Package: betaregscale (via r-universe)

October 22, 2026

Type Package

Title Beta Regression for Interval-Censored Scale-Derived Outcomes

Version 3.1.1

Maintainer José Evandeilton Lopes <evandeilton@gmail.com>

Description Maximum-likelihood estimation of beta regression models for responses derived from bounded rating scales. Observations are treated as interval-censored on (0, 1) after a scale-to-unit transformation, and the likelihood is built from the difference of the beta CDF at the interval endpoints. The complete likelihood supports mixed censoring types: uncensored, left-censored, right-censored, and interval-censored observations. Both fixed- and variable-dispersion submodels are supported, with flexible link functions for the mean and precision components. A compiled C++ backend (via 'Rcpp' and 'RcppArmadillo') provides numerically stable, high-performance log-likelihood evaluation. Standard S3 methods (print(), summary(), coef(), fitted(), residuals(), predict(), plot(), confint(), vcov(), logLik(), AIC(), BIC(), simulate()) are available for fitted objects.

License MIT + file LICENSE

URL https://evandeilton.github.io/betaregscale/, https://github.com/evandeilton/betaregscale

BugReports https://github.com/evandeilton/betaregscale/issues

Depends R (>= 4.1.0)

Imports Formula, ggplot2, numDeriv, Rcpp (>= 1.0.12), rlang, stats

Suggests gridExtra, knitr, rmarkdown, testthat (>= 3.0.0)

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation yes

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Author José Evandeilton Lopes [aut, cre] (ORCID: <https://orcid.org/0009-0007-5887-4084>), Wagner Hugo Bonat [aut] (ORCID: <https://orcid.org/0000-0002-0349-7054>)

Repository https://cran.r-universe.dev

Date/Publication 2026-10-22 13:30:17 UTC

RemoteUrl https://github.com/cran/betaregscale

RemoteRef HEAD

RemoteSha e5a4c315daf91bcaf16f6931fecb2ed5d1d7db6f

Contents

AIC.brs . . . 3
AIC.brsmm . . . 4
anova.brs . . . 5
anova.brsmm . . . 7
autoplot.brs . . . 8
autoplot.brs_bootstrap . . . 10
autoplot.brs_marginaleffects . . . 12
autoplot.brsmm . . . 13
BIC.brs . . . 15
BIC.brsmm . . . 16
brs . . . 17
brs_bootstrap . . . 22
brs_cens . . . 25
brs_check . . . 27
brs_coef . . . 30
brs_cv . . . 31
brs_est . . . 32
brs_gof . . . 34
brs_hessian . . . 35
brs_loglik_contrib . . . 36
brs_marginal . . . 37
brs_marginaleffects . . . 40
brs_predict_scoreprob . . . 42
brs_prep . . . 43
brs_repar . . . 46
brs_sim . . . 48
brs_table . . . 50
brsmm . . . 52
brsmm_cv . . . 55
brsmm_re_study . . . 57
brsmm_sim . . . 59
coef.brs . . . 61

coef.brsmm . . . 62
confint.brs . . . 63
confint.brsmm . . . 64
fitted.brs . . . 65
fitted.brsmm . . . 66
formula.brs . . . 67
formula.brsmm . . . 68
logLik.brs . . . 69
logLik.brsmm . . . 70
model.matrix.brs . . . 71
model.matrix.brsmm . . . 72
nobs.brs . . . 73
nobs.brsmm . . . 74
plot.brs . . . 75
plot.brsmm . . . 76
predict.brs . . . 78
predict.brsmm . . . 79
print.brs . . . 80
print.brsmm . . . 81
print.brsmm_re_study . . . 82
print.summary.brs . . . 83
print.summary.brsmm . . . 84
ranef . . . 85
ranef.brsmm . . . 86
residuals.brs . . . 87
residuals.brsmm . . . 89
simulate.brs . . . 91
summary.brs . . . 92
summary.brsmm . . . 94
vcov.brs . . . 95
vcov.brsmm . . . 96

Index **97**

`AIC.brs` *Akaike information criterion*

Description

$-2\ell + k\,p$, with p the number of estimated parameters.

Usage

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

Arguments

object	A fitted "brs" object.
...	Further fitted models (brs, brsmm or others with an AIC() method).
k	Penalty per parameter (default 2).

Value

Scalar AIC value; with several models, a data frame with columns df and AIC, one row per model, as stats::AIC().

See Also

brs, logLik.brs, BIC.brs, brs_gof

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
AIC(fit)
fit0 <- brs(y ~ 1, data = prep)
AIC(fit0, fit)
```

AIC.brsmm *AIC for brsmm models*

Description

$-2\ell + k\,p$, with p the number of parameters (fixed effects, precision, packed random-effect and cutpoint parameters).

Usage

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

Arguments

object	A fitted "brsmm" object.
...	Further fitted models (brsmm, brs or others with an AIC() method).
k	Numeric penalty per parameter (default 2).

Value

Numeric scalar; with several models, a data frame with columns `df` and `AIC`, one row per model, as `stats::AIC()`.

See Also

`brsmm`, `logLik.brsmm`, `BIC.brsmm`, `brs_gof`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
AIC(fit)
AIC(brs(y ~ x1, data = prep), fit)
```

---

anova.brs *Likelihood-ratio comparison of nested beta interval models*

Description

Compares fitted `"brs"` and `"brsmm"` models by log-likelihood, AIC, BIC and likelihood-ratio tests (Lopes, 2023, "Inferencia").

Usage

```
## S3 method for class 'brs'
anova(object, ..., test = c("Chisq", "none"))
```

Arguments

`object`	A fitted `"brs"` model.
`...`	Further fitted `"brs"` and/or `"brsmm"` models.
`test`	`"Chisq"` (default) or `"none"`.

Details

The models are sorted by their number of parameters. For consecutive models the statistic is $LR = 2(\ell_1 - \ell_0)$, with `Chi Df` the difference in the number of parameters, and $p = P(\chi^2_{df} > LR)$. The models must be nested; this is not checked. They must also describe the same response: the same observations, `interval`, `ncuts` and (under `"mid"`) `lim`, or the call stops: fits under different `interval` maps or `lim` are not nested, and different `ncuts` describe different responses. Fits of the same scores under different maps (with `lim = 0.5`) can still be compared by `AIC()` and `BIC()`, since each map gives a probability mass function over the same scores (section 'Interval direction' of `brs`).

When the larger model adds one random-effect term (a `"brs"` model against a random-intercept `"brsmm"`, or one more correlated random term), its variance lies on the boundary of the parameter space under H_0 and LR follows the mixture $\frac{1}{2}\chi^2_{df-1} + \frac{1}{2}\chi^2_{df}$ (Self and Liang, 1987; Stram and Lee, 1994); for one variance component alone this is $\frac{1}{2}\chi^2_0 + \frac{1}{2}\chi^2_1$, i.e. half the naive p-value. The printed heading names the rows that use it. When more than one random term is added at once, the naive χ^2_{df} p-value is kept and flagged as conservative.

A fit with `cutpoints = "fixed"` is nested in the same model with `cutpoints = "boundary"` at an interior point of the cutpoint parameters (section 'Boundary cutpoints' of `brs`), so their comparison is a regular χ^2_2 test when both cutpoints are identified ($K \geq 4$ in practice; with $K = 3$ the fit warns that they are weakly identified); the heading names those rows.

Value

An object of class `"anova"` (a data frame) with columns `Df`, `logLik`, `AIC`, `BIC` and, for `test = "Chisq"`, `Chisq`, `Chi Df` and `Pr(>Chisq)`; the attribute `"heading"` explains the p-values.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Self, S. G., and Liang, K.-Y. (1987). Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. *Journal of the American Statistical Association*, **82**(398), 605–610. doi:10.1080/01621459.1987.10478472

Stram, D. O., and Lee, J. W. (1994). Variance components testing in the longitudinal mixed effects model. *Biometrics*, **50**(4), 1171–1177. doi:10.2307/2533455

See Also

`anova.brsmm`, `summary.brs`, `logLik.brs`

Examples

```
# Synthetic NRS-11 scores: 4 groups x 3 times. Simulated, not real data.
set.seed(2023)
nrs <- expand.grid(id = 1:80, time = c("6h", "12h", "24h"))
nrs$group <- factor(paste0("g", (nrs$id - 1) %% 4 + 1))
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + c(0, -0.1, 0.05, 0.1)[nrs$group]
shp <- brs_repar(mu = plogis(eta), phi = 0.3, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))
```

```
# Time only (m1) nested in time + group (m2): LR test on 3 df
m1 <- brs(y ~ time, data = nrs, ncuts = 10)
m2 <- brs(y ~ time + group, data = nrs, ncuts = 10)
anova(m1, m2)

# Interval maps are not nested: no LR test, but AIC is comparable
m2_right <- brs(y ~ time + group, data = nrs, ncuts = 10, interval = "right")
try(anova(m2, m2_right))
c(mid = AIC(m2), right = AIC(m2_right))
```

---

anova.brsmm *Likelihood-ratio comparison involving mixed models*

Description

anova() for "brsmm" fits: the same table as anova.brs, with the chi-bar-square mixture $\frac{1}{2}\chi^2_{df-1} + \frac{1}{2}\chi^2_{df}$ for rows that add one random-effect term, whose variance is on the boundary under H_0. This is the test to use for a variance component: the Wald statistic of its log standard deviation is not meaningful (see summary.brsmm).

Usage

```
## S3 method for class 'brsmm'
anova(object, ..., test = c("Chisq", "none"))
```

Arguments

object	A fitted "brsmm" model.
...	Further fitted "brsmm" and/or "brs" models.
test	"Chisq" (default) or "none".

Value

An object of class "anova"; see anova.brs.

References

Self, S. G., and Liang, K.-Y. (1987). Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. *Journal of the American Statistical Association*, **82**(398), 605–610. doi:10.1080/01621459.1987.10478472

See Also

anova.brs, brsmm, summary.brsmm

Examples

```
set.seed(11)
g <- 20
d <- data.frame(id = factor(rep(1:g, each = 8)), x = runif(8 * g))
shp <- brs_repar(plogis(-0.4 + d$x + rnorm(g, sd = 0.6)[d$id]), phi = 0.25)
d$y <- round(10 * rbeta(nrow(d), shp$shape1, shp$shape2))
m0 <- brs(y ~ x, data = d, ncuts = 10)
m1 <- brsmm(y ~ x, random = ~ 1 | id, data = d, ncuts = 10)
anova(m0, m1)  # Pr(>Chisq) = half the chi2(1) tail
```

`autoplot.brs` *ggplot2 autoplot for brs models*

Description

Produces ggplot2 diagnostics tailored to interval-censored scale models.

Usage

```
## S3 method for class 'brs'
autoplot(
  object,
  type = c("calibration", "score_dist", "cdf", "residuals_by_delta", "all"),
  bins = 10L,
  scores = NULL,
  newdata = NULL,
  n_grid = 200L,
  max_curves = 6L,
  residual_type = "rqr",
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  ncol = 2L,
  theme = ggplot2::theme_minimal(),
  ...
)
```

Arguments

`object`	A fitted `"brs"` object.
`type`	Plot type: `"calibration"`, `"score_dist"`, `"cdf"`, `"residuals_by_delta"`, or `"all"` (produces all panels in a single grid).
`bins`	Number of bins for `"calibration"`.
`scores`	Optional integer vector of scores for `"score_dist"`. Defaults to all scores from `0` to `ncuts`.

`newdata`	Optional data frame of covariate scenarios for `"cdf"`.
`n_grid`	Number of points on $(0, 1)$ used to draw CDF curves.
`max_curves`	Maximum number of CDF curves shown when `newdata` is not provided.
`residual_type`	Residual type for `"residuals_by_delta"`; passed to `residuals.brs`.
`title`	Optional character: override the plot title via `ggplot2::labs(title = ...)`. Ignored when `type = "all"`.
`xlab`	Optional character: override the x-axis label. Ignored when `type = "all"`.
`ylab`	Optional character: override the y-axis label. Ignored when `type = "all"`.
`ncol`	Number of columns for the grid when `type = "all"`. Defaults to 2.
`theme`	A ggplot2 theme object (e.g., `ggplot2::theme_bw()`) or a theme function. Applied to every panel. Defaults to `ggplot2::theme_minimal()`.
`...`	Additional arguments forwarded to `ggplot2::theme()` and applied on top of `theme`. Use named theme element arguments, e.g. `legend.position = "none"`.

Details

`type = "calibration"` bins predictions and compares mean observed vs mean predicted response in each bin.

`type = "score_dist"` compares observed score frequencies against expected frequencies implied by the fitted beta interval model.

Value

A `ggplot2` object.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

`brs`, `plot.brs`, `autoplot.brs_bootstrap`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  x2 = rep(c(0, 0, 1, 1), 5)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1 + x2, data = prep)
ggplot2::autoplot(fit, type = "calibration")
ggplot2::autoplot(fit,
  type = "calibration",
  title = "My calibration", ylab = "Observed"
)
ggplot2::autoplot(fit, type = "all")
```

---

autoplot.brs_bootstrap

ggplot2 autoplot for bootstrap results

Description

Produces visual summaries for objects returned by brs_bootstrap.

Usage

```
## S3 method for class 'brs_bootstrap'
autoplot(
  object,
  type = c("ci_forest", "dist", "qq", "stability"),
  parameter = NULL,
  title = NULL,
  caption = NULL,
  max_parameters = 12L,
  ci_level = NULL,
  theme = NULL,
  ...
)
```

Arguments

object	An object of class "brs_bootstrap".
type	Plot type: "ci_forest", "dist", "qq", or "stability".

parameter	Optional parameter name used by type = "dist", "qq", and "stability". If NULL, the first parameter is used.
title	Optional plot title override.
caption	Optional subtitles/titles for plot types. Accepts: • a single string (used for the selected type); • a character vector/list with up to four entries in the order ci_forest, dist, qq, stability.
max_parameters	Maximum number of parameters shown in type = "ci_forest".
ci_level	Confidence level used in type = "stability". Defaults to the level stored in object.
theme	Optional ggplot2 theme object (e.g., ggplot2::theme_bw()). If NULL, ggplot2::theme_minimal() is used.
...	Currently ignored.

Details

For type = "dist", "qq", and "stability", bootstrap draws must be present in attr(object, "boot_draws"), obtained by fitting with brs_bootstrap(..., keep_draws = TRUE).

Value

A ggplot2 object.

See Also

brs_bootstrap, brs, autoplot.brs

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
boot <- brs_bootstrap(fit, R = 50)
ggplot2::autoplot(boot, type = "ci_forest")
```

`autoplot.brs_marginaleffects` *ggplot2 autoplot for marginal effects*

Description

Produces visual summaries for objects returned by `brs_marginaleffects`.

Usage

```
## S3 method for class 'brs_marginaleffects'
autoplot(
  object,
  type = c("forest", "magnitude", "dist"),
  variable = NULL,
  top_n = 12L,
  title = NULL,
  caption = NULL,
  theme = NULL,
  ...
)
```

Arguments

`object`	An object of class `"brs_marginaleffects"`.
`type`	Plot type: `"forest"`, `"magnitude"`, or `"dist"`.
`variable`	Optional variable name for `type = "dist"`.
`top_n`	Maximum number of variables shown in `"magnitude"` (ordered by `|AME|`).
`title`	Optional plot title override.
`caption`	Optional subtitle override.
`theme`	Optional ggplot2 theme object. If `NULL`, `ggplot2::theme_minimal()` is used.
`...`	Currently ignored.

Details

`type = "dist"` requires AME simulation draws stored in `attr(object, "ame_draws")`, which are available when marginal effects are computed with `keep_draws = TRUE` and `interval = TRUE`.

Value

A `ggplot2` object.

See Also

`brs_marginaleffects`, `brs`, `autoplot.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
ame <- brs_marginaleffects(fit)
ggplot2::autoplot(ame, type = "forest")
```

`autoplot.brsmm` *ggplot2 autoplot for brsmm models*

Description

Produces ggplot2 diagnostics tailored to mixed beta interval models.

Usage

```
## S3 method for class 'brsmm'
autoplot(
  object,
  type = c("calibration", "score_dist", "ranef_qq", "residuals_by_group",
    "ranef_caterpillar", "ranef_density", "ranef_pairs", "shrinkage", "all"),
  bins = 10L,
  scores = NULL,
  residual_type = c("response", "pearson"),
  max_groups = 25L,
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  ncol = 2L,
  theme = ggplot2::theme_minimal(),
  ...
)
```

Arguments

`object`	A fitted `"brsmm"` object.
`type`	Plot type. One of `"calibration"`, `"score_dist"`, `"ranef_qq"`, `"residuals_by_group"`, `"ranef_caterpillar"`, `"ranef_density"`, `"ranef_pairs"`, `"shrinkage"`, or `"all"` (produces all panels in a single grid).

bins	Number of bins for "calibration".
scores	Optional integer vector of scores for "score_dist". Defaults to all scores from 0 to ncuts.
residual_type	Residual type for "residuals_by_group"; passed to residuals.brsmm.
max_groups	Maximum groups in "residuals_by_group".
title	Optional character: override the plot title via ggplot2::labs(title = ...). Ignored when type = "all".
xlab	Optional character: override the x-axis label. Ignored when type = "all".
ylab	Optional character: override the y-axis label. Ignored when type = "all".
ncol	Number of columns for the grid when type = "all". Defaults to 2.
theme	A ggplot2 theme object (e.g., ggplot2::theme_bw()) or a theme function. Applied to every panel. Defaults to ggplot2::theme_minimal().
...	Additional arguments forwarded to ggplot2::theme() and applied on top of theme. Use named theme element arguments, e.g. legend.position = "none".

Value

A ggplot2 object, or (when type = "all") invisibly returns the list of all panels after rendering the grid.

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit_mm <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
ggplot2::autoplot(fit_mm, type = "calibration", bins = 4)
ggplot2::autoplot(fit_mm,
  type = "ranef_caterpillar",
  title = "My title", ylab = "Mode"
)
ggplot2::autoplot(fit_mm, type = "all")
```

BIC.brs *Bayesian information criterion*

Description

Bayesian information criterion

Usage

```
## S3 method for class 'brs'
BIC(object, ..., n = c("obs", "groups"))
```

Arguments

object	A fitted "brs" object.
...	Further fitted models (brs, brsmm or others with a BIC() method).
n	"obs" (default): the number of observations. "groups" applies to brsmm fits only (see BIC.brsmm) and is an error here.

Value

Scalar BIC value; with several models, a data frame with columns df and BIC, one row per model, as stats::BIC().

