hand-side variable, and has right-side variables that are a subset of those in `formula` specifying for which predictors the proportional odds assumption is relaxed.

The Peterson and Harrell (1990) constrained partial proportional odds is also implemented, and is usually preferred to the above unconstrained PPO model as it adds a vector of coefficients instead of a matrix of coefficients. In the constrained PPO model the user provides a function `cppo` that computes a score for all observed values of the dependent variable. For example with a discrete ordinal outcome `cppo` may return a value of 1.0 for a specific value of Y and zero otherwise. That will result in a departure from the proportional odds assumption for just that one level of Y. The value returned by `cppo` at the lowest Y value is never used in any case.

`blrm()` also handles single-level hierarchical random effects models for the case when there are repeated measurements per subject which are reflected as random intercepts, and a different experimental model that allows for AR(1) serial correlation within subject. For both setups, a `cluster` term in the model signals the existence of subject-specific random effects.

See `https://hbiostat.org/R/rms/blrm.html` for multiple examples with results.

Value

an `rms` fit object of class `blrm`, `rmsb`, `rms` that also contains `rstan` results under the name `rstan`. In the `rstan` results, which are also used to produce diagnostics, the intercepts are shifted because of the centering of columns of the design matrix done by `blrm()`. With `method='optimizing'` a class-less list is return with these elements: `coefficients` (MLEs), `beta` (non-intercept parameters on the QR decomposition scale), `deviance` (-2 log likelihood), `return_code` (see `rstan::optimizing()`), and, if you specified `hessian=TRUE` to `blrm()`, the Hessian matrix. To learn about the scaling of orthogonalized QR design matrix columns, look at the `xqrsd` object in the returned object. This is the vector of SDs for all the columns of the transformed matrix. Those kept out by the `keepsep` argument will have their original SDs.

Author(s)

Frank Harrell and Ben Goodrich

See Also

`print.blrm()`, `blrmStats()`, `stanDx()`, `stanGet()`, `coef.rmsb()`, `vcov.rmsb()`, `print.rmsb()`, `coef.rmsb()`

Examples

```
## Not run:
  getHdata(Titanic3)
  dd <- datadist(titanic3); options(datadist='dd')
  f <- blrm(survived ~ (rcs(age, 5) + sex + pclass)^2, data=titanic3)
  f                    # model summary using print.blrm
  coef(f)              # compute posterior mean parameter values
  coef(f, 'median')    # compute posterior median values
  stanDx(f)            # print basic Stan diagnostics
  s <- stanGet(f)      # extract rstan object from fit
  plot(s, pars=f$betas)        # Stan posteriors for beta parameters
  stanDxplot(s)        # Stan diagnostic plots by chain
  blrmStats(f)         # more details about predictive accuracy measures
  ggplot(Predict(...))   # standard rms output
  summary(f, ...)      # invokes summary.rms
  contrast(f, ...)     # contrast.rms computes HPD intervals
  plot(nomogram(f, ...)) # plot nomogram using posterior mean parameters

  # Fit a random effects model to handle multiple observations per
  # subject ID
  f <- blrm(outcome ~ rcs(age, 5) + sex + cluster(id), data=mydata)

## End(Not run)
```

`blrmStats` *blrmStats*

Description

Compute Indexes of Predictive Accuracy and Their Uncertainties

Usage

```
blrmStats(fit, ns = 400, prob = 0.95, pl = FALSE, dist = c("density", "hist"))
```

Arguments

`fit`	an object produced by `blrm()`
`ns`	number of posterior draws to use in the calculations (default is 400)
`prob`	HPD interval probability (default is 0.95)
`pl`	set to `TRUE` to plot the posterior densities using base graphics
`dist`	if `pl` is `TRUE` specifies whether to plot the density estimate (the default) or a histogram

Details

For a binary or ordinal logistic regression fit from `blrm()`, computes several indexes of predictive accuracy along with highest posterior density intervals for them. Optionally plots their posterior densities. When there are more than two levels of the outcome variable, computes Somers' Dxy and c-index on a random sample of 10,000 observations.