See Also

brs, logLik.brs, AIC.brs, brs_gof

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
BIC(fit)
```

BIC.brsmm *BIC for brsmm models*

Description

$-2\ell + \log(n)\, k$, with k the number of parameters (fixed effects, precision, packed random-effect and cutpoint parameters) and n the number of observations (`n = "obs"`, the default, as `lme4` does) or of groups (`n = "groups"`). There is no single sample size for a mixed model; counting groups penalizes less when groups have several observations each. Compare BIC values only between fits with the same convention.

Usage

```
## S3 method for class 'brsmm'
BIC(object, ..., n = c("obs", "groups"))
```

Arguments

`object`	A fitted `"brsmm"` object.
`...`	Further fitted models; with `n = "groups"` they must be `"brsmm"` fits, otherwise their own `BIC()` is used.
`n`	`"obs"` (default; `nobs(object)`) or `"groups"` (`object$ngroups`).

Value

Numeric scalar; with several models, a data frame with columns `df` and `BIC`, one row per model, as `stats::BIC()`.

See Also

`brsmm`, `logLik.brsmm`, `AIC.brsmm`, `brs_gof`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
BIC(fit)
BIC(fit, n = "groups")
```

brs *Fit a beta interval regression model*

Description

Fits by maximum likelihood a beta regression for scores on a bounded scale, treated as coarsened (interval-censored) observations of a latent beta variable (Lopes, 2023). A one-part formula `y ~ x1 + x2` keeps the second parameter constant (`brs_fit_fixed`); a two-part formula `y ~ x1 + x2 | z1 + z2` also models it (`brs_fit_var`).

Usage

```
brs(
  formula,
  data,
  link = NULL,
  link_phi = NULL,
  ncuts = NULL,
  lim = NULL,
  repar = 2L,
  method = c("BFGS", "L-BFGS-B"),
  hessian_method = c("cpp", "numDeriv", "optim"),
  interval = NULL,
  start = NULL,
  control = list(),
  cutpoints = c("fixed", "boundary")
)
```

Arguments

`formula`	A `Formula`-style formula with two parts: `y ~ x1 + x2 | z1 + z2`.
`data`	Data frame.
`link`	Link for the first parameter (the mean under `repar = 1, 2`; the shape p under `repar = 0`). `NULL` (default) selects the link implied by `repar`: `"logit"` for `repar = 1, 2`, `"log"` for `repar = 0`. See the 'Reparameterizations and links' section of `brs` for the admissible values.
`link_phi`	Link for the second parameter. `NULL` (default) selects `"logit"` for `repar = 2` (dispersion on $(0, 1)$) and `"log"` for `repar = 0, 1` (positive shape/precision).
`ncuts`	Integer K: the maximum score, so that the scale is $0, 1, \ldots, K$ ($K + 1$ categories). `NULL` (default) uses the value stored by `brs_prep` or `brs_sim`, or 100 for raw data.
`lim`	Half-width of the score cell in $(0, 0.5]$ (`interval = "mid"` only). `NULL` (default) uses `attr(data, "lim")` from `brs_prep`, or 0.5; same rule as `ncuts`. Values below 0.5 warn (partial coarsening).
`repar`	Reparameterization scheme (default 2); see `brs_repar`.

`method`	Optimization method (default `"BFGS"`).
`hessian_method`	How the Hessian for `vcov()` is computed: `"cpp"` (default; compiled chain rule on the linear predictors), `"numDeriv"` (`numDeriv::hessian()` of the log-likelihood) or `"optim"` (the finite-difference Hessian returned by `optim`).
`interval`	Direction of the uncertainty interval, `"mid"`, `"right"` or `"left"` (see `brs_check`). `NULL` (default) uses `attr(data, "interval")` from `brs_prep`, or `"mid"`; same rule as `ncuts`.
`start`	Optional numeric vector of starting values (mean coefficients, then dispersion coefficients). `NULL` (default) uses `compute_start()`. Refits (bootstrap, jackknife) pass the parent estimate here.
`control`	Control list for `optim`; its entries are merged into the default `list(maxit = 5000L)`.
`cutpoints`	`"fixed"` (default) keeps the cutpoints of the interval map; `"boundary"` also estimates the two boundary cutpoints (see the section 'Boundary cutpoints' of `brs`).

Details

The scores run over $0, 1, \ldots, K$ with $K =$ `ncuts`: the scale has $K + 1$ categories and K is its maximum. A scale that starts at 1, such as a Likert item 1–5, must be shifted to 0–4 and fitted with `ncuts = 4`; otherwise the lowest category is read as an interior score and the censoring of the lower border is lost.

Raw scores are converted to cells and censoring types by `brs_check` with `ncuts`, `lim` and `interval`; data from `brs_prep` or `brs_sim` are used as they are, with their stored `ncuts`, `lim` and `interval`. Values in $(0, 1)$ are exact observations. The model is

$$Y_i \sim \mathrm{Beta}(a_i, b_i), \qquad g_1(\mu_i) = x_i^\top \beta, \qquad g_2(\phi_i) = z_i^\top \gamma,$$

with (a_i, b_i) obtained from (μ_i, ϕ_i) by `brs_repar`.

Value

An object of class `"brs"`: a list with, among others, `par` (estimates on the link scales), `coefficients` (`mean` and `precision` parts), `value` (maximized log-likelihood), `hessian`, `convergence`, `diagnostics` (see 'Fit diagnostics'), `hatmu` (first parameter per observation: the mean, or the shape p under `repar = 0`), `hatphi`, `Y` (columns `left`, `right`, `yt`, `y`, `delta`; with `cutpoints = "boundary"` the cells and cell centers of the estimated cutpoints), `ncuts`, `lim`, `interval`, `repar`, `link`, `link_phi`, `cutpoints` (the type) and `cut` (`NULL` under the fixed map; otherwise a list with `par`, the logit-scale estimates, `value`, the cutpoints on $(0, 1)$, `jacobian`, `fixed`, the map's cutpoints, and `start`, their logit-scale values).

Likelihood

With cell $[l_i, u_i]$ and censoring type δ_i,

$$L(\beta, \gamma) = \prod_{i=1}^{n} f(y_i)^{I(\delta_i=0)} \, F(u_i)^{I(\delta_i=1)} \, \{1 - F(l_i)\}^{I(\delta_i=2)} \, \{F(u_i) - F(l_i)\}^{I(\delta_i=3)},$$

where f and F are the beta density and distribution function with shapes (a_i, b_i). The four factors are exact values ($\delta = 0$), the lowest score ($\delta = 1$, left-censored), the highest score ($\delta = 2$, right-censored) and the other scores ($\delta = 3$, interval-censored). This is eq. `eqn_verossimilhanca_geral` of Lopes (2023); the table of censoring types in the same section swaps the labels $\delta = 1$ and $\delta = 2$, and the package follows the equation.

Numerical details: endpoints are clamped to $[10^{-5}, 1 - 10^{-5}]$ and there is no probability floor; an interval probability uses lower-tail probabilities when the interval midpoint is at or below the mean $a/(a + b)$ and upper tails otherwise, which avoids cancellation; below 10^{-240} an endpoint Laplace approximation of the tail replaces it; a non-finite contribution becomes -10^6, as does a `NaN` parameter. The mean and the dispersion (`repar = 2`) are clamped to $[10^{-5}, 1 - 10^{-5}]$, the precision and the shape p to $[10^{-5}, 10^8]$, and the beta shapes to $[10^{-12}, 10^8]$.

Estimation

`optim` (`method = "BFGS"`, the default, or `"L-BFGS-B"`; `maxit = 5000` unless `control` says otherwise) maximizes the log-likelihood, which is evaluated in C++. The gradient uses the chain rule on the two linear predictors: the derivatives of each contribution with respect to η_{1i} and η_{2i} are Richardson central differences with step $10^{-4} \max(1, |\eta|)$, and the gradient is $X^\top d_1$ and $Z^\top d_2$. The Hessian behind `vcov()` is built the same way from the per-observation second derivatives (step $3 \times 10^{-4} \max(1, |\eta|)$; `hessian_method = "cpp"`, the default):

$$H = \begin{pmatrix} X^\top W_{11} X & X^\top W_{12} Z \\ Z^\top W_{12} X & Z^\top W_{22} Z \end{pmatrix},$$

with W_{jk} diagonal matrices of the second derivatives of each contribution in (η_{1i}, η_{2i}). `"numDeriv"` differentiates the log-likelihood with `numDeriv::hessian()` and `"optim"` takes the finite-difference Hessian of `optim`. Lopes (2023, "Estimacao") also uses BFGS with numerical derivatives; here they are taken on the linear predictors, not on the coefficients.

Starting values: `start` when given (bootstrap and jackknife refits pass the parent estimate); otherwise a quasi-binomial GLM of the cell midpoints for the mean, and for the second parameter the moment estimate on the midpoints as the intercept with zero slopes; under `repar = 0` both shapes by the method of moments. The covariance matrix is $(-H)^{-1}$ at the estimate (`vcov.brs`); no generalized inverse is used.

Fit diagnostics

Before `optim`, the mean and precision model matrices are checked (pivoted QR, tolerance 10^{-7}; condition number of the matrix with unit-length columns). After it, `fit$diagnostics` holds `grad_norm` (largest absolute gradient), `grad_gain` (the log-likelihood a Newton step would still gain, $\frac{1}{2} g^\top (-H)^{-1} g$), `grad_step` (that step in standard errors), `min_eig`, `max_eig` and `min_eig_scaled` (eigenvalues of $-H$, the last in correlation form), `hessian_nd`, `n_clamped` and `clamped` (observations on the clamps of the likelihood). Each problem gives one line:

"model matrix is rank deficient" (error) Some columns are linear combinations of others (the message names them). Remove or recode them.

"model matrix is nearly collinear" Condition number above 10^4: estimates and standard errors are unstable. Drop or combine the named columns, or center them.

"Optimizer did not converge" `optim` stopped at its iteration limit or failed. Try `method = "L-BFGS-B"`, rescale the covariates or simplify the model.

"Gradient not ~0 at the estimate" A Newton step would still gain more than 0.01 in log-likelihood although `optim` reported convergence (or, when the Hessian is not negative definite, the largest absolute gradient exceeds 0.1). Refit with the other `method`; the message asks to rescale the covariates when their scales differ by more than 10^3 (e.g. a raw income next to a dummy).

"Hessian near-singular or not negative definite (SEs unreliable)" The likelihood is nearly flat or curved the wrong way in some direction: a parameter that is not identified, or covariates on very different scales (the message then asks to rescale them). `vcov()` returns `NA` for variances it cannot estimate; use likelihood-ratio tests (`anova.brs`) or `brs_bootstrap` instead of Wald statements, rescale, or simplify the model.

"observations ... on the clamp boundary" Fitted means, dispersions or shapes sit on the numerical limits above, typically because a group of observations lies entirely at one end of the scale (separation) or every observation is censored on the same side. The coefficients drift towards infinity and are not interpretable; merge sparse groups or remove the separating covariate.

`brsmm` adds a check of the variance components.

Reparameterizations and links

`repar`	parameters (Lopes, 2023)	`link`
0	shapes $p, q > 0$; eq. `eqn_beta_p1` (regression on the shapes is a package extension)	`log`, `sqrt`
1	mean $\mu \in (0,1)$, precision $\phi > 0$; "parametrizacao 1" (Ferrari and Cribari-Neto, 2004)	`logit`, `probit`, `cauchit`
2	mean $\mu \in (0,1)$, dispersion $\phi \in (0,1)$; "parametrizacao 2" (Bayer, 2011), written σ there	`logit`, `probit`, `cauchit`

The first link of each cell is the default (`link = NULL`, `link_phi = NULL`); any other combination is an error. `"identity"`, `"inverse"` and `"1/mu^2"` are not accepted for positive parameters. With `"sqrt"` the inverse link is flat for $\eta \leq 0$, and a warning is issued when a fitted linear predictor lies there. The mean and variance of Y under each scheme are given in `brs_repar`. Under `repar = 0` the object stores the shape p in `hatmu`, while `fitted()`, `predict(type = "response")`, residuals and marginal effects use $E[Y] = p/(p+q)$.

Interval direction

`interval = "mid"` (default) uses the cells $[s - \lim, s + \lim]/K$; `"right"` and `"left"` use $K+1$ equal cells $[s, s+1]/(K+1)$ (the dissertation's r and l, with a package normalization). `"right"` and `"left"` give the same fit and differ only in `predict(type = "score")`; `"mid"` and `"right"`/`"left"` are different coarsening models. For score data (with `lim = 0.5`) each map gives a probability mass function over the same scores $0, \ldots, K$, so `logLik()`, AIC and BIC of fits under different maps are comparable; the maps are not nested, so there is no likelihood-ratio test between them (`anova.brs` stops). Details: `brs_check`.

Boundary cutpoints

Under the fixed map score s has the cell (c_{s-1}, c_s) with cutpoints fixed by `interval`, so the model is an ordinal threshold model whose thresholds are the scale labels. `cutpoints = "boundary"` estimates the two border cutpoints: c_{lo} between scores 0 and 1 and c_{hi} between $K-1$ and K; all other cutpoints keep the values of the map. Score 0 then has the cell $(0, c_{lo})$, score 1 (c_{lo}, u_1), score

$K-1$ (l_{K-1}, c_{hi}) and score K $(c_{hi}, 1)$, where u_1 and l_{K-1} are the ends given by the map. With $\epsilon = 10^{-5}$, the border clamp of the likelihood, the parameters are unconstrained and keep the order:

$$c_{lo} = \epsilon + (u_1 - \epsilon)\,\mathrm{logit}^{-1}(g_{lo}), \qquad c_{hi} = l_{K-1} + (1 - \epsilon - l_{K-1})\,\mathrm{logit}^{-1}(g_{hi}),$$

so the free cutpoints never enter the clamp and the likelihood is smooth in (g_{lo}, g_{hi}). The map is the interior point $g_{lo} = \mathrm{logit}\{(u_0 - \epsilon)/(u_1 - \epsilon)\}$, $g_{hi} = \mathrm{logit}\{(l_K - l_{K-1})/(1 - \epsilon - l_{K-1})\}$ (about $\mathrm{logit}(1/3)$ and $\mathrm{logit}(2/3)$ under `"mid"`, about 0 under `"right"`/`"left"`), so the fixed model is nested in the boundary model. When both cutpoints are identified ($K \geq 4$ in practice, see below) `anova.brs` gives a regular likelihood-ratio test on 2 degrees of freedom. The fit starts at the map.

`g_lo` and `g_hi` are the last two coefficients, `"(cut_lo)"` and `"(cut_hi)"` (`coef(fit, model = "cutpoints")`); `summary()` reports c_{lo} and c_{hi} on $(0, 1)$ and on the latent score scale of `interval` (the scale of `predict(type = "score")`), with delta-method standard errors. Their gradient is analytic: $\pm f(c)/\{F(u_i) - F(l_i)\}$ for each row whose cell ends at c (plus when c is the upper end), times the Jacobian of the map above. The compiled Hessian covers the regression coefficients; the rows and columns of the cutpoint parameters come from `numDeriv::jacobian()` of the gradient (`hessian_method = "cpp"`). The fitted cells (`fit$Y`), score probabilities, expected scores, randomized quantile residuals, `brs_bootstrap` and `brs_cv` use the estimated cutpoints; `brs_sim` simulates from given ones.

The extension needs score data: every row the cell of a score $0, \ldots, K$ (no exact values, no analyst-supplied censoring or bounds from `brs_prep`), `lim = 0.5`, $K \geq 3$, and the scores next to each free cutpoint observed (0, 1, $K-1$, K); otherwise `brs()` stops, since a cutpoint with an unobserved neighbor is not identified. With $K = 2$ the two cutpoints are not identified and `brs()` stops; with $K = 3$ they rest mainly on the shape of the beta distribution and a warning says they are weakly identified, so the χ^2_2 reference is not trustworthy there. A border cell holding less than 10^{-4} of the range its cutpoint can take ($c_{lo} - \epsilon$ or $u_1 - c_{lo}$ relative to $u_1 - \epsilon$, and the same for c_{hi}) warns: that cell is collapsing and its cutpoint is not identified. The cell centers in `fit$Y[, "yt"]` (midpoint residuals, pseudo R^2) are those of the estimated cells: $c_{lo}/2$, $(c_{lo} + u_1)/2$, $(l_{K-1} + c_{hi})/2$ and $(c_{hi} + 1)/2$. The border centers differ by construction from those of the fixed map (10^{-5} and $1 - 10^{-5}$ under `"mid"`), so pseudo R^2 and midpoint residuals are not directly comparable between fixed-map and boundary fits.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

Bayer, F. M. (2011). *Modelagem e inferencia em regressao beta*. PhD thesis, Universidade Federal de Pernambuco.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

See Also

brs_check, brs_prep, brs_repar, summary.brs, predict.brs, residuals.brs, brs_predict_scoreprob, anova.brs, brsmm

Examples

```
# Synthetic NRS-11 pain scores (0-10, so ncuts = 10): 4 groups x 3
# post-operative times, patterned on the knee-surgery design of Lopes (2023).
# Simulated, not real data.
set.seed(2023)
nrs <- expand.grid(id = 1:80, time = c("6h", "12h", "24h"))
nrs$group <- factor(paste0("g", (nrs$id - 1) %% 4 + 1))
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + c(0, -0.1, 0.05, 0.1)[nrs$group]
shp <- brs_repar(mu = plogis(eta), phi = 0.3, repar = 2)  # dispersion 0.3
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))

# Default cells [s - 0.5, s + 0.5] / 10; scores 0 and 10 are censored
fit <- brs(y ~ time + group, data = nrs, ncuts = 10)
summary(fit)
confint(fit)

# Post-fit checks: no log-likelihood left to gain, negative definite Hessian, no clamps
fit$diagnostics[c("grad_gain", "hessian_nd", "n_clamped")]

# Same scores read with right-direction cells [s, s + 1] / 11
fit_r <- brs(y ~ time + group, data = nrs, ncuts = 10, interval = "right")
cbind(mid = coef(fit), right = coef(fit_r))

# New patients: mean on (0, 1), latent score, expected score, P(S = s)
nd <- data.frame(time = c("6h", "12h", "24h"), group = "g1")
predict(fit, newdata = nd)
predict(fit, newdata = nd, type = "score")
predict(fit, newdata = nd, type = "expected_score")
round(brs_predict_scoreprob(fit, newdata = nd), 3)

# Estimated boundary cutpoints: a 2-df likelihood-ratio test against the map
fit_b <- brs(y ~ time + group, data = nrs, ncuts = 10, cutpoints = "boundary")
summary(fit_b)$cutpoints
anova(fit, fit_b)

# Randomized quantile residuals respect the censoring (N(0, 1) under the model)
set.seed(1)
summary(residuals(fit, type = "rqr"))
plot(fit, which = 1:2)
```

---

brs_bootstrap *Parametric bootstrap confidence intervals for brs models*

Description

Computes bootstrap-based confidence intervals for the parameters of a fitted "brs" model by repeatedly simulating data from the fitted model and re-estimating parameters. Only "brs" (fixed or variable-dispersion) objects are supported; "brsmm" is not supported.

Usage

```
brs_bootstrap(
  object,
  R = 199L,
  level = 0.95,
  ci_type = c("percentile", "basic", "normal", "bca"),
  max_tries = NULL,
  keep_draws = FALSE
)

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

Arguments

object	A fitted "brs" object (fixed or variable dispersion).
R	Integer: number of bootstrap replicates (default 199).
level	Numeric: confidence level (default 0.95).
ci_type	Character: type of confidence interval. One of "percentile" (default), "basic", "normal", or "bca". See the section on the cost of "bca" below.
max_tries	Optional integer: maximum number of bootstrap attempts to obtain converged replicates. If NULL, uses max(3 * R, 50).
keep_draws	Logical: if TRUE, stores successful bootstrap parameter draws in attribute "boot_draws".
x	Object returned by brs_bootstrap.
...	Ignored.

Details

Each replicate draws a new response $y_i^* \sim \mathrm{Beta}(a_i, b_i)$ at the fitted shapes of the rows used by the fit and substitutes it into a copy of object$data; covariates, factor levels, transformations and the formula stay those of the original fit, which is then re-fitted with brs under the same ncuts, lim, interval, links, repar and optimizer. The observation mechanism of each row is reproduced: exact observations ($\delta = 0$) stay continuous; scores are re-coarsened on the fit's grid (the mapping of brs_sim). Rows whose bounds came from the analyst (brs_prep Modes 2–4, bounds that are not the cell of the row's score) keep their thresholds as fixed and non-informative (independent of Y), and δ is re-drawn by the cell of the partition they induce where y^* falls: for an upper bound c, $y^* \le c$ gives $\delta = 1$ on $[\epsilon, c]$, otherwise $\delta = 2$ on $[c, 1-\epsilon]$ (an interval $[l, u]$ gives three cells). When the original design had more thresholds than a row records (e.g. Mode 4 intervals cut from a finer instrument) the bootstrap is conservative, and for Mode 2 rows with a forced δ (a threshold built from the score itself, which is informative) it is only an approximation. The response must be

a variable (not an expression such as `I(y / 10)`). A fit with `cutpoints = "boundary"` re-coarsens with its estimated boundary cutpoints and re-estimates them in every replicate; a replicate without one of the scores 0, 1, $K - 1$ or K cannot be refitted and counts as failed. The reasons of failed replicates are kept in `attr(, "fail_reasons")` and printed.

Each refit starts from the estimate of `object` and uses the compiled Hessian (`hessian_method = "cpp"`), so replicates are cheap.

Replicates that fail (refit error, non-convergence, non-finite estimates) are discarded and counted: attributes `"n_failed"` and `"fail_rate"`, also printed. Intervals are computed from the bootstrap distribution of each parameter, with the method controlled by `ci_type`: `"percentile"` (default) uses the raw empirical quantiles; `"basic"` uses reflected empirical quantiles; `"normal"` uses a normal approximation from the bootstrap standard error (no quantiles); `"bca"` uses bias-corrected-and-accelerated adjusted quantiles. With parametric resampling and a nonparametric (leave-one-out) jackknife acceleration, `"bca"` is an approximation; a warning says so once per session, and failed jackknife refits are reported in `"n_jack_failed"`.

Value

A data frame with columns `parameter`, `estimate` (original point estimate), `se_boot` (bootstrap standard error), `ci_lower`, `ci_upper`, `mcse_lower`, `mcse_upper`, `wald_lower`, `wald_upper`, and `level`. The attribute `"n_success"` gives the number of replicates that converged. Additional attributes include `"R"`, `"n_attempted"`, `"n_failed"`, `"fail_rate"`, `"n_jack_failed"` (BCa only), `"fail_reasons"` (counts of the failure messages, or `NULL`), `"ci_type"`, and optionally `"boot_draws"`.

Methods (by generic)

- `print(brs_bootstrap)`: Print method for bootstrap results

Cost of `ci_type = "bca"`

The bias-corrected and accelerated interval needs an acceleration constant, which is obtained here by a leave-one-out jackknife. That requires `n` additional model fits, one per observation, on top of the `R` bootstrap replicates: the total is `R + n` fits rather than `R`. The cost is therefore driven by the sample size, not by `R`, and grows quickly – for `n = 1000` the jackknife alone dominates the run time by an order of magnitude. The other three interval types need only the `R` replicates. Prefer `"percentile"` or `"basic"` for exploratory work on large samples, and reserve `"bca"` for a final result.

See Also

`confint.brs` for Wald intervals; `brs_sim` for simulation; `brs` for fitting.

Examples

```
# Synthetic NRS-11 scores at 6h, 12h, 24h (time is a factor)
set.seed(3)
nrs <- data.frame(time = factor(rep(c("6h", "12h", "24h"), each = 40),
                                levels = c("6h", "12h", "24h")))
shp <- brs_repar(mu = plogis(-1.3 + c(0, 0.75, 0.3)[nrs$time]), phi = 0.3)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))
```

```
fit <- brs(y ~ time, data = nrs, ncuts = 10)

# Percentile intervals from 30 parametric replicates (use R >= 199 in practice)
set.seed(4)
bt <- brs_bootstrap(fit, R = 30)
bt
# Bootstrap next to Wald limits, and the bootstrap/Wald SE ratio
cols <- c("parameter", "ci_lower", "ci_upper", "wald_lower", "wald_upper")
as.data.frame(bt)[, cols]
round(bt$se_boot / sqrt(diag(vcov(fit))), 2)
```

`brs_cens` *Graphical and tabular censoring summary*

Description

Produces a visual summary of the censoring structure in a fitted `"brs"` model or a response matrix produced by `brs_check`. The summary includes:

1. Bar chart of censoring-type counts, with percentage labels (censoring burden).
2. Boxplot (with jittered points) of the midpoint response y_t by censoring type.
3. Boxplot of interval width $(u_i - l_i)$ by censoring type.
4. Ordered interval map: $[l_i, u_i]$ segments for a sample of observations, colored by censoring type.

Usage

```
brs_cens(
  object,
  n_sample = 100L,
  which = 1:4,
  caption = NULL,
  gg = FALSE,
  title = "Censoring diagnostic overview",
  sub.caption = NULL,
  theme = NULL,
  palette = NULL,
  inform = FALSE,
  ...
)
```

Arguments

`object`	A fitted `"brs"` object, a matrix returned by `brs_check`, or a data frame returned by `brs_prep` (must contain columns `left`, `right`, `yt`, and `delta`).
`n_sample`	Integer: maximum number of observations to show in the interval plot (default 100). If the data has more observations, a random sample is drawn.

`which`	Integer vector selecting which panels to draw (default `1:4`).
`caption`	Optional panel captions. Accepts a character vector (or list coercible to character) with up to 4 labels, in the order: burden, midpoint-by-type, width-by-type, ordered interval map.
`gg`	Logical: use ggplot2? (default `FALSE`).
`title`	Optional global title for the plotting page.
`sub.caption`	Optional subtitle/caption for the plotting page.
`theme`	Optional ggplot2 theme object (e.g., `ggplot2::theme_bw()`). If `NULL`, a minimal theme is used when `gg = TRUE`.
`palette`	Optional named character vector with colors for censoring types `Exact`, `Left`, `Right`, and `Interval`.
`inform`	Logical; if `TRUE`, prints brief interpretation messages about boundary and interval censoring intensity.
`...`	Further arguments (currently ignored).

Value

Invisibly returns a data frame with censoring counts and proportions, percentages, and interpretation flags.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

Examples

```
y <- c(0, 3, 5, 7, 10)
Y <- brs_check(y, ncuts = 10)
brs_cens(Y)

prep <- brs_prep(data.frame(y = y), ncuts = 10)
brs_cens(prep)
```

brs_check *Map scale scores to intervals on (0, 1) and censoring types*

Description

Maps a score on $\{0, 1, \ldots, K\}$ ($K =$ `ncuts`, the maximum score; $K+1$ categories) to a cell $[l_s, u_s]$ of $(0, 1)$ and to a censoring type δ of the complete likelihood (Lopes, 2023, eq. `eqn_verossimilhanca_geral`; see `brs`): $\delta = 0$ density $f(y)$, $\delta = 1$ $F(u)$, $\delta = 2$ $1 - F(l)$, $\delta = 3$ $F(u) - F(l)$. A value in $(0, 1)$ is exact ($\delta = 0$), observation by observation. A scale that starts at 1 (e.g. a Likert item 1–5) must be shifted to start at 0 (0–4, `ncuts = 4`): its lowest category is then the left-censored border.

Usage

```
brs_check(
  y,
  ncuts = 100L,
  lim = 0.5,
  delta = NULL,
  interval = c("mid", "right", "left")
)
```

Arguments

`y`	Numeric vector: scores on $\{0, 1, \ldots, K\}$ or values in $(0, 1)$.
`ncuts`	Integer K: the maximum score (default 100); the scale is $0, 1, \ldots, K$, with $K+1$ categories. Must be $\geq \max(y)$.
`lim`	Numeric in $(0, 0.5]$: half-width of the cell under `interval = "mid"` (default 0.5, adjacent cells touch). Values below 0.5 give a partial coarsening (warning); ignored, with a warning, for `"right"`/`"left"`.
`delta`	Integer vector or `NULL`. If `NULL` (default), censoring types are derived from the scores. If provided, it must have the length of `y` with elements in `{0, 1, 2, 3}`; it overrides the type per observation (see Details).
`interval`	Direction of the uncertainty interval: `"mid"` (default), `"right"` or `"left"`; see the section 'Interval direction'.

Details

With `delta = NULL`, each value in $(0, 1)$ is exact and each other value is a score with its cell and the type above; this is the rule of `brs_prep` too. Input that mixes values in $(0, 1)$ with values ≥ 1 is ambiguous (proportions next to scores, or rescaled scores) and gives one warning; rescale it if the values in $(0, 1)$ are meant as scores (half-point scores 0, 0.5, 1, ...: use `y * 2` and `ncuts * 2`). A user-supplied `delta` (the mechanism `brs_sim` uses in Monte Carlo studies) forces the type per observation and keeps the cell endpoints of the score:

δ	l_i	u_i
0	cell center (or y when in $(0, 1)$)	same

1	ϵ	u_s
2	l_s	$1-\epsilon$
3	l_s	u_s

A forced $\delta = 1$ or 2 uses the cell of the observed score as the censoring threshold, so the threshold depends on Y: applied to every observation this is informative censoring and the likelihood has no finite maximum (`brs_sim` warns). Under `"mid"` with `lim = 0.5`: $u_0 = 0.5/K$, $l_K = (K-0.5)/K$ and $[l_s, u_s] = [(s-0.5)/K, (s+0.5)/K]$. Scores outside $[0, K]$ are an error (as in `brs_prep`), and so is a $\delta = 3$ observation with $l_i \geq u_i$ (zero-probability interval). When the clamp squeezes a cell to zero width (a cell narrower than 10^{-5} at a border, i.e. `ncuts` too large or `lim` too small), the error says so.

`yt` is the cell center: s/K under `"mid"` (so the border scores sit at ϵ and $1-\epsilon$) and $(s+0.5)/(K+1)$ otherwise; exact values are kept. It is the density argument for $\delta = 0$ and the point used by the midpoint residuals; censored contributions do not use it.

Data carrying the `"is_prepared"` attribute (from `brs_prep` or `brs_sim`) are used by `brs` as they are; `brs_check()` runs only for raw scores.

Value

A numeric matrix with one row per observation and columns `left` (l_i), `right` (u_i), `yt` (cell center), `y` (the input) and `delta`.

Interval direction

`interval` is the direction of the uncertainty interval around the score (Lopes, 2023, "Mapeamento de intervalos para beta": $m = [s-0.5, s+0.5]$, $r = [s, s+1]$, $l = [s-1, s]$).

`interval`	cell of score s	latent score of $y^* \in (0,1)$
`"mid"`	$[s-\lim, s+\lim]/K$	Ky^*
`"right"`	$[s, s+1]/(K+1)$	$(K+1)y^*$
`"left"`	$[s, s+1]/(K+1)$	$(K+1)y^* - 1$

The latent score is the back-transformation used by `predict(type = "score")`: it lies in $[s-0.5, s+0.5]$, $[s, s+1]$ or $[s-1, s]$ when y^* is in the cell of s. The $K+1$ cells of `"right"` and `"left"` are equal and partition $[0,1]$. This is a package choice that differs from the dissertation, which divides r and l by K; there the two directions differ by $1/K$ and chapter 4 reports opposite intercept biases for them, while here they give the same likelihood and the same coefficients. The three directions are different coarsening models of the same scores. With `lim = 0.5` each one gives a probability mass function over $0, \ldots, K$, so their log-likelihoods, AIC and BIC are comparable; the models are not nested, so `anova.brs` refuses a likelihood-ratio test between them. `lim` applies to `"mid"` only.

Scale change and borders

Lopes (2023, "Mudanca de escala intervalar") rescales scores by the range and handles the edge effect either with the transformation $y^* = \{y(n-1)/R + 1/2\}/n$ (Smithson and Verkuilen, 2006), which depends on the sample size n and the observed range R, or by moving 0 and 1 inwards by $\zeta = 10^{-4}$. The package does neither: each score is coarsened into its cell, using the scale

maximum K rather than the observed range; the border scores 0 and K become left- and right-censored observations with contributions $F(u_0)$ and $1 - F(l_K)$; and the endpoints are clamped to $[\epsilon, 1-\epsilon]$, $\epsilon = 10^{-5}$, only as a numerical guard. The border cells therefore keep their full probability, no data are moved towards 1/2, and the fit does not depend on n. The clamp does not enter the border contributions ($F(u_0)$ ignores l_0, $1 - F(l_K)$ ignores u_K); it matters only for exact values and for extreme user-supplied intervals.

The censoring type is derived from the score, before any clamping: $s = 0 \to \delta = 1$, $s = K \to \delta = 2$, otherwise $\delta = 3$.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Smithson, M., and Verkuilen, J. (2006). A better lemon squeezer? Maximum-likelihood regression with beta-distributed dependent variables. *Psychological Methods*, **11**(1), 54–71. doi:10.1037/1082989X.11.1.54

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

brs_prep for analyst-supplied censoring and intervals; brs; brs_sim.

Examples

```
# NRS-11 scores (K = 10) under the three interval directions
s <- 0:10
brs_check(s, ncuts = 10)                      # mid:   [s - 0.5, s + 0.5] / 10
brs_check(s, ncuts = 10, interval = "right")  # right: [s, s + 1] / 11
# "left" has the same cells as "right"; only the latent score read back differs
identical(brs_check(s, 10, interval = "left"), brs_check(s, 10, interval = "right"))

# Borders: score 0 -> delta 1 (F(u)), score 10 -> delta 2 (1 - F(l))
brs_check(c(0, 10), ncuts = 10)[, c("left", "right", "delta")]

# A Likert item 1-5: shift to 0-4 so that K = ncuts = 4 is its maximum
likert <- c(1, 2, 3, 5, 4)
brs_check(likert - 1, ncuts = 4)[, c("left", "right", "delta")]
# Unshifted, category 1 would be an interior cell and 0 a category never used
brs_check(likert, ncuts = 5)[, "delta"]

# Values already in (0, 1) are exact observations (delta 0)
brs_check(c(0.12, 0.5, 0.97))
# Per observation: 0.3 is exact, 5 and 10 are scores (with a warning)
brs_check(c(0.3, 5, 10), ncuts = 10)

# A forced delta keeps the cell of the score
brs_check(c(30, 60), ncuts = 100, delta = c(1L, 2L))
```

`brs_coef` *Internal coefficient table (deprecated, use brs_est() or summary())*

Description

Deprecated convenience wrapper. Use `brs_est` for coefficient estimates or `summary.brs` for a full model summary.

Usage

```
brs_coef(fit, alpha = 0.05)
```

Arguments

`fit`	A fitted `"brs"` object.
`alpha`	Significance level.

Value

A list with components `est` (from `brs_est`) and `gof` (from `brs_gof`).

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

`brs_est`, `brs_gof`, `summary.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
suppressWarnings(brs_coef(fit))  # deprecated; use brs_est()
```

`brs_cv` *K-fold cross-validation for brs models*

Description

Performs repeated k-fold cross-validation for `brs` models.

Usage

```
brs_cv(formula, data, k = 5L, repeats = 1L, ...)
```

Arguments

`formula`	Model formula passed to `brs`.
`data`	Data frame.
`k`	Number of folds (default 5).
`repeats`	Number of repeated k-fold runs (default 1); each repeat draws new folds with the current RNG state.
`...`	Additional arguments forwarded to `brs` (e.g., `repar`, `link`, `interval`, `method`).

Details

The `log_score` is the mean log predictive contribution under the complete likelihood contribution implied by each observation's censoring type (`delta`).

With `cutpoints = "boundary"` in `...`, held-out rows use the cells of the training fit's cutpoints. A training fold without one of the scores 0, 1, $K-1$ or K cannot be fitted: its row has `converged = FALSE` and the message in `error`.

Value

A data frame with one row per fold and columns: `repeat`, `fold`, `n_train`, `n_test`, `log_score`, `rmse_yt`, `mae_yt`, `converged`, and `error`. The object has class `"brs_cv"`.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

`brsmm_cv` for mixed models (folds by group).

Examples

```
# Synthetic NRS-11 scores: 4 groups x 3 times. Simulated, not real data.
set.seed(3)
nrs <- expand.grid(id = 1:40, time = c("6h", "12h", "24h"))
nrs$group <- factor(paste0("g", (nrs$id - 1) %% 4 + 1))
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + c(0, -0.1, 0.05, 0.1)[nrs$group]
shp <- brs_repar(mu = plogis(eta), phi = 0.3, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))

# 3-fold CV of two nested models, time only (m1) and time + group (m2);
# log_score is the mean held-out log-likelihood contribution (higher is better)
set.seed(5)
cv1 <- brs_cv(y ~ time, data = nrs, k = 3, ncuts = 10)
set.seed(5)
cv2 <- brs_cv(y ~ time + group, data = nrs, k = 3, ncuts = 10)
c(m1 = mean(cv1$log_score), m2 = mean(cv2$log_score))
cv2
```

`brs_est` *Coefficient estimates with inference*

Description

Coefficient estimates with inference

Usage

```
brs_est(object, alpha = 0.05)
```

Arguments

`object`	A fitted `"brs"` object.
`alpha`	Significance level (default 0.05).