Value

list of class `blrmStats` whose most important element is `Stats`. The indexes computed are defined below, with gp, B, EV, and vp computed using the intercept corresponding to the median value of Y. See `https://fharrell.com/post/addvalue` for more information.

"Dxy" Somers' Dxy rank correlation between predicted and observed. The concordance probability (c-index; AUROC in the binary Y case) may be obtained from the relationship Dxy=2(c-0.5).

"g" Gini's mean difference: the average absolute difference over all pairs of linear predictor values

"gp" Gini's mean difference on the predicted probability scale

"B" Brier score

"EV" explained variation

"v" variance of linear predictor

"vp" variable of estimated probabilities

Author(s)

Frank Harrell

See Also

`Hmisc::rcorr.cens()`

Examples

```
## Not run: 
  f <- blrm(...)
  blrmStats(f, pl=TRUE)   # print and plot

## End(Not run)
```

`coef.rmsb` *coef.rmsb*

Description

Extract Bayesian Summary of Coefficients

Usage

```
## S3 method for class 'rmsb'
coef(object, stat = c("mean", "median", "mode"), ...)
```

Arguments

`object`	an object created by an `rms` package Bayesian fitting function
`stat`	name of measure of posterior distribution central tendency to compute
`...`	ignored

Details

Computes either the posterior mean (default), posterior median, or posterior mode of the parameters in an `rms` Bayesian regression model

Value

a vector of intercepts and regression coefficients

Author(s)

Frank Harrell

Examples

```
## Not run:
  f <- blrm(...)
  coef(f, stat='mode')

## End(Not run)
```

`compareBmods` *compareBmods*

Description

Compare Bayesian Model Fits

Usage

```
compareBmods(..., method = "stacking", r_eff_list = NULL)
```

Arguments

`...`	a series of model fits
`method`	see `loo::loo_model_weights()`
`r_eff_list`	see `loo::loo_model_weights()`

Details

Uses `loo::loo_model_weights()` to compare a series of models such as those created with `blrm()`

Value

a `loo::loo_model_weights()` object

Author(s)

Frank Harrell

`distSym` *distSym*

Description

Distribution Symmetry Measure

Usage

```
distSym(x, prob = 0.9, na.rm = FALSE)
```

Arguments

`x`	a numeric vector representing a sample from a continuous distribution
`prob`	quantile interval coverage
`na.rm`	set to `TRUE` to remove `NA`s before proceeding.

Details

From a sample from a distribution computes a symmetry measure. By default it is the gap between the mean and the 0.95 quantile divided by the gap between the 0.05 quantile and the mean.

Value

a scalar with a value of 1.0 indicating symmetry

Author(s)

Frank Harrell

`ExProb.blrm` *ExProb.blrm*

Description

Function Generator for Exceedance Probabilities for `blrm()`

Usage

```
## S3 method for class 'blrm'
ExProb(object, posterior.summary = c("mean", "median"), ...)
```

Arguments

`object`	a `blrm()` fit
`posterior.summary`	defaults to posterior mean; may also specify `"median"`. Must be consistent with the summary used when creating `lp`.
`...`	ignored

Details

For a `blrm()` object generates a function for computing the estimates of the function Prob(Y>=y) given one or more values of the linear predictor using the reference (median) intercept. This function can optionally be evaluated at only a set of user-specified `y` values, otherwise a right-step function is returned. There is a plot method for plotting the step functions, and if more than one linear predictor was evaluated multiple step functions are drawn. `ExProb` is especially useful for `nomogram()`. The linear predictor argument is a posterior summarized linear predictor lp (e.g. using posterior mean of intercepts and slopes) computed at the reference intercept. `lptau` must be provided when call the created function if the model is a partial proportional odds model.

Value

an R function

Author(s)

Frank Harrell

`getParamCoef` *getParamCoef*

Description

Get a Bayesian Parameter Vector Summary

Usage

```
getParamCoef(
  fit,
  posterior.summary = c("mean", "median", "mode"),
  what = c("both", "betas", "taus")
)
```

Arguments

`fit`	a Bayesian model fit from `rmsb`
`posterior.summary`	which summary statistic (Bayesian point estimate) to fetch
`what`	specifies which coefficients to include. Default is all. Specify `what="betas"` to include only intercepts and betas if the model is a partial proportional odds model (i.e.,, exclude the tau parameters). Specify `what="taus"` to include only the tau parameters.