Value

Data frame of estimates, standard errors, z-values, p-values and Wald intervals. The cutpoint parameters of a `cutpoints = "boundary"` fit get `NA` z- and p-values (their zero is not a hypothesis of interest; see `summary()` and `anova.brs`).

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

`brs`, `brs_gof`, `brs_hessian`, `summary.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
brs_est(fit)
```

brs_gof *Goodness-of-fit measures*

Description

Goodness-of-fit measures

Usage

```
brs_gof(object)
```

Arguments

object	A fitted "brs" or "brsmm" object.

Value

Data frame with logLik, AIC, BIC, and pseudo-R-squared.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

brs, brs_est, brs_hessian

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
```

```
fit <- brs(y ~ x1, data = prep)
brs_gof(fit)
```

`brs_hessian` *Extract the Hessian matrix*

Description

Extract the Hessian matrix

Usage

```
brs_hessian(object)
```

Arguments

`object`	A fitted `"brs"` or `"brsmm"` object.

Value

Numeric Hessian matrix.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

`brs`, `vcov.brs`, `brs_est`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
brs_hessian(fit)
```

`brs_loglik_contrib` *Log-likelihood contributions per observation or per group*

Description

The terms whose sum is the log-likelihood: one per observation for a `brs` fit, and one per group (cluster) for a `brsmm` fit, the marginal log-likelihood of the group's observations with its random effects integrated out. With `newdata`, the contributions of new observations at the fitted parameters (held-out scoring).

Usage

```
brs_loglik_contrib(object, newdata = NULL, int_method = NULL, n_points = NULL)
```

Arguments

`object`	A fitted `"brs"` or `"brsmm"` object.
`newdata`	Optional data frame of new observations (see Details).
`int_method, n_points`	For `brsmm` fits: integration rule (`"laplace"`, `"aghq"` or `"qmc"`) and number of nodes (nodes per dimension for `"aghq"`, points for `"qmc"`); `NULL` (default) uses those of the fit. Ignored for `brs` fits.

Details

For a `brs` fit the contribution of a score row is $\log P(S = s_i)$, the log probability of its cell (the log density for exact observations, $\delta = 0$), computed as in the compiled likelihood. For a `brsmm` fit the contribution of group g is

$$\log \int \prod_i P(S = s_{gi} \mid b)\, \varphi(b; 0, D)\, db,$$

computed by the fit's `int_method` and `n_points` (or `qmc_points`) unless `int_method` / `n_points` are given. With the defaults and `newdata = NULL`, the contributions add up to `logLik(object)`

(under "laplace", up to the tolerance of the inner mode search, about 10^{-8}). A group without a positive-definite random-effect mode, which adds a penalty value to the fitted log-likelihood, gets NA with a warning.

With newdata, rows need the response, the covariates and (for brsmm) the grouping variable; rows with missing values in them are dropped. The groups need not be groups of the fit: their random effects are integrated out, so no random-effect estimate is used. Scores are coarsened with the cells of the fit (its estimated cutpoints under cutpoints = "boundary"). For score data the contributions are log probabilities of the observed scores (no Jacobian term), on the same scale as the log-likelihood of any model that gives a probability mass function on $0, \ldots, K$.

Value

Named numeric vector: by row name of the data (brs) or by group level (brsmm, with attribute "n", the number of observations of each group).

See Also

logLik.brsmm, brsmm_cv, brs_marginal

Examples

```
set.seed(21)
nrs <- expand.grid(id = 1:30, time = c("6h", "12h", "24h"))
b0 <- rnorm(30, sd = 0.8)
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + b0[nrs$id]
shp <- brs_repar(mu = plogis(eta), phi = 0.2, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))
m <- brsmm(y ~ time, random = ~ 1 | id, data = nrs, ncuts = 10)

ll_g <- brs_loglik_contrib(m)
head(ll_g)
c(sum = sum(ll_g), logLik = as.numeric(logLik(m)))
# Same groups, adaptive Gauss-Hermite with 31 nodes
head(brs_loglik_contrib(m, int_method = "aghq", n_points = 31))
```

---

brs_marginal *Population-averaged score probabilities, expected scores and exceedance*

Description

Score distribution of a brs or brsmm fit at given covariate values, with standard errors and intervals: the probabilities $P(S = s)$ of the scores $0, \ldots, K$, the expected score or the exceedance probabilities $P(S \geq k)$, and differences of these between pairs of rows. For brsmm fits the random effects are integrated out (marginal, population-averaged quantities) or set to zero (conditional = TRUE).

Usage

```
brs_marginal(
  object,
  newdata = NULL,
  type = c("prob", "expected_score", "exceed"),
  k = NULL,
  contrast = NULL,
  conditional = FALSE,
  level = 0.95,
  se_method = c("delta", "simulation"),
  n_sim = 1000L,
  n_points = 8L
)
```

Arguments

`object`	A fitted `"brs"` or `"brsmm"` object.
`newdata`	Data frame of covariate values, one prediction per row; `NULL` (default) uses the rows of the fit.
`type`	`"prob"` (default; probabilities of the scores $0, \ldots, K$), `"expected_score"` or `"exceed"`.
`k`	Integer thresholds in $0, \ldots, K$ for `type = "exceed"`; `NULL` (default) uses $1, \ldots, K$.
`contrast`	Optional pairs of rows of `newdata` to compare: a two-column matrix with one pair per row, a list of length-2 vectors, or a single length-2 vector. The pair `c(i, j)` gives the quantity of row `i` minus that of row `j`.
`conditional`	Logical; `TRUE` sets the random effects to zero instead of integrating them out (no effect for `brs` fits).
`level`	Confidence level of the intervals (default 0.95).
`se_method`	`"delta"` (default) or `"simulation"`.
`n_sim`	Number of parameter draws for `se_method = "simulation"` (default 1000).
`n_points`	Gauss-Legendre nodes per panel of the random-effect integral (default 8; see Details).

Details

For a row with fixed-effect linear predictor η and random-effect design x_r, the marginal score probabilities are

$$P(S = s) = \int P(S = s \mid \eta + x_r^\top b)\, \varphi(b; 0, D)\, db = E\{P(S = s \mid \eta + u)\}, \qquad u \sim N(0, x_r^\top D x_r),$$

because b enters the model only through $u = x_r^\top b$ (the dispersion submodel has no random effects), so the integral is one-dimensional whatever the number of random effects. It is computed on $z = u/\mathrm{SD}(u) \in [-8.5, 8.5]$ by a composite Gauss-Legendre rule with `n_points` nodes per panel and panels of width at most 1 on the scale of z and at most 0.5 on the scale of u (the conditional probabilities can change quickly with u when the precision is high or the variance large, where

Gauss-Hermite rules converge slowly). A warning is given when halving the panel width changes a value by more than 10^{-6}.

$P(S = s \mid \cdot)$ is the probability of the cell of s under the latent beta law, with the cells of the fit's `interval` map or its estimated boundary cutpoints (as in `brs_predict_scoreprob`, but without a probability floor), so each row sums to 1 whenever the cells partition (0, 1) (`lim = 0.5`, or `interval = "right"` or `"left"`). `"expected_score"` is $\sum_s s\,P(S = s)$ on the $0, \ldots, K$ scale and `"exceed"` is $P(S \geq k) = \sum_{s \geq k} P(S = s)$. When the cells do not partition (0, 1) (`interval = "mid"` with `lim < 0.5`), the probabilities sum to less than 1 and these two types are not moments or tail probabilities of a distribution; they are computed with a warning.

Under a nonlinear link the marginal quantities differ from the conditional ones at $b = 0$: the latter describe a subject whose random effects are zero, not the population average, and the gap grows with the random-effect variance. For a `brs` fit both coincide.

Uncertainty: with `se_method = "delta"` (default), delta-method standard errors from a Richardson central-difference Jacobian of the quantities in all parameters of `coef(object)` (regression, precision, random-effect Cholesky and boundary-cutpoint parameters) and `vcov(object)`. The Wald intervals are computed on the logit scale for probabilities and for $E[S]/K$ and transformed back (so they stay in range), and on the natural scale for contrasts. With `se_method = "simulation"`, `n_sim` parameter vectors are drawn from $N(\hat{\theta}, V)$; the standard error is the standard deviation of the quantity over the draws and the interval its percentile interval (the draws use the RNG: call `set.seed()` first for reproducible values). The estimate is always the plug-in value at $\hat{\theta}$.

Rows of `newdata` with missing covariates give `NA`; the response and the grouping variable are not needed.

Value

A data frame of class `"brs_marginal"` with the columns of `newdata` (or, for contrasts, `contrast`, `row1` and `row2`), `row`, `score` (`type = "prob"`) or `k` (`type = "exceed"`), `estimate`, `std.error`, `lower` and `upper`. Attributes: `type`, `conditional`, `level`, `se_method`, `n_nodes` (total quadrature nodes; `NA` without random effects) and, for the delta method, `jacobian` (quantities by parameters).

See Also

`brsmm`, `brs_predict_scoreprob`, `predict.brsmm`, `brs_loglik_contrib`

Examples

```
# Synthetic NRS-11 scores of 40 patients at three times. Simulated data.
set.seed(21)
nrs <- expand.grid(id = 1:40, time = c("6h", "12h", "24h"))
b0 <- rnorm(40, sd = 0.8)
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + b0[nrs$id]
shp <- brs_repar(mu = plogis(eta), phi = 0.2, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))
m <- brsmm(y ~ time, random = ~ 1 | id, data = nrs, ncuts = 10)

nd <- data.frame(time = c("6h", "12h", "24h"))
# Population-averaged mean score and P(S >= 4) at each time
brs_marginal(m, nd, type = "expected_score")
brs_marginal(m, nd, type = "exceed", k = 4)
```

```
# Same at b = 0 (a typical patient): not the population average
brs_marginal(m, nd, type = "expected_score", conditional = TRUE)
# Change from 6h: 12h - 6h and 24h - 6h
brs_marginal(m, nd, type = "expected_score",
             contrast = rbind(c(2, 1), c(3, 1)))
```

`brs_marginaleffects` *Marginal effects for brs models*

Description

Computes average marginal effects (AME) for numeric covariates in the mean or precision submodel of a fitted `"brs"` object.

Usage

```
brs_marginaleffects(
  object,
  newdata = NULL,
  model = c("mean", "precision"),
  type = c("response", "link"),
  variables = NULL,
  h = 1e-05,
  interval = TRUE,
  level = 0.95,
  n_sim = 400L,
  keep_draws = FALSE
)
```

Arguments

`object`	A fitted `"brs"` object.
`newdata`	Optional data frame for evaluation; defaults to the data used in fitting.
`model`	Character; `"mean"` (default) or `"precision"`.
`type`	Character prediction scale: `"response"` (default) or `"link"`.
`variables`	Optional character vector of covariate names. Defaults to all numeric covariates in the selected submodel.
`h`	Base finite-difference step for non-binary numeric covariates (default `1e-5`); the step used is `h * max(sd(x), 1)`.
`interval`	Logical; compute interval estimates via simulation (default `TRUE`).
`level`	Confidence level for interval estimates (default 0.95).
`n_sim`	Number of parameter draws when `interval = TRUE` (default 400; at least 50).
`keep_draws`	Logical; if `TRUE` and `interval = TRUE`, stores AME simulation draws in attribute `"ame_draws"`.

Details

AMEs for a numeric covariate are computed by a central difference on predictions, with the step size scaled by the covariate's standard deviation:

$$\mathrm{AME}_j = \frac{1}{n}\sum_{i=1}^{n}\frac{\hat{g}_i(x_{ij}+h_j)-\hat{g}_i(x_{ij}-h_j)}{2h_j}, \qquad h_j = h\cdot\max(\mathrm{sd}(x_{\cdot j}),1),$$

where $\hat{g}_i$ is the selected prediction scale and `h` is the (small) base step supplied by the caller.

For binary covariates coded as `0/1`, the effect is computed as the average discrete difference $\hat{g}(x_j = 1) - \hat{g}(x_j = 0)$.

If `interval = TRUE`, uncertainty is approximated by asymptotic parameter simulation from $\mathcal{N}(\hat{\theta}, \hat{V})$.

Value

A data frame with one row per variable and columns: `variable`, `ame`, `std.error`, `ci.lower`, `ci.upper`, `model`, `type`, and `n`. The returned object has class `"brs_marginaleffects"` and attributes with analysis metadata.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

Examples

```
# Synthetic NRS-11 scores with a 0/1 time dummy and a numeric covariate (age)
set.seed(3)
nrs <- data.frame(t12 = rep(0:1, each = 60), age = round(rnorm(120, 40, 12)))
shp <- brs_repar(mu = plogis(-1.3 + 0.75 * nrs$t12 + 0.03 * (nrs$age - 40)),
                 phi = 0.3, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))
fit <- brs(y ~ t12 + age, data = nrs, ncuts = 10)

# Average marginal effects on E[Y] (0/1 dummy: discrete change; age: slope),
# with intervals from 50 draws of the estimates (use n_sim >= 400 in practice)
set.seed(6)
brs_marginaleffects(fit, n_sim = 50)
```

```
# On the 0-10 score scale (mid cells): multiply by K = 10
set.seed(6)
ame <- brs_marginaleffects(fit, n_sim = 50)
transform(ame[, c("variable", "ame", "ci.lower", "ci.upper")],
          ame = 10 * ame, ci.lower = 10 * ci.lower, ci.upper = 10 * ci.upper)
```

`brs_predict_scoreprob` *Predict score probabilities from a fitted brs model*

Description

Computes predicted probabilities for integer scores on the original scale $\{0, 1, \ldots, K\}$ implied by the fitted beta interval model.

Usage

```
brs_predict_scoreprob(
  object,
  newdata = NULL,
  scores = NULL,
  format = c("matrix", "long"),
  id_col = "id"
)
```

Arguments

`object`	A fitted `"brs"` object.
`newdata`	Optional data frame for prediction.
`scores`	Optional integer vector of scores to evaluate. Defaults to all scores in `0:ncuts`.
`format`	Output format: `"matrix"` (default) or `"long"`.
`id_col`	Column name for observation id when `format = "long"`.

Details

For a score s with cell $[l_s, u_s]$ of the fit's `interval` (see `brs_check`), $P(S = s) = F(u_s) - F(l_s)$ with $l_0 = 0$ and $u_K = 1$, where F is the beta CDF under the fitted (μ_i, ϕ_i). Under `"mid"` this is $F(\mathrm{lim}/K)$ for $s = 0$, $1 - F((K - \mathrm{lim})/K)$ for $s = K$ and $F((s + \mathrm{lim})/K) - F((s - \mathrm{lim})/K)$ otherwise; under `"right"`/`"left"` the cells are $[s, s + 1]/(K + 1)$. The rows sum to 1 whenever the cells partition $[0, 1]$ (always for `"right"`/`"left"`; `lim = 0.5` for `"mid"`). A fit with `cutpoints = "boundary"` uses its estimated cutpoints: $u_0 = l_1 = \hat{c}_{lo}$ and $u_{K-1} = l_K = \hat{c}_{hi}$.

Value

If `format = "matrix"`, a numeric matrix with one row per observation and one column per requested score.

If `format = "long"`, a long data frame with columns `id_col`, `score`, and `prob`.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  x2 = rep(c(0, 0, 1, 1), 5)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
pmat <- brs_predict_scoreprob(fit)
head(pmat[, 1:5])
plong <- brs_predict_scoreprob(fit, scores = 0:10, format = "long")
head(plong)
```

`brs_prep` *Pre-process analyst data for beta interval regression*

Description

Turns analyst data into the cells $[l_i, u_i]$ and censoring types δ_i used by `brs` and `brsmm`. Scores run over $0, 1, \ldots, K$ with $K =$ `ncuts` the maximum; shift a scale that starts at 1 (a Likert item 1–5 becomes 0–4, `ncuts = 4`). Four input modes are recognized per row:

1. **Score only** (`y`): the cell of the score and δ from the score, exactly as `brs_check` ($0 \to 1$, $K \to 2$, otherwise 3; a value in $(0, 1) \to 0$, per observation).
2. **Score and** `delta`: the analyst's censoring type, with the cell of the score.
3. **Bounds only** (`left` and/or `right`, `y` missing): an interval, or a one-sided censoring when one bound is `NA`.

4. **Score and both bounds**: the analyst's interval.

Covariate columns are kept unchanged.

Usage

```
brs_prep(
  data,
  y = "y",
  delta = "delta",
  left = "left",
  right = "right",
  ncuts = 100L,
  lim = 0.5,
  interval = c("mid", "right", "left")
)
```

Arguments

`data`	Data frame with the response columns and covariates.
`y`	Character: name of the score column (default `"y"`).
`delta`	Character: name of the censoring-type column (default `"delta"`); values in `{0, 1, 2, 3}` or `NA`.
`left`, `right`	Character: names of the lower and upper bound columns (defaults `"left"`, `"right"`), on the latent score scale.
`ncuts`	Integer K: the maximum score (default 100); the scale is $0, 1, \ldots, K$ ($K+1$ categories). Must be at least the largest `y`; bounds may reach the latent range given in Details.
`lim`	Numeric in $(0, 0.5]$: half-width of the score cell under `interval = "mid"` (default 0.5); see `brs_check`.
`interval`	Direction of the uncertainty interval, `"mid"` (default), `"right"` or `"left"`; see `brs_check`.

Details

A non-missing `delta` always wins. Otherwise the type comes from the pattern of `left`, `right` and `y` (here $K = 10$, `interval = "mid"`):

`left`	`right`	`y`	cell on $(0, 1)$	δ
NA	3	NA	$[\epsilon, 0.3]$ (below 3)	1
7	NA	NA	$[0.7, 1-\epsilon]$ (above 7)	2
2	5	NA	$[0.2, 0.5]$	3
4	6	5	$[0.4, 0.6]$ (analyst interval)	3
-0.5	4	NA	$[\epsilon, 0.4]$ (reaches 0)	1
3	10.5	NA	$[0.3, 1-\epsilon]$ (reaches 1)	2
(no columns)		5	$[0.45, 0.55]$ (Mode 1)	3
NA	NA	5	0.5 (exact reading)	0

NA	NA	0	$[\epsilon, 0.05]$	1
NA	NA	10	$[0.95, 1-\epsilon]$	2

When the data have `left`/`right` columns, a row that gives only an interior score (both bounds `NA`) is an exact value at the cell center, with a density contribution; without those columns the same score is interval-censored (Mode 1). An analyst interval that reaches 0 (or 1) on the unit scale is left- (or right-) censored; one that reaches both borders covers the whole scale, stays $\delta = 3$ and gives a warning, since such a row carries no information about the parameters. A row with neither a score nor a bound is an error (a `delta` alone defines no interval).

Score-based rows (Modes 1 and 2) use the cells of `interval`, as in `brs_check`: $[s-\lim, s+\lim]/K$ for `"mid"`, $[s, s+1]/(K+1)$ for `"right"`/`"left"`; a forced $\delta = 1$ keeps u_s and a forced $\delta = 2$ keeps l_s. Analyst bounds L (Modes 3 and 4) are values of the latent score of the chosen direction (the scale of `predict(type = "score")`) and map to L/K (`"mid"`), $L/(K+1)$ (`"right"`) or $(L+1)/(K+1)$ (`"left"`), so that the dissertation's intervals $[s-0.5, s+0.5]$, $[s, s+1]$ and $[s-1, s]$ all give the cell of score s. They must lie in the latent range of the direction: $[-0.5, K+0.5]$ (`"mid"`), $[0, K+1]$ (`"right"`) or $[-1, K]$ (`"left"`); a bound outside it is an error. Rows with no score get `y` = the latent score of their cell center, so that `model.frame()` keeps them.

Endpoints are clamped to $[\epsilon, 1-\epsilon]$, $\epsilon = 10^{-5}$ (why this replaces the edge-effect transformation of Lopes, 2023: section 'Scale change and borders' of `brs_check`). A $\delta = 3$ row with `left >= right` is an error (zero probability; use $\delta = 0$), and so is a cell the clamp squeezes to zero width (the message tells whether `ncuts` is too large for the cells or the analyst interval lies inside the clamp). As in `brs_check`, values in $(0, 1)$ mixed with values ≥ 1 give one warning. Unusual combinations, e.g. $\delta = 1$ with $y \neq 0$, give a warning but are kept.

Value

A data frame with columns `left`, `right` (cell on $(0, 1)$), `yt` (cell center), `y` (score, or the filled latent score for rows without one) and `delta`, followed by the covariates. Attributes `"is_prepared"` (`TRUE`), `"ncuts"`, `"lim"` and `"interval"` are reused by `brs` and `brsmm`; an explicit different value there is ignored with a warning.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

See Also

`brs_check` for the cells and the automatic classification; `brs` for fitting the model.

Examples

```
# Mode 1: score only; delta from the score (0 -> 1, K -> 2, else 3)
d1 <- data.frame(y = c(0, 3, 10), x = c(1.2, 0.4, -0.3))
brs_prep(d1, ncuts = 10)

# Mode 2: score + analyst delta (the same score read as exact and as a cell)
d2 <- data.frame(y = c(4, 4), delta = c(0, 3))
brs_prep(d2, ncuts = 10)
```

```
# Mode 3: only left and/or right bounds (NA pattern gives delta)
d3 <- data.frame(left = c(NA, 7, 2), right = c(3, NA, 5))
brs_prep(d3, ncuts = 10)

# Mode 4: score with analyst bounds (used as given, divided by K)
d4 <- data.frame(y = 5, left = 4, right = 6)
brs_prep(d4, ncuts = 10)

# An analyst interval reaching a border is one-sided censoring
brs_prep(data.frame(left = c(-0.5, 3), right = c(4, 10.5)), ncuts = 10)

# A Likert item 1-5: shift it to 0-4, so that ncuts = 4 is the maximum
lk <- data.frame(item = c(1, 2, 5, 3, 4), x = c(0.1, 0.5, 0.9, 0.3, 0.7))
brs_prep(data.frame(y = lk$item - 1, x = lk$x), ncuts = 4)

# Right-direction cells [s, s + 1] / 11; the choice is stored as attributes
p <- brs_prep(d1, ncuts = 10, interval = "right")
p[, c("left", "right", "delta")]
attributes(p)[c("ncuts", "lim", "interval")]
```

`brs_repar` *Beta shape parameters from the parameters of each scheme*

Description

Converts the pair (`mu`, `phi`) of one of three parameterizations into the shapes (a, b) of the beta density $f(y) = y^{a-1}(1-y)^{b-1}/B(a,b)$, $0 < y < 1$.

Usage

```
brs_repar(mu, phi, repar = 2L)
```

Arguments

`mu`	Numeric vector: the first parameter, i.e. the mean in $(0,1)$ for `repar` = `1`, `2` and the shape $p > 0$ for `repar` = `0`.
`phi`	Numeric vector (or scalar): the second parameter, i.e. the precision $\phi > 0$, the dispersion $\phi \in (0,1)$ or the shape $q > 0$.
`repar`	Integer (0, 1 or 2) selecting the scheme (default 2).

Details

`repar`	`mu`, `phi`	(a, b)	$E[Y]$	$\mathrm{Var}[Y]$
0	shapes $p, q > 0$	(p, q)	$p/(p+q)$	$pq/\{(p+q)^2(p+q+1)\}$
1	mean $\mu \in (0,1)$, precision $\phi > 0$	$(\mu\phi, (1-\mu)\phi)$	μ	$\mu(1-\mu)/(1+\phi)$
2	mean $\mu \in (0,1)$, dispersion $\phi \in (0,1)$	$(\mu\tau, (1-\mu)\tau)$, $\tau = (1-\phi)/\phi$	μ	$\phi\,\mu(1-\mu)$

The three describe the same family: $a + b$ is the precision, and the dispersion of `repar = 2` is

$$\phi = \frac{\mathrm{Var}[Y]}{\mu(1-\mu)} = \frac{1}{1+a+b},$$

the share of the largest possible variance $\mu(1-\mu)$, which is approached as $a + b \to 0$. It is not a coefficient of variation.

In Lopes (2023) these are eq. `eqn_beta_p1` (shapes p, q), "parametrizacao 1" (Ferrari and Cribari-Neto, 2004; eq. `eqn_beta_p2`) and "parametrizacao 2" (Bayer, 2011; eq. `eqn_beta_p3`), where the dispersion is written σ. The package names the second parameter `phi` in every scheme. Regression on the shapes (`repar = 0`) is a package extension. Admissible links per scheme: section 'Reparameterizations and links' of `brs`.

Value

A `data.frame` with columns `shape1` (a) and `shape2` (b), one row per element of the recycled inputs.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

Bayer, F. M. (2011). *Modelagem e inferencia em regressao beta*. PhD thesis, Universidade Federal de Pernambuco.

Examples

```
# One beta distribution in the three parameterizations: shapes (6, 14)
brs_repar(mu = 0.3, phi = 20, repar = 1)      # mean 0.3, precision 20
brs_repar(mu = 0.3, phi = 1 / 21, repar = 2)  # mean 0.3, dispersion 1/(1 + 20)
brs_repar(mu = 6, phi = 14, repar = 0)        # shapes p = 6, q = 14

# Back from shapes: mean, precision a + b, dispersion 1 / (1 + a + b)
sh <- brs_repar(mu = 0.3, phi = 20, repar = 1)
c(mean = sh$shape1 / (sh$shape1 + sh$shape2),
  precision = sh$shape1 + sh$shape2,
  dispersion = 1 / (1 + sh$shape1 + sh$shape2))

# E[Y] and Var[Y] agree across parameterizations: 0.3 and 0.21 / 21 = 0.01
mu <- 0.3
c(var_repar1 = mu * (1 - mu) / (1 + 20),
  var_repar2 = mu * (1 - mu) * (1 / 21),
  var_shapes = 6 * 14 / ((6 + 14)^2 * (6 + 14 + 1)))

# Vectorized: one row per observation
brs_repar(mu = c(0.2, 0.5, 0.8), phi = 0.1, repar = 2)
```

brs_sim *Simulate data from beta interval models*

Description

Simulates scores on $0, 1, \ldots, K$ ($K =$ `ncuts`) from a fixed- or variable-dispersion beta regression, coarsened exactly as the likelihood of `brs` assumes. The output can be passed to `brs` as it is.

Usage

```
brs_sim(
  formula,
  data,
  beta,
  phi = 1/5,
  zeta = NULL,
  link = NULL,
  link_phi = NULL,
  ncuts = 100L,
  lim = 0.5,
  repar = 2L,
  delta = NULL,
  interval = c("mid", "right", "left"),
  cutpoints = NULL
)
```

Arguments

`formula`	Model formula with one (mean) or two parts (mean | precision).
`data`	Data frame with the covariates.
`beta`	Coefficients of the first parameter, on its link scale.
`phi`	Second parameter on its link scale (one-part formulas).
`zeta`	Coefficients of the second parameter, on its link scale (two-part formulas).
`link, link_phi`	Links; `NULL` (default) selects those implied by `repar` (see `brs`).
`ncuts`	Integer K: the maximum score (default 100); scores run over $0, \ldots, K$.
`lim`	Half-width of the cell under `"mid"` (default 0.5).
`repar`	Parameterization (0, 1 or 2); see `brs_repar`.
`delta`	`NULL` (default) or one censoring type (0, 1, 2 or 3) forced on every row; see Details.
`interval`	`"mid"` (default), `"right"` or `"left"`.
`cutpoints`	`NULL` (default: the cells of `interval`) or `c(c_lo, c_hi)`, the boundary cutpoints on $(0, 1)$ between scores 0 and 1 and between $K-1$ and K; they must keep every cell non-empty ($c_{lo} < u_1$, $c_{hi} > l_{K-1}$). See Details.

Details

`formula` follows `brs`: a one-part formula (`~ x1 + x2`) uses the scalar `phi`, a two-part formula (`~ x1 + x2 | z1`) the coefficients `zeta`. A left-hand side is ignored. For each row, $Y \sim \mathrm{Beta}(a, b)$ with the shapes of `brs_repar`, and the score is $s = \mathrm{round}(KY)$ under `"mid"` (cells centered on the score; exact only for `lim = 0.5`) or $s = \lfloor (K+1)Y \rfloor$ under `"right"`/`"left"` ($K+1$ equal cells). The cells and censoring types then come from `brs_check` with the same `interval`: 0 is left-censored, K right-censored, the rest interval-censored. Censoring is therefore non-informative: the cells are fixed and do not depend on Y.

`delta` overrides the type for every row: `0` keeps the continuous Y as exact values; `3` keeps the scores away from the borders ($1 \le s \le K-1$); `1` or `2` censors every row on one side at the cell of its own score. The last two make the threshold depend on Y (informative censoring): the likelihood has no finite maximum and `brs` returns diverging coefficients, with clamp warnings. `brs_sim()` warns in that case, and also whenever all simulated rows fall on the same border.

`cutpoints = c(c_lo, c_hi)` coarsens with estimated boundary cutpoints (the model of `brs(..., cutpoints = "boundary")`): score 0 below c_{lo}, score K at or above c_{hi}, the cells of `interval` in between. It needs `delta = NULL` and `lim = 0.5`; the returned cells are those of the map, as `brs` expects for score data.

A Monte Carlo study of the estimator is a loop of `brs_sim()` and `brs()` over a fixed design; a template is in `vignette("brs-advanced-workflows")`.

Value

A data frame with columns `left`, `right`, `yt`, `y` (score), `delta` and the non-intercept columns of the model matrices, with the attributes `"is_prepared"`, `"ncuts"`, `"lim"` and `"interval"` that `brs` reuses.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

See Also

`brs`, `brs_check`, `brs_repar`, `brs_bootstrap`

Examples

```
# One design, three parameterizations with their default links
set.seed(42)
d <- data.frame(x = rnorm(300))

# repar = 2 (logit/logit): mean and dispersion on (0, 1)
s2 <- brs_sim(~ x, data = d, beta = c(0.3, 0.6), phi = qlogis(0.2), ncuts = 10)
f2 <- brs(y ~ x, data = s2)
round(cbind(true = c(0.3, 0.6, qlogis(0.2)), est = coef(f2)), 2)

# repar = 1 (logit/log): precision 20
s1 <- brs_sim(~ x, data = d, beta = c(0.3, 0.6), phi = log(20), ncuts = 10,
              repar = 1)
f1 <- brs(y ~ x, data = s1, repar = 1)
round(cbind(true = c(0.3, 0.6, log(20)), est = coef(f1)), 2)

# repar = 0 (log/log): shapes p = 3 exp(0.3 x), q = 2
s0 <- brs_sim(~ x, data = d, beta = c(log(3), 0.3), phi = log(2), ncuts = 10,
              repar = 0)
f0 <- brs(y ~ x, data = s0, repar = 0)
round(cbind(true = c(log(3), 0.3, log(2)), est = coef(f0)), 2)

# The output is ready for brs(): endpoints, delta and attributes
head(s2)
attributes(s2)[c("ncuts", "lim", "interval")]

# Right-direction cells: scores drawn as floor(11 * Y)
sr <- brs_sim(~ x, data = d, beta = c(0.3, 0.6), phi = qlogis(0.2),
              ncuts = 10, interval = "right")
table(sr$delta)

# Heaping at the borders: score 0 covers (0, 0.11), score 10 covers (0.9, 1)
sb <- brs_sim(~ x, data = d, beta = c(0.3, 0.6), phi = qlogis(0.2),
              ncuts = 10, cutpoints = c(0.11, 0.9))
table(sb$y)

# Forcing delta = 1 on every row is informative censoring: a warning
sl <- brs_sim(~ x, data = d, beta = c(0.3, 0.6), phi = qlogis(0.2),
              ncuts = 10, delta = 1)
table(sl$delta)
```

`brs_table` *Compare fitted brs models in a single table*

Description

Builds a comparison table for one or more fitted `"brs"` or `"brsmm"` objects, summarizing fit statistics and (optionally) censoring composition. Models of both classes may be mixed in the same call.

Usage

```
brs_table(
  ...,
  models = NULL,
  include_censoring = TRUE,
  sort_by = c("none", "AIC", "BIC", "logLik"),
  decreasing = FALSE,
  digits = 4L
)
```

Arguments

`...`	Fitted `"brs"` or `"brsmm"` objects passed individually.
`models`	Optional list of fitted `"brs"` or `"brsmm"` objects. Use either `...` or `models`, not both.
`include_censoring`	Logical; include censoring counts/proportions. Default is `TRUE`.
`sort_by`	Character; optional sort criterion: `"none"` (default), `"AIC"`, `"BIC"`, or `"logLik"`.
`decreasing`	Logical; sort direction when `sort_by != "none"`.
`digits`	Integer number of digits used for numeric rounding.

Value

A data frame with one row per model.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Hawker, G. A., Mian, S., Kendzerska, T., and French, M. (2011). Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF-36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP). Arthritis Care and Research, 63(S11), S240-S252. doi:10.1002/acr.20543

Hjermstad, M. J., Fayers, P. M., Haugen, D. F., et al. (2011). Studies comparing Numerical Rating Scales, Verbal Rating Scales, and Visual Analogue Scales for assessment of pain intensity in adults: a systematic literature review. Journal of Pain and Symptom Management, 41(6), 1073-1093. doi:10.1016/j.jpainsymman.2010.08.016

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
```

```
  x1 = rep(c(1, 2), 10),
  x2 = rep(c(0, 0, 1, 1), 5)
)
prep <- brs_prep(dat, ncuts = 100)
m1 <- brs(y ~ 1, data = prep)
m2 <- brs(y ~ x1, data = prep)
brs_table(null = m1, x1 = m2, sort_by = "AIC")
```

`brsmm` *Fit a mixed-effects beta interval regression model*

Description

Beta interval regression (`brs`) with Gaussian random effects in the linear predictor of the first parameter (the mean, or the shape p under `repar = 0`), fitted by marginal maximum likelihood. `random = ~ 1 | id` gives a random intercept per group, `~ 1 + x | id` a random intercept and slope with a free correlation.

Usage