Details

Retrieves posterior mean, median, or mode (if available)

Value

vector of regression coefficients

Author(s)

Frank Harrell

`HPDint` *HPDint*

Description

Highest Posterior Density Interval

Usage

```
HPDint(x, prob = 0.95)
```

Arguments

`x`	a vector of posterior draws
`prob`	desired probability coverage

Details

Adapts code from `coda::HPDinterval()` to compute a highest posterior density interval from posterior samples for a single parameter. Quoting from the coda help file, for each parameter the interval is constructed from the empirical cdf of the sample as the shortest interval for which the difference in the ecdf values of the endpoints is the nominal probability. Assuming that the distribution is not severely multimodal, this is the HPD interval.

Value

a 2-vector with elements `Lower` and `Upper`

Author(s)

Douglas Bates and Frank Harrell

`Mean.blrm` *Mean.blrm*

Description

Function Generator for Mean Y for `blrm()`

Usage

```
## S3 method for class 'blrm'
Mean(object, codes = FALSE, posterior.summary = c("mean", "median"), ...)
```

Arguments

`object`	a `blrm()` fit
`codes`	if `TRUE`, use the integer codes $1, 2, \ldots, k$ for the k-level response in computing the predicted mean response.
`posterior.summary`	defaults to posterior mean; may also specify `"median"`. Must be consistent with the summary used when creating `lp`.
`...`	ignored

Details

Creates a function to turn a posterior summarized linear predictor lp (e.g. using posterior mean of intercepts and slopes) computed at the reference intercept into e.g. an estimate of mean Y using the posterior mean of all the intercept. `lptau` must be provided when call the created function if the model is a partial proportional odds model.

Value

an R function

Author(s)

Frank Harrell

`Ocens` *Ocens*

Description

Censored Ordinal Variable

Usage

```
Ocens(a, b = a)
```

Arguments

`a`	vector representing a `factor`, numeric, or alphabetically ordered character strings
`b`	like `a`. If omitted, it copies `a`, representing nothing but uncensored values

Details

Creates a 2-column integer matrix that handles left- right- and interval-censored ordinal or continuous values for use in `blrm()`. A pair of values [a, b] represents an interval-censored value known to be in the interval [a, b] inclusive of `a` and `b`. It is assumed that all distinct values are observed as uncensored for at least one observation. When both input variables are `factors` it is assume that the one with the higher number of levels is the one that correctly specifies the order of levels, and that the other variable does not contain any additional levels. If the variables are not `factors` it is assumed their original values provide the orderings. Since all values that form the left or right endpoints of an interval censored value must be represented in the data, a left-censored point is is coded as `a=1` and a right-censored point is coded as `b` equal to the maximum observed value. If the maximum observed value is not really the maximum possible value, everything still works except that predictions involving values above the highest observed value cannot be made. As with most censored-data methods, `blrm()` assumes that censoring is independent of the response variable values that would have been measured had censoring not occurred.

Value

a 2-column integer matrix of class `"Ocens"` with an attribute `levels` (ordered). When the original variables were `factors`, these are factor levels, otherwise are numerically or alphabetically sorted distinct (over `a` and `b` combined) values. When the variables are not factors and are numeric, another attribute `median` is also returned. This is the median of the uncensored values. When the variables are factor or character, the median of the integer versions of variables for uncensored observations is returned as attribute `mid`. A final attribute `freq` is the vector of frequencies of occurrences of all uncensored values. `freq` aligns with `levels`.

Author(s)

Frank Harrell

`pdensityContour` *pdensityContour*

Description

Bivariate Posterior Contour

Usage

```
pdensityContour(
  x,
  y,
  method = c("ellipse", "kernel"),
  prob = 0.95,
  otherprob = c(0.01, 0.1, 0.25, 0.5, 0.75, 0.9),
  h = c(1.3 * MASS::bandwidth.nrd(x), 1.3 * MASS::bandwidth.nrd(y)),
  n = 70,
  pl = FALSE
)
```

Arguments

`x`	a numeric vector
`y`	a numeric vector the same length of x
`method`	defaults to `'ellipse'`, can be set to `'kernel'`
`prob`	main probability coverage (the only one for `method='ellipse'`)
`otherprob`	vector of other probability coverages for `method='kernel'`
`h`	vector of bandwidths for x and y. See `MASS::kde2d()`.
`n`	number of grid points in each direction, defaulting to normal reference bandwidth (see `bandwidth.nrd`).
`pl`	set to `TRUE` to plot contours

Details

Computes coordinates of a highest density contour containing a given probability volume given a sample from a continuous bivariate distribution, and optionally plots. The default method assumes an elliptical shape, but one can optionally use a kernel density estimator. Code adapted from `embbook::HPDregionplot`. See `http://www.sumsar.net/blog/2014/11/how-to-summarize-a-2d-posterior-using`

Value

a 2-column matrix with x and y coordinates unless `pl=TRUE` in which case a `ggplot2` graphic is returned

Author(s)

Ben Bolker and Frank Harrell

`plot.PostF` *plot.PostF*

Description

Plot Posterior Density of `PostF`

Usage

```
## S3 method for class 'PostF'
plot(
  x,
  ...,
  cint = 0.95,
  label = NULL,
  type = c("linetype", "facet"),
  ltitle = ""
)
```

Arguments

`x`	result of running a function created by `PostF`
`...`	other results created by such functions
`cint`	interval probability
`label`	x-axis label if not the expression originally evaluated. When more than one result is plotted, `label` is a vector of character strings, one for each result.
`type`	when plotting more than one result specifies whether to make one plot distinguishing results by line type, or whether to make separate panels
`ltitle`	used of `type='linetype'` to specify name of legend for the line types

Details

Computes highest posterior density and posterior mean and median as vertical lines, and plots these on the density function. You can transform the posterior draws while plotting.

Value

`ggplot2` object

Author(s)

Frank Harrell

`plot.rmsb` *plot.rmsb*

Description

Plot Posterior Densities and Summaries

Usage

```
## S3 method for class 'rmsb'
plot(
  x,
  which = NULL,
  nrow = NULL,
  ncol = NULL,
  prob = 0.95,
  bivar = FALSE,
  bivarmethod = c("ellipse", "kernel"),
  ...
)
```

Arguments

`x`	an `rms` Bayesian fit object
`which`	names of parameters to plot, defaulting to all non-intercepts. Can instead be a vector of integers.
`nrow`	number of rows of plots
`ncol`	number of columns of plots
`prob`	probability for HPD interval
`bivar`	set to `TRUE` to plot bivariate density contours instead of univariate results (ignored if the number of parameters plotted is not exactly two)
`bivarmethod`	passed as `method` argument to `pdensityContour`
`...`	passed to `pdensityContour`

Details

For an `rms` Bayesian fit object, plots posterior densities for selected parameters along with posterior mode, mean, median, and highest posterior density interval. If the fit was produced by `stackMI` the density represents the distribution after stacking the posterior draws over imputations, and the per-imputation density is also drawn as pale curves. If exactly two parameters are being plotted and `bivar=TRUE`, hightest bivariate posterior density contrours are plotted instead, for a variety of `prob` values including the one specified, using

Value

`ggplot2` object

Author(s)

Frank Harrell

`PostF` *PostF*

Description

Function Generator for Posterior Probabilities of Assertions

Usage

```
PostF(fit, name = c("short", "orig"), pr = FALSE)
```

Arguments

`fit`	a Bayesian fit or `contrast.rms` object
`name`	specifies whether assertions will refer to shortened parameter names (the default) or original names. Shorted names are of the form a1, ..., ak where `k` is the number of intercepts in the model, and b1, ..., bp where `p` is the number of non-intercepts. When using original names that are not legal R variable names, you must enclose them in backticks. For `contrast` objects, `name` is ignored and you must use contrast names. The `cnames` argument to `contrast.rms` is handy for assigning your own names.
`pr`	set to `TRUE` to have a table of short names and original names printed when `name='short'`. For `contrasts` the contrast names are printed if `pr=TRUE`.