```
brsmm(
  formula,
  random = ~1 | id,
  data,
  link = NULL,
  link_phi = NULL,
  repar = 2L,
  ncuts = NULL,
  lim = NULL,
  int_method = c("laplace", "aghq", "qmc"),
  n_points = 11L,
  qmc_points = 1024L,
  start = NULL,
  method = c("BFGS", "L-BFGS-B"),
  hessian_method = c("cpp", "numDeriv", "optim"),
  control = list(maxit = 2000L),
  interval = NULL,
  cutpoints = c("fixed", "boundary")
)
```

Arguments

`formula`	Model formula: `y ~ x1 + x2` or `y ~ x1 + x2 | z1 + z2` (see `brs`).
`random`	Random-effects formula `~ terms | group`, e.g. `~ 1 | id` or `~ 1 + x | id`.
`data`	Data frame (raw scores, or the output of `brs_prep`).

`link`, `link_phi`	Links for the first and second parameter; `NULL` (default) selects those implied by `repar` (see `brs`).
`repar`	Parameterization (0, 1 or 2); see `brs_repar`.
`ncuts`	Integer K: the maximum score (scale $0, \ldots, K$). `NULL` (default) uses the value stored by `brs_prep`, or 100; an explicit different value is ignored with a warning.
`lim`	Half-width of the cell in $(0, 0.5]$ (`interval = "mid"` only); `NULL` uses the stored value, or 0.5.
`int_method`	`"laplace"` (default), `"aghq"` or `"qmc"`; see Details.
`n_points`	Nodes per dimension for `"aghq"` (default 11).
`qmc_points`	Halton points for `"qmc"` (default 1024).
`start`	Optional starting vector: fixed effects, precision coefficients, then the packed Cholesky parameters.
`method`	`"BFGS"` (default) or `"L-BFGS-B"`.
`hessian_method`	`"cpp"` (default; Richardson differences of the compiled gradient), `"numDeriv"` or `"optim"`.
`control`	Control list for `optim`, merged into the default `list(maxit = 2000L)`.
`interval`	`"mid"`, `"right"` or `"left"` (see `brs_check`); `NULL` uses the stored value, or `"mid"`.
`cutpoints`	`"fixed"` (default) or `"boundary"`: also estimate the cutpoints between scores 0 and 1 and between $K-1$ and K (section 'Boundary cutpoints' of `brs`; same requirements). Their two parameters come last in `coef()`. The gradient in them is a Richardson central difference of the marginal log-likelihood (steps $10^{-3}\max(1, \lvert g\rvert)$ and half); the Hessian (`"cpp"`) adds to the compiled block the `numDeriv::jacobian()` of the compiled gradient in the cutpoint parameters and the `numDeriv::hessian()` of the marginal log-likelihood in them.

Details

For group g with observations i, the model is

$$g_1(\mu_{gi}) = x_{gi}^\top \beta + x_{r,gi}^\top b_g, \qquad g_2(\phi_{gi}) = z_{gi}^\top \gamma, \qquad b_g \sim N(0, D),$$

with the censored contributions of `brs`. The marginal log-likelihood is $\sum_g \log \int \exp\{h_g(b)\}\, db$, where $h_g(b) = \sum_i \ell_{gi}(b) + \log \varphi(b; 0, D)$. The integral is computed around the mode $\hat{b}_g$ of h_g, with $H_g = -\partial^2 h_g / \partial b\, \partial b^\top$ there:

`"laplace"` $h_g(\hat{b}_g) + \frac{q}{2}\log(2\pi) - \frac{1}{2}\log\lvert H_g\rvert$; fast, accurate when groups are not tiny.

`"aghq"` adaptive Gauss-Hermite quadrature: a product grid of `n_points` nodes per dimension at $\hat{b}_g + \sqrt{2}\, H_g^{-1/2} z$, with the symmetric square root $H_g^{-1/2}$ (`n_points`^{q} nodes, at most 500000).

`"qmc"` importance sampling from $N(\hat{b}_g, H_g^{-1})$ (nodes $\hat{b}_g + H_g^{-1/2} z$) on `qmc_points` Halton points. It is deterministic and, with two or more random effects, underestimates the log-likelihood (about -0.05 at 1024 points in the package's two-effect checks); prefer `"aghq"` for up to three random effects.

The inner mode is found by a Levenberg–Marquardt Newton method, warm-started from the modes of the previous evaluation (the cache is cleared at the start of each fit). A group whose curvature is not positive definite at its mode adds the penalty value -10^6 instead of a silently regularized term, and `brsmm()` warns.

$D = LL^\top$ is parameterized by its lower Cholesky factor: the log of each diagonal entry and the off-diagonal entries as they are (`(re_chol_logsd)_` and `(re_chol)_` in `coef()`). `summary.brsmm` reports the standard deviations and correlations with intervals; `brsmm_re_study` the intraclass correlation.

Value

An object of class `"brsmm"` with the components of a `"brs"` fit (`par`, `coefficients` with a `random` part, `value`, `hessian`, `diagnostics`, ...) and `random` (group variable, levels, conditional modes `mode_b`, `D`, `L` and the SDs `sd_b`), `ngroups`, `int_method`, `cutpoints` and `cut` (as in `brs`).

Estimation

`optim` maximizes the marginal log-likelihood with its compiled gradient: the derivative of the chosen approximation by the chain rule on the linear predictor and the implicit-function theorem at the group modes (including the movement of the quadrature nodes), with per-observation derivatives by central differences. The Hessian for `vcov()` (`hessian_method = "cpp"`, the default) is a Richardson central difference of that gradient. Starting values: `start` when given; otherwise those of `brs` for the fixed effects and $\log$ of the between-group SD of the cell centers (at least 0.1) for the random effects.

Diagnostics

The checks of `brs` ('Fit diagnostics') apply, with the compiled gradient (a central difference with step 10^{-3} if it is not finite). In addition:

"Variance component at the boundary" A random-effect term has log SD below -6, or raises the log-likelihood by less than 10^{-3} over the same fit with that SD at zero. Its variance is essentially zero; the standard error of its log SD is meaningless. Test the term with `anova.brsmm` (chi-bar-square mixture) and drop it if not needed. When the gain is negative the message adds that SD ≈ 0 has a higher log-likelihood: `optim` stopped short of the maximum.

"group(s) have no positive-definite random-effect mode" Those groups contribute the penalty value; the fit is not reliable. Simplify the random-effects structure or check the data of those groups.

`fit$diagnostics` stores `re_boundary`, `re_gain` and `inner` (number of such groups, largest gradient norm at the modes). Rank-deficient fixed-effect or random-effect design matrices are an error.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

Pinheiro, J. C., and Bates, D. M. (1995). Approximations to the log-likelihood function in the nonlinear mixed-effects model. *Journal of Computational and Graphical Statistics*, **4**(1), 12–35. doi:10.1080/10618600.1995.10474663

See Also

brs, summary.brsmm, anova.brsmm, brsmm_re_study, ranef.brsmm, predict.brsmm

Examples

```
# Synthetic NRS-11 scores (0-10) of 40 patients at 6h, 12h and 24h;
# intercepts and time slopes vary by patient. Simulated, not real data.
set.seed(21)
nrs <- expand.grid(id = 1:40, time = c("6h", "12h", "24h"))
nrs$tc <- c(-1, 0, 1)[nrs$time]                # centered time for the slope
b0 <- rnorm(40, sd = 0.8)
b1 <- rnorm(40, sd = 0.5)
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + b0[nrs$id] + b1[nrs$id] * nrs$tc
shp <- brs_repar(mu = plogis(eta), phi = 0.2, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))

# Random intercept per patient (Laplace): SD with its interval, no z-test
m1 <- brsmm(y ~ time, random = ~ 1 | id, data = nrs, ncuts = 10)
summary(m1)

# Random intercept and slope: SDs and their correlation
m2 <- brsmm(y ~ time, random = ~ 1 + tc | id, data = nrs, ncuts = 10)
summary(m2)$varcorr
head(ranef(m2))

# Is the slope needed? One added random term: the p-value uses the
# mixture 1/2 chi2(1) + 1/2 chi2(2) (see the printed heading)
anova(m1, m2)

# A known patient uses its conditional mode; a new one gets b = 0
nd <- data.frame(time = "12h", tc = 0, id = c(1, 999))
predict(m2, newdata = nd)
```

---

brsmm_cv *Group-level cross-validation for brsmm models*

Description

K-fold cross-validation of a brsmm model with folds formed by group: all observations of a group (cluster) are in the same fold. Each held-out group is scored by its marginal log-likelihood at the estimates of the fit to the other folds.

Usage

```
brsmm_cv(
  formula,
  random = ~1 | id,
  data,
  k = 10L,
  repeats = 1L,
  folds = NULL,
  seed = NULL,
  score_points = NULL,
  ...
)
```

Arguments

formula, random	Model and random-effects formulas passed to brsmm.
data	Data frame.
k	Number of folds (default 10).
repeats	Number of repeated k-fold runs (default 1).
folds	Optional fold assignment: a vector named by the group levels (read by name when its unique names are exactly the levels), or else a vector with one fold label per row of data, constant within each group. Overrides k, repeats and seed.
seed	Optional integer seed for the fold assignment.
score_points	Nodes per dimension of the adaptive Gauss-Hermite rule that scores held-out groups; NULL for the default (see Details).
...	Further arguments for brsmm (e.g. ncuts, int_method, n_points, cutpoints).

Details

The held-out log score of a fold is $\sum_g \log \int \prod_i P(S = s_{gi} \mid b)\, \varphi(b; 0, \hat{D})\, db$ over its groups, at the training-fold estimates (brs_loglik_contrib with newdata). The random effects of held-out groups are integrated out, so no random-effect estimates are needed. The integral is computed by adaptive Gauss-Hermite quadrature with score_points nodes per random-effect dimension, whatever int_method the fits use; the default is 31 for one or two random effects and otherwise the largest number whose product grid has at most 20000 nodes (27 for three, 11 for four, 7 for five; at least 2). The grid may not exceed 500000 nodes. For score data the log score is the log probability of the observed scores (no Jacobian term), so it is on the same scale as the held-out log-likelihood of any model that gives a probability mass function on $0, \ldots, K$, evaluated on the same folds (attribute "folds").

Rows with missing values in the model variables (response, covariates, random-effect terms or group) are dropped first; a group without complete rows is not used, and n_train / n_test count complete rows.

Folds: k folds of nearly equal numbers of groups, drawn at random for each repeat. With seed, the folds are drawn after set.seed(seed) and the RNG state is restored afterwards; without it, the current RNG state is used. folds gives the assignment directly (one repeat). Held-out rows use the

cells of the training fit (its estimated cutpoints under `cutpoints = "boundary"`). A fold whose fit or scoring fails (for example, a factor level or a border score missing from the training folds) has `log_score = NA` and the message in `error`.

Value

A data frame of class `"brsmm_cv"` with one row per fold: `repeat`, `fold`, `n_groups_train`, `n_groups_test`, `n_train`, `n_test`, `log_score` (sum over the held-out groups), `converged`, `error` and `warning` (messages of the fold fit). Attributes: `log_score` (total per repeat; `NA` if a fold failed), `group_scores` (held-out contribution of each group, one vector per repeat), `folds` (fold of each group, one vector per repeat, named by group level: integer for drawn folds, the type of the supplied `folds` otherwise, character for a factor) and `score_points`.

See Also

`brs_cv`, `brs_loglik_contrib`, `brsmm`

Examples

```
set.seed(21)
nrs <- expand.grid(id = 1:30, time = c("6h", "12h", "24h"))
b0 <- rnorm(30, sd = 0.8)
eta <- -1.3 + c(0, 0.75, 0.3)[nrs$time] + b0[nrs$id]
shp <- brs_repar(mu = plogis(eta), phi = 0.2, repar = 2)
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))

cv <- brsmm_cv(y ~ time, random = ~ 1 | id, data = nrs, k = 5, seed = 1,
               ncuts = 10)
cv
attr(cv, "log_score")
```

---

brsmm_re_study *Random-effects study for brsmm models*

Description

Provides a compact numeric study of random effects, including: estimated covariance matrix, correlation matrix, per-term standard deviations, empirical mean/SD of posterior modes, shrinkage ratio, and a normality check by Shapiro-Wilk (when applicable).

Usage

```
brsmm_re_study(object, ...)
```

Arguments

`object`	A fitted `"brsmm"` object.
`...`	Currently ignored.

Details

`icc` is the intraclass correlation of $\mathrm{logit}(Y)$ implied by the fitted model: for two observations of the same group with the covariates of observation i,

$$\mathrm{ICC}_i = \frac{\mathrm{Var}_b[\psi(a_i) - \psi(b_i)]}{\mathrm{Var}_b[\psi(a_i) - \psi(b_i)] + E_b[\psi_1(a_i) + \psi_1(b_i)]},$$

where $a_i(b), b_i(b)$ are the beta shapes with random part $b \sim N(0, x_{r,i}^\top D x_{r,i})$, and ψ, ψ_1 are the digamma and trigamma functions ($E[\mathrm{logit}\, Y] = \psi(a) - \psi(b)$, $\mathrm{Var}[\mathrm{logit}\, Y] = \psi_1(a) + \psi_1(b)$). The expectations over b use 40-point Gauss-Hermite quadrature and the reported value is the mean of ICC_i over the observations. The level-1 variance is that of the beta, so the value depends on the precision; with the logit link (only) and a large precision it approaches $\sigma_b^2/(\sigma_b^2 + \psi_1(a) + \psi_1(b))$. It replaces the logistic-latent formula $\sigma_b^2/(\sigma_b^2 + \pi^2/3)$, which does not describe a beta response.

The moments of $\mathrm{logit}(Y)$ over b can be infinite: with a probit link when $\sigma_b^2 \geq 1/2$, with a cloglog link for every $\sigma_b > 0$, and numerically with any link when σ_b is very large. The value would then be set by the clamp of the mean (10^{-5}), so `icc` is `NA`, with a warning, whenever $E_b[\psi_1(a) + \psi_1(b)]$ changes by more than 10% when that clamp is tightened to 10^{-8}.

Value

A list with class `"brsmm_re_study"`.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

See Also

`brsmm`, `ranef.brsmm`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
rs <- brsmm_re_study(fit)
print(rs)
rs$summary
```

`brsmm_sim` *Simulate clustered scores from a mixed beta interval model*

Description

Simulates scores on $0, 1, \ldots, K$ ($K =$ `ncuts`) from the model of `brsmm`: Gaussian random effects in the linear predictor of the first parameter, a latent beta value per row, and the coarsening of `brs_sim`. The output can be passed to `brsmm` as it is.

Usage

```
brsmm_sim(
  formula,
  random = ~1 | id,
  data,
  beta,
  phi = 1/5,
  zeta = NULL,
  re_sd,
  re_cor = NULL,
  link = NULL,
  link_phi = NULL,
  ncuts = 100L,
  lim = 0.5,
  repar = 2L,
  interval = c("mid", "right", "left"),
  cutpoints = NULL
)
```

Arguments

`formula`	Model formula with one (mean) or two parts (mean | dispersion); a left-hand side is ignored.
`random`	Random-effects formula `~ terms | group`, as in `brsmm`.
`data`	Data frame with the covariates and the grouping variable.
`beta`	Fixed-effect coefficients of the first parameter, on its link scale.
`phi`	Second parameter on its link scale (one-part formulas).
`zeta`	Coefficients of the second parameter, on its link scale (two-part formulas).
`re_sd`	Standard deviations of the random effects, one per column of the random-effects design (e.g. intercept, then slope); positive.
`re_cor`	Correlation matrix of the random effects; `NULL` (default) for independent effects. With two random effects a single correlation is accepted.
`link, link_phi`	Links; `NULL` (default) selects those implied by `repar` (see `brs`).
`ncuts`	Integer K: the maximum score (default 100).

`lim`	Half-width of the cell under `"mid"` (default 0.5).
`repar`	Parameterization (0, 1 or 2); see `brs_repar`.
`interval`	`"mid"` (default), `"right"` or `"left"`.
`cutpoints`	`NULL` (default) or `c(c_lo, c_hi)`, boundary cutpoints on $(0, 1)$, as in `brs_sim`.

Details

For group g of the grouping variable of `random`, $b_g \sim N(0, D)$ with $D = \mathrm{diag}(s)\, R\, \mathrm{diag}(s)$, $s =$ `re_sd` and $R =$ `re_cor` (the standard deviations and correlations reported by `summary.brsmm`). Row i of group g has

$$g_1(\mu_{gi}) = x_{gi}^\top \beta + x_{r,gi}^\top b_g, \qquad g_2(\phi_{gi}) = z_{gi}^\top \zeta,$$

a latent $Y_{gi} \sim \mathrm{Beta}$ with the shapes of `brs_repar`, and the score of `brs_sim`: $\mathrm{round}(KY)$ under `"mid"`, $\lfloor (K+1)Y \rfloor$ under `"right"`/`"left"`, with the boundary cutpoints `c(c_lo, c_hi)` when given. Score 0 is left-censored, K right-censored and the rest interval-censored.

The random effects are drawn first (a $G \times q$ matrix of standard normals, G groups in the order of `levels(factor(group))`, times $L^\top$, with $D = LL^\top$ the lower Cholesky factor), then the n beta values; `set.seed()` before the call makes the output reproducible, and `simulate.brsmm` uses the same sequence.

Value

`data` with the columns `left`, `right`, `yt`, `y` (score) and `delta` in front (existing columns with these names are replaced), and the attributes `"is_prepared"`, `"ncuts"`, `"lim"` and `"interval"` that `brsmm` reuses. Attribute `"ranef"`: the simulated b_g, one row per group level. Attribute `"truth"`: a list with `beta`, `phi` or `zeta`, `re_sd`, `re_cor`, `D`, `L`, `cutpoints` and `par`, the true parameters in the order and on the scale of `coef()` of a `brsmm` fit.

See Also

`brs_sim`, `brsmm`, `simulate.brsmm`

Examples

```
# 100 patients with 5 visits; random intercept and time slope
d <- expand.grid(visit = 1:5, id = 1:100)
d$tc <- (d$visit - 3) / 2
set.seed(7)
s <- brsmm_sim(~ tc, random = ~ 1 + tc | id, data = d,
               beta = c(-0.5, 0.4), phi = qlogis(0.2),
               re_sd = c(0.8, 0.3), re_cor = 0.3, ncuts = 10)
head(s)
head(attr(s, "ranef"))
attr(s, "truth")$par

# Random intercept only, boundary cutpoints (heaping at 0 and 10)
set.seed(8)
s1 <- brsmm_sim(~ tc, random = ~ 1 | id, data = d, beta = c(-0.5, 0.4),
                phi = qlogis(0.2), re_sd = 0.8, ncuts = 10,
```

```
                cutpoints = c(0.1, 0.9))
table(s1$y)

m <- brsmm(y ~ tc, random = ~ 1 | id, data = s1, cutpoints = "boundary")
round(cbind(true = attr(s1, "truth")$par, est = coef(m)), 2)
```

`coef.brs` *Extract model coefficients*

Description

Extract model coefficients

Usage

```
## S3 method for class 'brs'
coef(object, model = c("full", "mean", "precision", "cutpoints"), ...)
```

Arguments

`object`	A fitted `"brs"` object.
`model`	Character: which component to return. `"full"` (default) returns all parameters, `"mean"` returns only the mean-model coefficients, `"precision"` returns only the precision coefficients, `"cutpoints"` the logit-scale boundary cutpoint parameters of a fit with `cutpoints = "boundary"` (see `brs`).
`...`	Ignored.

Value

Named numeric vector of estimated parameters.

See Also

`brs`, `brs_est`, `vcov.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
```

```
coef(fit)
coef(fit, model = "mean")
coef(fit, model = "precision")
```

`coef.brsmm` *Extract coefficients from a brsmm fit*

Description

Extract coefficients from a brsmm fit

Usage

```
## S3 method for class 'brsmm'
coef(
  object,
  model = c("full", "mean", "precision", "random", "cutpoints"),
  ...
)
```

Arguments

`object`	A fitted `"brsmm"` object.
`model`	Character: `"full"` (default), `"mean"`, `"precision"`, `"random"` or `"cutpoints"` (fits with `cutpoints = "boundary"`).
`...`	Currently ignored.

Value

Named numeric vector.

See Also

`brsmm`, `vcov.brsmm`, `confint.brsmm`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
```

```
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
coef(fit)
coef(fit, model = "mean")
coef(fit, model = "random")
```

`confint.brs` *Wald confidence intervals*

Description

Wald intervals $\hat{\theta}_j \pm z_{1-\alpha/2} SE_j$ on the link scale, with SE_j from `vcov.brs` (Lopes, 2023, "Inferencia").

Usage

```
## S3 method for class 'brs'
confint(
  object,
  parm,
  level = 0.95,
  model = c("full", "mean", "precision", "cutpoints"),
  ...
)
```

Arguments

`object`	A fitted `"brs"` object.
`parm`	Character or integer: which parameters. If missing, all parameters of `model` are returned.
`level`	Confidence level (default 0.95).
`model`	Character: `"full"`, `"mean"`, `"precision"` or `"cutpoints"` (logit scale; see `summary()` for the cutpoints on (0, 1)).
`...`	Currently ignored.

Details

Intervals for a mean or precision on the response scale follow by the inverse link of the limits (monotone links). A limit is `NA` when the variance is not estimable (see 'Fit diagnostics' in `brs`). For small samples or parameters near the border of the scale, `brs_bootstrap` gives intervals that do not rely on the normal approximation.