Details

From a Bayesian fit object such as that from `blrm()` generates an R function for evaluating the probability that an assertion is true. The probability, within simulation error, is the proportion of times the assertion is true over the posterior draws. If the assertion does not evaluate to a logical or 0/1 quantity, it is taken as a continuous derived parameter and the vector of draws for that parameter is returned and can be passed to the `PostF` plot method. `PostF` can also be used on objects created by `contrast.rms`

Value

an R function

Author(s)

Frank Harrell

Examples

```
## Not run:
  f <- blrm(y ~ age + sex)
  P <- PostF(f)
  P(b2 > 0)     # Model is a1 + b1*age + b2*(sex == 'male')
  P(b1 < 0 & b2 > 0)   # Post prob of a compound assertion
  # To compute probabilities using original parameter names:
  P <- PostF(f, name='orig')
  P(age < 0)    # Post prob of negative age effect
  P(`sex=male` > 0)
  f <- blrm(y ~ sex + pol(age, 2))
  P <- PostF(f)
  # Compute and plot posterior density of the vertex of the
  # quadratic age effect
  plot(P(-b2 / (2 * b3)))

  # The following would be useful in age and sex interacted
  k <- contrast(f, list(age=c(30, 50), sex='male'),
                   list(age=c(30, 50), sex='female'),
                cnames=c('age 30 M-F', 'age 50 M-F'))
```

```
  P <- PostF(k)
  P(`age 30 M-F` > 0 & `age 50 M-F` > 0)
##'
## End(Not run)
```

`predict.blrm` *predict.blrm*

Description

Make predictions from a `blrm()` fit

Usage

```
## S3 method for class 'blrm'
predict(
  object,
  ...,
 type = c("lp", "fitted", "fitted.ind", "mean", "x", "data.frame", "terms", "cterms",
    "ccterms", "adjto", "adjto.data.frame", "model.frame"),
  se.fit = FALSE,
  codes = FALSE,
  posterior.summary = c("mean", "median"),
  cint = 0.95
)
```

Arguments

`object, ..., type, se.fit, codes`	see predict.lrm
`posterior.summary`	set to `'median'` or `'mode'` to use posterior median/mode instead of mean
`cint`	probability for highest posterior density interval

Details

Predict method for `blrm()` objects

Value

a data frame, matrix, or vector with posterior summaries for the requested quantity, plus an attribute `'draws'` that has all the posterior draws for that quantity. For `type='fitted'` and `type='fitted.ind'` this attribute is a 3-dimensional array representing draws x observations generating predictions x levels of Y.

Author(s)

Frank Harrell

See Also

predict.lrm

Examples

```
## Not run:
  f <- blrm(...)
  predict(f, newdata, type='...', posterior.summary='median')

## End(Not run)
```

`print.blrm` *print.blrm*

Description

Print `blrm()` Results

Usage

```
## S3 method for class 'blrm'
print(
  x,
  dec = 4,
  coefs = TRUE,
  intercepts = x$non.slopes < 10,
  prob = 0.95,
  ns = 400,
  title = NULL,
  ...
)
```

Arguments

`x`	object created by `blrm()`
`dec`	number of digits to print to the right of the decimal
`coefs`	specify `FALSE` to suppress printing parameter estimates, and in integer k to print only the first k
`intercepts`	set to `FALSE` to suppress printing intercepts. Default is to print them unless there are more than 9.
`prob`	HPD interval probability for summary indexes
`ns`	number of random samples of the posterior draws for use in computing HPD intervals for accuracy indexes
`title`	title of output, constructed by default
`...`	passed to `prModFit`

Details

Prints main results from `blrm()` along with indexes and predictive accuracy and their highest posterior density intervals computed from `blrmStats`.

Author(s)

Frank Harrell

Examples

```
## Not run: 
  f <- blrm(...)
  options(lang='html')   # default is lang='plain'; also can be latex
  f               # print using defaults
  print(f, posterior.summary='median')   # instead of post. means

## End(Not run)
```

`print.blrmStats` *print.blrmStats*

Description

Print Details for `blrmStats` Predictive Accuracy Measures

Usage

```
## S3 method for class 'blrmStats'
print(x, dec = 3, ...)
```

Arguments

`x`	an object produced by `blrmStats`
`dec`	number of digits to round indexes
`...`	ignored

Details

Prints results of `blrmStats` with brief explanations

Author(s)

Frank Harrell

Examples

```
## Not run:
  f <- blrm(...)
  s <- blrmStats(...)
  s    # print with defaults
  print(s, dec=4)

## End(Not run)
```

`print.predict.blrm` *print.predict.blrm*

Description

Print Predictions for `blrm()`

Usage

```
## S3 method for class 'predict.blrm'
print(x, digits = 3, ...)
```

Arguments

`x`	result from `predict.blrm`
`digits`	number of digits to round numeric results
`...`	ignored

Details

Prints the summary portion of the results of `predict.blrm`

Author(s)

Frank Harrell

`print.rmsb` *print.rmsb*

Description

Basic Print for Bayesian Parameter Summary

Usage

```
## S3 method for class 'rmsb'
print(x, prob = 0.95, dec = 4, intercepts = TRUE, pr = TRUE, ...)
```

Arguments

`x`	an object created by an `rms` Bayesian fitting function
`prob`	HPD interval coverage probability (default is 0.95)
`dec`	amount of rounding (digits to the right of the decimal)
`intercepts`	set to `FALSE` to not print intercepts
`pr`	set to `FALSE` to return an unrounded matrix and not print
`...`	ignored

Details

For a Bayesian regression fit prints the posterior mean, median, SE, highest posterior density interval, and symmetry coefficient from the posterior draws. For a given parameter, the symmetry measure is computed using the `distSym` function.

Value

matrix (rounded if `pr=TRUE`)

Author(s)

Frank Harrell

Examples

```
## Not run:
  f <- blrm(...)
  print.rmsb(f)

## End(Not run)
```

`Quantile.blrm` *Quantile.blrm*

Description

Function Generator for Quantiles of Y for `blrm()`

Usage

```
## S3 method for class 'blrm'
Quantile(object, codes = FALSE, posterior.summary = c("mean", "median"), ...)
```

Arguments

`object`	a `blrm()` fit
`codes`	if `TRUE`, use the integer codes $1, 2, \ldots, k$ for the k-level response in computing the quantile
`posterior.summary`	defaults to posterior mean; may also specify `"median"`. Must be consistent with the summary used when creating `lp`.
`...`	ignored

Details

Creates a function to turn a posterior summarized linear predictor lp (e.g. using posterior mean of intercepts and slopes) computed at the reference intercept into e.g. an estimate of a quantile of Y using the posterior mean of all the intercepts. `lptau` must be provided when call the created function if the model is a partial proportional odds model.

Value

an R function

Author(s)

Frank Harrell

`selectedQr` *selectedQr*

Description

QR Decomposition Preserving Selected Columns

Usage

```
selectedQr(X, not = NULL, corner = FALSE, center = TRUE)
```

Arguments

`X`	a numeric matrix
`not`	an integer vector specifying which columns of `X` are to be kept with their original values
`corner`	set to `FALSE` to not treat the last column specially. You may not specify both `not` and `corner`.
`center`	set to `FALSE` to not center columns of `X` first

Details

Runs a matrix through the QR decomposition and returns the transformed matrix and the forward and inverse transforming matrices R, Rinv. If columns of the input matrix `X` are centered the QR transformed matrix will be orthogonal. This is helpful in understanding the transformation and in scaling prior distributions on the transformed scale. `not` can be specified to keep selected columns as-is. `cornerQr` leaves the last column of `X` alone (possibly after centering). When `not` is specified, the square transforming matrices have appropriate identity submatrices inserted so that recreation of original `X` is automatic.