Value

Matrix with the lower and upper limits.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

See Also

brs, vcov.brs, brs_bootstrap, brs_est

Examples

```
set.seed(2023)
d <- data.frame(x = runif(150))
s <- brs_sim(~ x, data = d, beta = c(-0.5, 1), phi = qlogis(0.3), ncuts = 10)
fit <- brs(y ~ x, data = s)
confint(fit)
# Mean at x = 0 on (0, 1): inverse logit of the intercept limits
plogis(confint(fit, parm = "(Intercept)"))
```

---

confint.brsmm *Wald confidence intervals for brsmm models*

Description

Wald confidence intervals for brsmm models

Usage

```
## S3 method for class 'brsmm'
confint(
  object,
  parm,
  level = 0.95,
  model = c("full", "mean", "precision", "random", "cutpoints"),
  ...
)
```

Arguments

object	A fitted "brsmm" object.
parm	Character or integer: which parameters.
level	Confidence level (default 0.95).
model	Character: "full", "mean", "precision", "random" or "cutpoints".
...	Currently ignored.

Value

Matrix with columns for lower and upper confidence bounds.

See Also

brsmm, coef.brsmm, vcov.brsmm

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
confint(fit, model = "mean")
```

---

fitted.brs *Extract fitted values*

Description

Extract fitted values

Usage

```
## S3 method for class 'brs'
fitted(object, type = c("mu", "phi"), ...)
```

Arguments

object	A fitted "brs" object.
type	Character: "mu" (default) or "phi".
...	Currently ignored.

Value

Numeric vector of fitted values.

See Also

brs, residuals.brs, predict.brs

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
head(fitted(fit))
head(fitted(fit, type = "phi"))
```

`fitted.brsmm` *Fitted values from a brsmm model*

Description

Fitted values from a brsmm model

Usage

```
## S3 method for class 'brsmm'
fitted(object, type = c("mu", "phi"), ...)
```

Arguments

`object`	A fitted `"brsmm"` object.
`type`	Character: `"mu"` (default) or `"phi"`.
`...`	Currently ignored.

Value

Numeric vector.

See Also

`brsmm`, `residuals.brsmm`, `predict.brsmm`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
```

```
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
head(fitted(fit))
head(fitted(fit, type = "phi"))
```

`formula.brs` *Extract model formula*

Description

Extract model formula

Usage

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

Arguments

`x`	A fitted `"brs"` object.
`...`	Ignored.

Value

The formula used to fit the model.

See Also

`brs`, `model.matrix.brs`, `coef.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
formula(fit)
```

`formula.brsmm` *Extract model formula*

Description

Extract model formula

Usage

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

Arguments

`x`	A fitted "`brsmm`" object.
`...`	Ignored.

Value

The formula used to fit the model.

See Also

`brsmm`, `model.matrix.brsmm`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
formula(fit)
```

`logLik.brs` *Extract log-likelihood*

Description

Extract log-likelihood

Usage

```
## S3 method for class 'brs'
logLik(object, ...)
```

Arguments

`object`	A fitted `"brs"` object.
`...`	Ignored.

Value

An object of class `"logLik"` with attributes `df` (number of estimated parameters) and `nobs` (number of observations).

See Also

`brs`, `AIC.brs`, `BIC.brs`, `brs_gof`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
logLik(fit)
```

`logLik.brsmm` *Log-likelihood for brsmm models*

Description

Log-likelihood for brsmm models

Usage

```
## S3 method for class 'brsmm'
logLik(object, ...)
```

Arguments

`object`	A fitted "`brsmm`" object.
`...`	Currently ignored.

Value

Object of class "`logLik`".

See Also

`brsmm`, `AIC.brsmm`, `BIC.brsmm`, `brs_gof`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
logLik(fit)
```

`model.matrix.brs` *Extract design matrix*

Description

Extract design matrix

Usage

```
## S3 method for class 'brs'
model.matrix(object, model = c("mean", "precision"), ...)
```

Arguments

`object`	A fitted `"brs"` object.
`model`	Character: `"mean"` (default) or `"precision"`.
`...`	Ignored.

Value

The design matrix for the specified submodel.

See Also

`brs`, `formula.brs`, `coef.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
head(model.matrix(fit))
head(model.matrix(fit, model = "precision"))
```

`model.matrix.brsmm` *Extract design matrix*

Description

Extract design matrix

Usage

```
## S3 method for class 'brsmm'
model.matrix(object, model = c("mean", "precision", "random"), ...)
```

Arguments

`object`	A fitted `"brsmm"` object.
`model`	Character: `"mean"` (default), `"precision"`, or `"random"`.
`...`	Ignored.

Value

The design matrix for the specified submodel.

See Also

`brsmm`, `formula.brsmm`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
head(model.matrix(fit))
head(model.matrix(fit, model = "random"))
```

nobs.brs *Number of observations*

Description

Number of observations

Usage

```
## S3 method for class 'brs'
nobs(object, ...)
```

Arguments

object	A fitted "brs" object.
...	Ignored.

Value

Integer: number of observations.

See Also

brs, fitted.brs, brs_gof

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
nobs(fit)
```

nobs.brsmm *Number of observations in a brsmm fit*

Description

Number of observations in a brsmm fit

Usage

```
## S3 method for class 'brsmm'
nobs(object, ...)
```

Arguments

object	A fitted "brsmm" object.
...	Currently ignored.

Value

Integer.

See Also

brsmm, fitted.brsmm

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
nobs(fit)
```

plot.brs *Diagnostic plots for beta interval regression*

Description

Produces up to six diagnostic plots for a fitted "brs" model: residuals vs indices, Cook's distance, residuals vs linear predictor, residuals vs fitted values, a half-normal plot with simulated envelope, and predicted vs observed.

Usage

```
## S3 method for class 'brs'
plot(
  x,
  which = 1:4,
  type = "rqr",
  nsim = 100L,
  level = 0.9,
 caption = c("Residuals vs indices", "Cook's distance", "Residuals vs linear predictor",
    "Residuals vs fitted values", "Half-normal plot", "Predicted vs observed"),
  sub.caption = NULL,
  ask = prod(par("mfcol")) < length(which) && dev.interactive(),
  gg = FALSE,
  title = NULL,
  theme = NULL,
  ...
)
```

Arguments

x	A fitted "brs" object.
which	Integer vector selecting which plots to draw (default 1:4).
type	Character: residual type passed to residuals.brs (default "rqr").
nsim	Integer: number of simulations for the half-normal envelope (default 100).
level	Numeric: confidence level for the envelope (default 0.9).
caption	Character vector of panel captions.
sub.caption	Subtitle; defaults to the model call.
ask	Logical: prompt before each page of plots?
gg	Logical: use ggplot2? (default FALSE).
title	Optional global title for ggplot output. If NULL, panel captions are used.
theme	Optional ggplot2 theme object (e.g., ggplot2::theme_bw()). If NULL, a minimal theme is used.
...	Further arguments passed to base plot().

Value

Invisibly returns `x`.

See Also

`brs`, `residuals.brs`, `autoplot.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
plot(fit, which = 1:4)
```

`plot.brsmm` *Diagnostic plots for mixed beta interval regression*

Description

Produces diagnostic plots for fitted `"brsmm"` models: residuals vs indices, Cook's distance, residuals vs linear predictor, residuals vs fitted values, half-normal envelope, and predicted vs observed.

Usage

```
## S3 method for class 'brsmm'
plot(
  x,
  which = 1:4,
  type = c("response", "pearson"),
  nsim = 100L,
  level = 0.9,
 caption = c("Residuals vs indices", "Cook's distance", "Residuals vs linear predictor",
    "Residuals vs fitted values", "Half-normal plot", "Predicted vs observed",
    "Random-effects Q-Q", "Random-effects caterpillar"),
  sub.caption = NULL,
  ask = prod(par("mfcol")) < length(which) && dev.interactive(),
  gg = FALSE,
  title = NULL,
  theme = NULL,
  ...
)
```

Arguments

x	A fitted "brsmm" object.
which	Integer vector selecting which panels to draw (default 1:4).
type	Residual type passed to residuals.brsmm ("response" or "pearson").
nsim	Number of simulations for half-normal envelope.
level	Confidence level for the half-normal envelope.
caption	Character vector of plot captions.
sub.caption	Optional subtitle; defaults to model call.
ask	Logical: prompt before each new page?
gg	Logical: use ggplot2 backend?
title	Optional global title for ggplot output. If NULL, panel captions are used.
theme	Optional ggplot2 theme object (e.g., ggplot2::theme_bw()). If NULL, a minimal theme is used.
...	Further arguments passed to base plot().

Value

Invisibly returns x.

See Also

brsmm, residuals.brsmm, autoplot.brsmm

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
plot(fit, which = 1:4)
```

predict.brs *Predict from a fitted model*

Description

Predict from a fitted model

Usage

```
## S3 method for class 'brs'
predict(
  object,
  newdata = NULL,
  type = c("response", "link", "precision", "variance", "quantile", "score",
    "expected_score"),
  at = 0.5,
  ...
)
```

Arguments

object	A fitted "brs" object.
newdata	Optional data frame for prediction.
type	Prediction type: "response" (default; the mean $E[Y] = a/(a+b)$), "link" (linear predictor of the first parameter), "precision" (second parameter on its own scale), "variance", "quantile", "score" or "expected_score". "score" is the latent continuous score of the fit's interval at $y^* = E[Y]$: Ky^* on $(0, K)$ for "mid", $(K+1)y^*$ on $(0, K+1)$ for "right" and $(K+1)y^* - 1$ on $(-1, K)$ for "left" (it can be negative); under "right"/"left" it is about 0.5 above/below the expected recorded score, because a recorded score is the lower/upper end of its latent interval. "expected_score" is the expected recorded score $\sum_s s\,P(S = s)$ (brs_predict_scoreprob), the same for "right" and "left", and a proper expectation only when the cells partition $[0, 1]$.
at	Numeric vector of probabilities for quantile predictions (default 0.5).
...	Currently ignored.

Value

Numeric vector or matrix.

See Also

brs, fitted.brs, brs_predict_scoreprob

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
head(predict(fit))
head(predict(fit, type = "precision"))
newdat <- data.frame(x1 = c(1, 2))
predict(fit, newdata = newdat)
```

---

predict.brsmm *Predict from a brsmm model*

Description

Predict from a brsmm model

Usage

```
## S3 method for class 'brsmm'
predict(
  object,
  newdata = NULL,
  type = c("response", "link", "precision", "variance", "quantile", "score",
    "expected_score"),
  at = 0.5,
  ...
)
```

Arguments

object	A fitted "brsmm" object.
newdata	Optional data frame.
type	Character: "response" (default), "link", "precision", "variance", "quantile", "score" or "expected_score". "score" is the latent score of the fit's interval at the conditional mean (support $(0, K)$, $(0, K+1)$ or $(-1, K)$; about 0.5 above/below the expected recorded score under "right"/"left"); "expected_score" is the expected recorded score $\sum_s s\, P(S = s)$. Details: predict.brs.
at	Numeric vector of probabilities for quantile predictions (default 0.5).
...	Currently ignored.

Value

Numeric vector, except when `type = "quantile"` and `at` has length greater than 1, in which case a numeric matrix with one column per requested quantile (named `q_<value>`, e.g. `"q_0.5"`) and one row per observation.

See Also

`brsmm`, `fitted.brsmm`, `brs_predict_scoreprob`; `brs_marginal` for population-averaged (marginal) quantities with standard errors

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
head(predict(fit))
head(predict(fit, type = "precision"))
```

---

print.brs *Print a fitted model (brief betareg style)*

Description

Print a fitted model (brief betareg style)

Usage

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

Arguments

`x`	A fitted `"brs"` object.
`digits`	Number of significant digits.
`...`	Included for consistency with generic methods. Currently passed to internal methods where applicable.

Value

Invisibly returns the input object x. The function is called for its side effect of printing a formatted summary of the fitted model to the console, including the model call, mean coefficients (with link function), and precision coefficients (with link function).

See Also

summary.brs, print.summary.brs, brs

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
print(fit)
```

---

print.brsmm *Print a fitted brsmm model*

Description

Print a fitted brsmm model

Usage

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

Arguments

x	A fitted "brsmm" object.
digits	Number of digits.
...	Included for consistency with generic methods.

Value

Invisibly returns x.

See Also

summary.brsmm, print.summary.brsmm, brsmm

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
print(fit)
```

---

print.brsmm_re_study *Print a random-effects study*

Description

Prints a compact summary of the random-effects study returned by brsmm_re_study, including per-term standard deviations, shrinkage ratios, Shapiro-Wilk p-values, and the estimated covariance and correlation matrices.

Usage

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

Arguments

x	A "brsmm_re_study" object returned by brsmm_re_study.
digits	Integer: number of significant digits for rounding (default max(3, getOption("digits") - 3)).
...	Currently ignored.

Value

Invisibly returns x. Called for its side-effect of printing the study to the console.

See Also

brsmm_re_study, brsmm, ranef.brsmm

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
rs <- brsmm_re_study(fit)
print(rs)
```

---

print.summary.brs *Print a model summary (betareg style)*

Description

Print a model summary (betareg style)

Usage

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

Arguments

x	A "summary.brs" object.
digits	Number of digits.
...	Passed to printCoefmat.

Value

Invisibly returns the input object x. The function is called for its side effect of printing a comprehensive summary to the console, including the model call, quantile residuals, coefficient tables for mean and precision submodels with significance stars, goodness-of-fit statistics (log-likelihood, pseudo R-squared), optimization details, and censoring information.

See Also

summary.brs, brs, print.brs

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
print(summary(fit))
```

`print.summary.brsmm` *Print summary for brsmm models*

Description

Print summary for brsmm models

Usage

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

Arguments

`x`	A "`summary.brsmm`" object.
`digits`	Number of digits.
`...`	Passed to `printCoefmat`.

Value

Invisibly returns `x`.

See Also

`summary.brsmm`, `brsmm`, `print.brsmm`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
```

```
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
print(summary(fit))
```

`ranef` *Extract random effects*

Description

Generic function for extracting random effects.

Usage

```
ranef(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments passed to methods.

Value

Method-specific; for `"brsmm"` objects, a matrix or named numeric vector of group-specific random-effect modes.

See Also

`ranef.brsmm`, `brsmm_re_study`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
ranef(fit)
```

`ranef.brsmm` *Extract random effects from a brsmm model*

Description

Extract random effects from a brsmm model

Usage

```
## S3 method for class 'brsmm'
ranef(object, ...)
```

Arguments

`object`	A fitted "brsmm" object.
`...`	Currently ignored.

Value

A matrix or named numeric vector of group-specific random-effect posterior modes.

See Also

`brsmm`, `brsmm_re_study`, `ranef`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
ranef(fit)
```

`residuals.brs` *Residuals of a fitted beta interval model*

Description

Residuals of a `"brs"` fit. Randomized quantile residuals (`type = "rqr"`) use the censoring of each observation and are the recommended ones; the other types are evaluated at one point of the cell (see Details).

Usage

```
## S3 method for class 'brs'
residuals(
  object,
  type = c("response", "pearson", "deviance", "rqr", "weighted", "sweighted",
    "cell_deviance"),
  ...
)
```

Arguments

`object`	A fitted `"brs"` object.
`type`	Residual type: `"response"` (default), `"pearson"`, `"deviance"`, `"rqr"`, `"weighted"`, `"sweighted"` or `"cell_deviance"`.
`...`	Currently ignored.

Details

Let (a_i, b_i) be the fitted shapes, $\hat{\mu}_i = a_i/(a_i + b_i)$ the fitted mean and V_i the fitted variance (`brs_repar`). All types except `"rqr"` use $y_i =$ `yt`, the center of the cell: s/K under `interval = "mid"`, so that the border scores sit at 10^{-5} and $1-10^{-5}$, and $(s+0.5)/(K+1)$ under `"right"`/`"left"`; exact values are used as they are. With `cutpoints = "boundary"` scores 0, 1, $K-1$ and K use the centers of their estimated cells (see `brs`); since the centers of scores 0 and K differ by construction, these residuals and the pseudo R^2 are not directly comparable between fixed-map and boundary fits. This is the midpoint convention of Lopes (2023, "Analise de residuos"); it makes these residuals unreliable at the borders of the scale.

`"response"` $y_i - \hat{\mu}_i$.

`"pearson"` $(y_i - \hat{\mu}_i)/\sqrt{V_i}$. Lopes (2023) writes V_i as $\mu(1-\mu)/(1+\phi)$ (parameterization 1); it is the same variance under every `repar`.

`"deviance"` $\mathrm{sign}(y_i - \hat{\mu}_i)\sqrt{|2\{\ell_i(y_i) - \ell_i(\hat{\mu}_i)\}|}$, where $\ell_i(m)$ is the beta log-density at y_i with mean m and precision $a_i + b_i$ (Ferrari and Cribari-Neto, 2004). The saturated mean is taken as y_i, as in **betareg**. For a small precision (U- or J-shaped densities) $\ell_i(y_i)$ can be below $\ell_i(\hat{\mu}_i)$; the absolute value is then used and only the sign carries information.

`"rqr"` $\Phi^{-1}(u_i)$, with u_i uniform on $(F(l_i), F(u_i))$ for $\delta_i = 3$, on $(0, F(u_i))$ for $\delta_i = 1$ and on $(F(l_i), 1)$ for $\delta_i = 2$, and $u_i = F(y_i)$ for exact values (Dunn and Smyth, 1996); u_i is kept in $[10^{-10}, 1 - 10^{-10}]$. They are standard normal under the model whatever the censoring. They are random: set a seed to reproduce them; `summary()` draws them without changing the caller's RNG state.

`"weighted"`, `"sweighted"` $(y_i^* - \mu_i^*)/\sqrt{(a_i + b_i)v_i}$ and $(y_i^* - \mu_i^*)/\sqrt{v_i}$, with $y_i^* = \mathrm{logit}(y_i)$, $\mu_i^* = \psi(a_i) - \psi(b_i)$ and $v_i = \psi'(a_i) + \psi'(b_i)$ (Espinheira, Ferrari and Cribari-Neto, 2008).

`"cell_deviance"` Deviance residuals of the censored likelihood: $d_i = \mathrm{sign}_i\sqrt{-2\log P_i}$, with P_i the fitted probability of the observed cell (the contribution of row i to `logLik()`). The saturated model gives each cell probability 1, so $-2\log P_i$ is the deviance contribution and, for score data, $\sum_i d_i^2 = -2\,\mathrm{logLik}$. $\mathrm{sign}_i = +1$ when the observed score is at least the expected score $\sum_s s\,P(S = s)/\sum_s P(S = s)$ (the denominator is 1 unless the cells leave gaps, `interval = "mid"` with `lim < 0.5`) and -1 otherwise; censored rows that are not score cells (analyst intervals) compare the cell center with the fitted mean $E[Y_i]$. Exact rows ($\delta_i = 0$) give `NA` (use `"deviance"`).

Lopes (2023) also recommends the adjusted quantile residuals of Pereira (2019), which are not implemented.

Value

Numeric vector of residuals, one per observation.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Dunn, P. K., and Smyth, G. K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics*, **5**(3), 236–244.

Espinheira, P. L., Ferrari, S. L. P., and Cribari-Neto, F. (2008). On beta regression residuals. *Journal of Applied Statistics*, **35**(4), 407–419.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

Pereira, G. H. A. (2019). On quantile residuals in beta regression. *Communications in Statistics - Simulation and Computation*, **48**(1), 302–316.

See Also

`brs`, `fitted.brs`, `plot.brs`, `residuals.brsmm`

Examples