Value

list with elements X, R, Rinv, xbar where `xbar` is the vector of means (vector of zeros if `center=FALSE`)

Author(s)

Ben Goodrich and Frank Harrell

Examples

```
x <- 1 : 10
X <- cbind(x, x^2)
w <- selectedQr(X)
w
with(w, X %*% R)  # = scale(X, center=TRUE, scale=FALSE)
Xqr <- w$X
plot(X[, 1], Xqr[, 1])
```

```
plot(X[, 1], Xqr[, 2])
cov(X)
cov(Xqr)
X <- cbind(x, x^3, x^4, x^2)
w <- selectedQr(X, not=2:3)
with(w, X %*% R)
```

`stackMI` *Bayesian Model Fitting and Stacking for Multiple Imputation*

Description

Runs an `rmsb` package Bayesian fitting function such as `blrm` separately for each completed dataset given a multiple imputation result such as one produced by `Hmisc::aregImpute`. Stacks the posterior draws and diagnostics across all imputations, and computes parameter summaries on the stacked posterior draws. stackMI

Usage

```
stackMI(
  formula,
  fitter,
  xtrans,
  data,
  n.impute = xtrans$n.impute,
  dtrans,
  derived,
  subset,
  refresh = 0,
  progress = if (refresh > 0) "stan-progress.txt" else "",
  ...
)
```

Arguments

`formula`	a model formula
`fitter`	a Bayesian fitter
`xtrans`	an object created by `transcan`, `aregImpute`, or `mice`
`data`	data frame
`n.impute`	number of imputations to run, default is the number saved in `xtrans`
`dtrans`	see `Hmisc::fit.mult.impute`
`derived`	see `Hmisc::fit.mult.impute`
`subset`	an integer or logical vector specifying the subset of observations to fit
`refresh`	see rstan::sampling. The default is 0, indicating that no progress notes are output. If `refresh` > `0` and `progress` is not `''`, progress output will be appended to file `progress`. The default file name is `'stan-progress.txt'`.

`progress`	see `refresh`. Defaults to `''` if `refresh = 0`. Note: If running interactively but not under RStudio, `rstan` will open a browser window for monitoring progress.
`...`	arguments passed to `fitter`

Value

an `rmsb` fit object with expanded posterior draws and diagnostics

Author(s)

Frank Harrell

`stanDx` *stanDx*

Description

Print Stan Diagnostics

Usage

```
stanDx(object)
```

Arguments

`object`	an object created by an `rms` package Bayesian fitting function such as `blrm()` or `stackMI()`

Details

Retrieves the effect samples sizes and Rhats computed after a fitting function ran `rstan`, and prepares it for printing. If the fit was created by `stackImpute`, the diagnostics for all imputations are printed (separately).

Value

matrix suitable for printing

Author(s)

Frank Harrell

Examples

```
## Not run:
  f <- blrm(...)
  stanDx(f)

## End(Not run)
```

`stanDxplot` *stanDxplot*

Description

Diagnostic Trace Plots

Usage

```
stanDxplot(
  x,
  which = NULL,
  rstan = FALSE,
  previous = TRUE,
  plotno = 1,
  rev = FALSE,
  stripsize = 8,
  ...
)
```

Arguments

`x`	an `rms` Bayesian fit object
`which`	names of parameters to plot, defaulting to all non-intercepts. When `rstan=FALSE` these are the friendly `rms` names, otherwise they are the `rstan` parameter names. If the model fit was run through `stackMI` for multiple imputation, the number of traces is multiplied by the number of imputations. Set to `'ALL'` to plot all parameters.
`rstan`	set to `TRUE` to use `rstan::traceplot()` on a (presumed) stored `rstan` object in `x`, otherwise only real iterations are plotted and parameter values are shown as points instead of lines, with chains separated
`previous`	see details
`plotno`	see details
`rev`	set to `TRUE` to reverse direction for faceting chains
`stripsize`	specifies size of chain facet label text, default is 8
`...`	passed to `rstan::traceplot()`

Details

For an `rms` Bayesian fit object, uses by default the stored posterior draws to check convergence properties of posterior sampling. If instead `rstan=TRUE`, calls the `rstan traceplot` function on the `rstan` object inside the `rmsb` object, to check properties of posterior sampling. If `rstan=TRUE` and the `rstan` object has been removed and `previous=TRUE`, attempts to find an already existing plot created by a previous run of the `knitr` chunk, assuming it was the `plotno` numbered plot of the chunk.

Value

`ggplot2` object if `rstan` object was in `x`

Author(s)

Frank Harrell

`stanGet` *stanGet*

Description

Get Stan Output

Usage

```
stanGet(object)
```

Arguments

`object`	an objected created by an `rms` package Bayesian fitting function

Details

Extracts the object created by `rstan::sampling()` so that standard Stan diagnostics can be run from it

Value

the object created by `rstan::sampling()`

Author(s)

Frank Harrell

Examples

```
## Not run:
  f <- blrm(...)
  s <- stanGet(f)

## End(Not run)
```

`tauFetch` *tauFetch*

Description

Fetch Partial Proportional Odds Parameters

Usage

```
tauFetch(fit, intercept, what = c("tau", "nontau", "both"))
```

Arguments

`fit`	an object created by `blrm()`
`intercept`	integer specifying which intercept to fetch
`what`	specifies the result to return

Details

Fetches matrix of posterior draws for partial proportional odds parameters (taus) for a given intercept. Can also form a matrix containing both regular parameters and taus, or for just non-taus. For the constrained partial proportional odds model the function returns the appropriate `cppo` function value multiplied by tau (tau being a vector in this case and not a matrix).

Value

matrix with number of raws equal to the numnber of original draws

Author(s)

Frank Harrell

`vcov.rmsb` *vcov.rmsb*

Description

Variance-Covariance Matrix

Usage

```
## S3 method for class 'rmsb'
vcov(object, regcoef.only = TRUE, intercepts = "all", ...)
```

Arguments

`object`	an object produced by an `rms` package Bayesian fitting function
`regcoef.only`	set to `FALSE` to also include non-regression coefficients such as shape/scale parameters
`intercepts`	set to `'all'` to include all intercepts (the default), `'none'` to exclude them all, or a vector of integers to get selected intercepts
`...`	ignored

Details

Computes the variance-covariance matrix from the posterior draws by compute the sample covariance matrix of the draws

Value

matrix

Author(s)

Frank Harrell

See Also

vcov.rms

Examples

```
## Not run:
  f <- blrm(...)
  v <- vcov(f)

## End(Not run)
```

Index

aregImpute, *28*
as.data.frame.Ocens, 3

blrm, 4
blrm(), *5–9*, *11, 12*, *14–16*, *20–24*, *26*, *29*, *32*
blrmStats, 8
blrmStats(), *8*

coda::HPDinterval(), *14*
coef.rmsb, 10
coef.rmsb(), *8*
compareBmods, 11

distSym, 11

ExProb.blrm, 12

getParamCoef, 13

Hmisc::rcorr.cens(), *9*
HPDint, 14

loo::loo_model_weights(), *11*

MASS::kde2d(), *17*
Mean.blrm, 14
mice, *28*

Ocens, 15

pdensityContour, 16
plot.PostF, 17
plot.rmsb, 18
PostF, 19
predict.blrm, 21
predict.lrm, *21, 22*
print.blrm, 22
print.blrm(), *8*
print.blrmStats, 23
print.predict.blrm, 24
print.rmsb, 25
print.rmsb(), *8*

Quantile.blrm, 26

rmsb *(*rmsb-package*)*, 2
rmsb-package, 2
rstan::optimizing(), *6, 7*
rstan::sampling, *28*
rstan::sampling(), *6*, *31*
rstan::traceplot(), *30*

selectedQr, 27
stackMI, 28
stackMI(), *29*
stanDx, 29
stanDx(), *8*
stanDxplot, 30
stanGet, 31
stanGet(), *8*

tauFetch, 32

vcov.rms, *33*
vcov.rmsb, 32
vcov.rmsb(), *8*