```
# Synthetic NRS-11 scores: 3 post-operative times. Simulated, not real data.
set.seed(2023)
nrs <- data.frame(time = factor(rep(c("6h", "12h", "24h"), each = 80),
                                levels = c("6h", "12h", "24h")))
shp <- brs_repar(mu = plogis(-1.3 + c(0, 0.75, 0.3)[nrs$time]), phi = 0.3,
                 repar = 2)
```

```
nrs$y <- round(10 * rbeta(nrow(nrs), shp$shape1, shp$shape2))
fit <- brs(y ~ time, data = nrs, ncuts = 10)

# Randomized quantile residuals: approximately N(0, 1), borders included
set.seed(1)
r_q <- residuals(fit, type = "rqr")
qqnorm(r_q); qqline(r_q)

# Midpoint-based residuals are extreme at the border scores 0 and 10
r_p <- residuals(fit, type = "pearson")
tapply(r_p, cut(nrs$y, c(-1, 0, 9, 10), labels = c("0", "1-9", "10")), mean)

# Deviance residuals from the cell probabilities add up to -2 logLik
r_d <- residuals(fit, type = "cell_deviance")
c(sum(r_d^2), -2 * as.numeric(logLik(fit)))
```

`residuals.brsmm` *Residuals from a brsmm model*

Description

Residuals of a `"brsmm"` fit. The types of `residuals.brs` are evaluated at the conditional modes of the random effects (`ranef.brsmm`); for `"rqr"` and `"cell_deviance"`, `re` selects how the random effects enter.

Usage

```
## S3 method for class 'brsmm'
residuals(
  object,
  type = c("response", "pearson", "deviance", "rqr", "weighted", "sweighted",
    "cell_deviance"),
  re = c("conditional", "marginal", "posterior"),
  n_draws = 200L,
  ...
)
```

Arguments

`object`	A fitted `"brsmm"` object.
`type`	Character: `"response"` (default), `"pearson"`, `"deviance"`, `"rqr"`, `"weighted"`, `"sweighted"` or `"cell_deviance"` (see `residuals.brs`).
`re`	`"conditional"` (default), `"marginal"` or `"posterior"`; for `type = "rqr"` and `"cell_deviance"` only (see Details).
`n_draws`	Number of importance-sampling proposals per group for `re = "posterior"` (default 200; 1 keeps the Gaussian draw).
`...`	Currently ignored.

Details

`re` applies to `type = "rqr"` and `"cell_deviance"`:

`"conditional"` (default) at the conditional modes $\hat{b}_g$. The modes are shrunk toward zero, so the `"rqr"` residuals are not exactly $N(0,1)$ under the model; the shrinkage is strongest for small groups.

`"marginal"` from the marginal (population-averaged) distribution of each observation, with the random effects integrated out: $v_i \sim U(F_m(l_i), F_m(u_i))$, $F_m(y) = \int F(y \mid b)\,\varphi(b; 0, \hat{D})\,db$, computed on the quadrature rule of `brs_marginal` (for score data $F_m(l_i)$ and $F_m(u_i)$ are the marginal cdf of the score at $s_i - 1$ and s_i). The rule integrates $z = u/\mathrm{SD}(u)$ over $[-8.5, 8.5]$, so marginal probabilities below about 10^{-15} are approximate (this matters for `"cell_deviance"` in extreme tails only). Each `"rqr"` residual is $N(0,1)$ under the model; residuals of the same group are correlated, since they share the group's random effects.

`"posterior"` at one draw of each b_g from its posterior given the group's data, by sampling-importance-resampling: `n_draws` proposals from $N(\hat{b}_g, H_g^{-1})$ (H_g, the curvature of the log posterior at the mode, by central differences; the prior $N(0, \hat{D})$ shape when it is not positive definite) weighted by posterior over proposal density, and one of them kept. At the true parameters, (b_g, y_g) with b_g drawn from its posterior has the joint law of the model, so the `"rqr"` residuals are independent $N(0,1)$ (up to the resampling approximation, of order `1/n_draws`) without the shrinkage of the modes; at the estimates this holds approximately. Unlike the marginal residuals, the residuals of a group are then nearly independent.

`"rqr"` residuals use v_i uniform on $(F(l_i), F(u_i))$ (survival scale above the median, to keep the precision of upper-tail cells), kept in $[10^{-10}, 1-10^{-10}]$. `"cell_deviance"` residuals are $\mathrm{sign}_i\sqrt{-2\log P_i}$ with the conditional, marginal or posterior-draw cell probability P_i and expected score $\sum_s sP(S = s)/\sum_s P(S = s)$, as in `residuals.brs`. The `"rqr"` residuals and the `"posterior"` draws are random: set a seed to reproduce them.

Value

Numeric vector, one value per observation.

See Also

`brsmm`, `fitted.brsmm`, `plot.brsmm`, `residuals.brs`

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
head(residuals(fit))
```

```
head(residuals(fit, type = "pearson"))

# Quantile residuals without the shrinkage of the conditional modes
d <- expand.grid(visit = 1:5, id = 1:60)
set.seed(3)
s <- brsmm_sim(~ visit, random = ~ 1 | id, data = d, beta = c(-0.6, 0.1),
               phi = qlogis(0.2), re_sd = 0.7, ncuts = 10)
m <- brsmm(y ~ visit, random = ~ 1 | id, data = s)
set.seed(1)
r_m <- residuals(m, type = "rqr", re = "marginal")
r_p <- residuals(m, type = "rqr", re = "posterior")
round(c(mean_m = mean(r_m), sd_m = sd(r_m), mean_p = mean(r_p), sd_p = sd(r_p)), 2)
```

`simulate.brs` *Simulate responses from a fitted beta interval model*

Description

Draws new responses from a `brs` or `brsmm` fit at the estimated parameters, coarsened with the cells of the fit, as `brs_sim` and `brsmm_sim` do.

Usage

```
## S3 method for class 'brs'
simulate(object, nsim = 1, seed = NULL, ...)

## S3 method for class 'brsmm'
simulate(object, nsim = 1, seed = NULL, re = c("marginal", "conditional"), ...)
```

Arguments

`object`	A fitted `"brs"` or `"brsmm"` object.
`nsim`	Number of simulated response vectors (default 1).
`seed`	`NULL` or an integer seed, as in `simulate`: with a seed the caller's RNG state is restored afterwards.
`...`	Currently ignored.
`re`	For `brsmm` fits: `"marginal"` (default; new random effects in each simulation) or `"conditional"` (random effects at the conditional modes).

Details

Each simulation draws one latent beta value per row with the fitted shapes and returns its score on $0, \ldots, K$ (cells of the fit's `interval` map, or its estimated boundary cutpoints); rows fitted as exact values ($\delta = 0$) get the latent value itself.

For a `brsmm` fit, `re = "marginal"` (default) draws new random effects $b_g \sim N(0, \hat{D})$ for each simulation (new groups with the same design: the marginal distribution of the data), in the order of

brsmm_sim, so simulate(fit, seed = s) and set.seed(s); brsmm_sim(...) with the estimates as true values give the same scores. re = "conditional" keeps the random effects at their conditional modes (ranef.brsmm) and draws only the latent values: the same groups again.

Value

A data frame with nsim columns sim_1, ..., one row per observation of the fit, and the attribute "seed" (as simulate). For brsmm fits with re = "marginal", the attribute "ranef" holds the random effects of each simulation (a list of matrices, one row per group).

See Also

brs_sim, brsmm_sim

Examples

```
set.seed(2023)
d <- data.frame(x = runif(150))
s <- brs_sim(~ x, data = d, beta = c(-0.5, 1), phi = qlogis(0.3), ncuts = 10)
fit <- brs(y ~ x, data = s)
sims <- simulate(fit, nsim = 200, seed = 1)
# Observed and simulated frequencies of the scores
rbind(observed = tabulate(s$y + 1, 11) / nrow(s),
      simulated = tabulate(unlist(sims) + 1, 11) / (200 * nrow(s)))
```

---

summary.brs *Summarize a fitted beta interval model*

Description

Wald tables for the mean and precision (or dispersion) coefficients, information criteria, a pseudo R^2, the censoring counts and a summary of the randomized quantile residuals.

Usage

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

Arguments

object	A fitted "brs" object.
...	Currently ignored.

Details

For each coefficient $\hat{\theta}_j$, the standard error is $SE_j = \sqrt{[(-H)^{-1}]_{jj}}$ from `vcov.brs`, the Wald statistic is $z_j = \hat{\theta}_j / SE_j$ and the two-sided p-value is $2\Phi(-|z_j|)$ (Lopes, 2023, "Inferencia"). The test is on the link scale ($H_0 : \theta_j = 0$). When $-H$ is singular, or its inverse has negative variances, the affected standard errors, statistics and p-values are `NA` (see 'Fit diagnostics' in `brs`).

$\mathrm{AIC} = -2\ell + 2k$ and $\mathrm{BIC} = -2\ell + k \log n$, with ℓ the maximized log-likelihood, k the number of coefficients and n the number of observations. The pseudo R^2 is the squared correlation between the fitted linear predictor of the mean and $g_1(y_i)$ at the cell centers `yt` (Ferrari and Cribari-Neto, 2004); under `repar = 0` both sides are on the logit scale. It uses the cell centers, so it is rough when most observations are censored (the print says so).

The randomized quantile residuals (`residuals.brs`, `type = "rqr"`) are drawn without changing the caller's RNG state.

Value

A list of class `"summary.brs"` with `coefficients` (tables `mean` and `precision` with columns `Estimate`, `Std. Error`, `z value`, `Pr(>|z|)`), `residuals` (RQR), `loglik`, `AIC`, `BIC`, `df`, `nobs`, `pseudo.r2`, `censoring` (counts by type), `link`, `link_phi`, `repar`, `convergence` and `iterations`.

References

Lopes, J. E. (2023). *Modelos de regressao beta para dados de escala*. Master's dissertation, Universidade Federal do Parana, Curitiba. URI: https://acervodigital.ufpr.br/xmlui/handle/1884/86624.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31**(7), 799–815. doi:10.1080/0266476042000214501

See Also

`brs`, `confint.brs`, `anova.brs`, `brs_gof`

Examples

```
set.seed(2023)
d <- data.frame(time = factor(rep(c("6h", "12h", "24h"), each = 60),
                              levels = c("6h", "12h", "24h")))
shp <- brs_repar(mu = plogis(-1.3 + c(0, 0.75, 0.3)[d$time]), phi = 0.3)
d$y <- round(10 * rbeta(nrow(d), shp$shape1, shp$shape2))
fit <- brs(y ~ time, data = d, ncuts = 10)
s <- summary(fit)
s
s$coefficients$mean
c(AIC = s$AIC, BIC = s$BIC, pseudo_R2 = s$pseudo.r2)
```

summary.brsmm *Summarize a fitted brsmm model*

Description

Wald tests for the fixed effects. The random effects are reported as standard deviations and correlations (varcorr) with Wald intervals built on a transformed scale and mapped back: $\exp$ of the interval for $\log SD$, $\tanh$ of the interval for $\operatorname{atanh}(\rho)$ (delta method from the packed Cholesky parameters). No test or p-value is given for them: a z-test of $\log SD$ tests $SD = 1$, and $SD = 0$ lies on the boundary; use anova.brsmm (chi-bar-square mixture) against the model without the term. The randomized quantile residuals are drawn without changing the caller's RNG state.

Usage

```
## S3 method for class 'brsmm'
summary(object, level = 0.95, ...)
```

Arguments

object	A fitted "brsmm" object.
level	Confidence level of the varcorr intervals.
...	Currently ignored.

Value

Object of class "summary.brsmm"; coefficients$random holds the packed Cholesky parameters (estimate and standard error only) and varcorr the SD/correlation table.

See Also

brsmm, print.summary.brsmm, brs_gof, brsmm_re_study

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
s <- summary(fit)
s$coefficients$mean
```

vcov.brs *Variance-covariance matrix of estimated coefficients*

Description

Variance-covariance matrix of estimated coefficients

Usage

```
## S3 method for class 'brs'
vcov(object, model = c("full", "mean", "precision", "cutpoints"), ...)
```

Arguments

object	A fitted "brs" object.
model	Character: which component ("full", "mean", "precision" or "cutpoints").
...	Ignored.

Details

$(-H)^{-1}$ with H the Hessian of the log-likelihood at the estimate. No generalized inverse is used: a singular Hessian gives an NA matrix, and negative or non-finite variances become NA (row and column); both cases warn (see fit$diagnostics).

Value

A square numeric matrix.

See Also

brs, coef.brs, confint.brs

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10)
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brs(y ~ x1, data = prep)
vcov(fit)
vcov(fit, model = "mean")
```

vcov.brsmm *Variance-covariance matrix for brsmm coefficients*

Description

Variance-covariance matrix for brsmm coefficients

Usage

```
## S3 method for class 'brsmm'
vcov(
  object,
  model = c("full", "mean", "precision", "random", "cutpoints"),
  ...
)
```

Arguments

object	A fitted "brsmm" object.
model	Character: "full", "mean", "precision", "random" or "cutpoints".
...	Currently ignored.

Value

Numeric matrix.

See Also

brsmm, coef.brsmm, confint.brsmm

Examples

```
dat <- data.frame(
  y = c(
    0, 5, 20, 50, 75, 90, 100, 30, 60, 45,
    10, 40, 55, 70, 85, 25, 35, 65, 80, 15
  ),
  x1 = rep(c(1, 2), 10),
  id = factor(rep(1:4, each = 5))
)
prep <- brs_prep(dat, ncuts = 100)
fit <- brsmm(y ~ x1, random = ~ 1 | id, data = prep)
vcov(fit, model = "mean")
```

Index

AIC.brs, 3, *15*, *69*
AIC.brsmm, 4, *16*, *70*
anova.brs, 5, *7*, *20–22*, *28*, *33*, *93*
anova.brsmm, *6*, 7, *54, 55*, *94*
autoplot.brs, 8, *11, 12*, *76*
autoplot.brs_bootstrap, *9*, 10
autoplot.brs_marginaleffects, 12
autoplot.brsmm, 13, *77*

BIC.brs, *4*, 15, *69*
BIC.brsmm, *5*, *15*, 16, *70*
brs, *4*, *6*, *9*, *11, 12*, *15*, 17, *17, 18*, *23, 24*, *27–29*, *31*, *33–37*, *43*, *45*, *47–49*, *52–55*, *59*, *61*, *63–65*, *67*, *69*, *71*, *73*, *76*, *78*, *81*, *83*, *87, 88*, *91*, *93*, *95*
brs_bootstrap, *10, 11*, *20, 21*, 22, *49*, *63, 64*
brs_cens, 25
brs_check, *18*, *20*, *22*, *25*, 27, *42–45*, *49*, *53*
brs_coef, 30
brs_cv, *21*, 31, *57*
brs_est, *30*, 32, *34, 35*, *61*, *64*
brs_fit_fixed, *17*
brs_fit_var, *17*
brs_gof, *4, 5*, *15, 16*, *30*, *33*, 34, *69, 70*, *73*, *93, 94*
brs_hessian, *33, 34*, 35
brs_loglik_contrib, 36, *39*, *56, 57*
brs_marginal, 37, *37*, *80*, *90*
brs_marginaleffects, *12*, 40
brs_predict_scoreprob, *22*, *39*, 42, *78*, *80*
brs_prep, *17, 18*, *21–23*, *25*, *27–29*, 43, *52, 53*
brs_repar, *17, 18*, *20*, *22*, 46, *48, 49*, *53*, *60*, *87*
brs_sim, *17, 18*, *21*, *23, 24*, *27–29*, 48, *59, 60*, *91, 92*
brs_table, 50
brsmm, *5*, *7*, *16*, *20*, *22*, *36, 37*, *39*, *43*, *45*, 52, *55–60*, *62*, *65, 66*, *68*, *70*, *72*, *74*, *77*, *80–82*, *84*, *86*, *90, 91*, *94*, *96*
brsmm_cv, *32*, *37*, 55
brsmm_re_study, *54, 55*, 57, *82*, *85, 86*, *94*
brsmm_sim, 59, *91, 92*

coef.brs, 61, *67*, *71*, *95*
coef.brsmm, 62, *65*, *96*
confint.brs, *24*, 63, *93*, *95*
confint.brsmm, *62*, 64, *96*

fitted.brs, 65, *73*, *78*, *88*
fitted.brsmm, 66, *74*, *80*, *90*
Formula, *17*
formula.brs, 67, *71*
formula.brsmm, 68, *72*

logLik.brs, *4*, *6*, *15*, 69
logLik.brsmm, *5*, *16*, *37*, 70

model.matrix.brs, *67*, 71
model.matrix.brsmm, *68*, 72

nobs.brs, 73
nobs.brsmm, 74

optim, *18, 19*, *53, 54*

plot.brs, *9*, 75, *88*
plot.brsmm, 76, *90*
predict.brs, *22*, *65*, 78, *79*
predict.brsmm, *39*, *55*, *66*, 79
print.brs, 80, *83*
print.brs_bootstrap *(*brs_bootstrap*)*, 22
print.brsmm, 81, *84*
print.brsmm_re_study, 82
print.summary.brs, *81*, 83
print.summary.brsmm, *81*, 84, *94*

ranef, 85, *86*
ranef.brsmm, *55*, *58*, *82*, *85*, 86, *89*, *92*
residuals.brs, *9*, *22*, *65*, *75, 76*, 87, *89, 90*, *93*
residuals.brsmm, *14*, *66*, *77*, *88*, 89

simulate, *91*, *92*
simulate.brs, 91
simulate.brsmm, *60*
simulate.brsmm *(*simulate.brs*)*, 91
summary.brs, *6*, *22*, *30*, *33*, *81*, *83*, 92
summary.brsmm, *7*, *54*, *55*, *60*, *81*, *84*, 94

vcov.brs, *19*, *35*, *61*, *63*, *64*, *93*, 95
vcov.brsmm, *62*, *65*, 96