Package ‘qbrms’

December 10, 2025

Title Quick Bayesian Regression Models Using 'INLA' with 'brms' Syntax

Version 1.0.1

Date 2025-11-26

Maintainer Tony Myers <admyers@aol.com>

Description Provides a 'brms'-like interface for fitting Bayesian regression models using 'INLA' (Integrated Nested Laplace Approximations) and 'TMB' (Template Model Builder). The package offers faster model fitting while maintaining familiar 'brms' syntax and output formats. Supports fixed and mixed effects models, multiple probability distributions, conditional effects plots, and posterior predictive checks with summary methods compatible with 'brms'. 'TMB' integration provides fast ordinal regression capabilities. Implements methods adapted from 'emmeans' for marginal means estimation and 'bayestestR' for Bayesian inference assessment. Methods are based on Rue et al. (2009) <doi:10.1111/j.1467-9868.2008.00700.x>, Kristensen et al. (2016) <doi:10.18637/jss.v070.i05>, Lenth (2016) <doi:10.18637/jss.v069.i01>, Bürkner (2017) <doi:10.18637/jss.v080.i01>, Makowski et al. (2019) <doi:10.21105/joss.01541>, and Kruschke (2014, ISBN:9780124058880).

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

RoxygenNote 7.3.2

Depends R (>= 4.0.0)

Imports methods, ggplot2, mvtnorm, cowplot, lme4, patchwork, posterior, scales, shiny, miniUI, future, future.apply, loo, TMB, jsonlite

Suggests testthat (>= 3.0.0), INLA, knitr, rmarkdown, DHARMa, MASS, ordinal, rstudioapi, emmeans, bayestestR, gridExtra, data.table, tibble, quantreg, readxl, haven

LinkingTo Rcpp, RcppEigen, TMB

Config/testthat/edition 3

URL https://github.com/Tony-Myers/qbrms

BugReports https://github.com/Tony-Myers/qbrms/issues

Additional_repositories https://inla.r-inla-download.org/R/stable/

VignetteBuilder knitr

NeedsCompilation yes

Author Tony Myers [aut, cre] (ORCID: <https://orcid.org/0000-0003-4516-4829>)

Repository CRAN

Date/Publication 2025-12-10 21:10:13 UTC

Contents

qbrms-package *qbrms: Quick Bayesian Regression Models using INLA*

Description

The qbrms package provides a brms-like interface for fitting Bayesian regression models using INLA (Integrated Nested Laplace Approximations). It offers faster model fitting while maintaining familiar brms syntax and output formats.

Details

The main function is `qbrms` which fits Bayesian models using INLA with brms-like syntax. The package supports:

- Fixed and mixed effects models
- Multiple probability distributions
- Conditional effects plots
- Posterior predictive checks
- Summary methods compatible with brms
- Interactive model building with `model_builder`
- RStudio add-in for point-and-click model specification

Author(s)

Tony Myers

See Also

Useful links:

- `https://github.com/Tony-Myers/qbrms`
- Report bugs at `https://github.com/Tony-Myers/qbrms/issues`

asymmetric_laplace *Asymmetric Laplace for Quantile Regression*

Description

Asymmetric Laplace for Quantile Regression

Usage

```
asymmetric_laplace()
```

Value

An object of class `"family"` representing the Asymmetric Laplace distribution.

bayesfactor *Bayesian Hypothesis Testing (very simple approximations)*

Description

Compute a crude Bayes factor for a point, interval, or comparison hypothesis using approximate posterior draws recovered from a `qbrms_fit`. This is deliberately simple and intended for exploratory use.

Usage

```
bayesfactor(
  object,
  hypothesis,
  prior = NULL,
  null = 0,
  direction = "two-sided",
  rope = NULL,
  nsim = 4000,
  verbose = TRUE
)
```

Arguments

`object` A `qbrms_fit` object.

`hypothesis` Character string, for example "Intercept > 0", "b_x = 0", or "b_x > 0.2".

`prior` Optional prior information (unused here, kept for API compatibility).

`null` Numeric null value for point tests (default 0).

`direction` One of "two-sided", "greater", "less" (kept for API compatibility).

`rope` Optional numeric length-2 vector `c(lower, upper)` to define a ROPE for point tests.

`nsim` Number of posterior draws to simulate from the fitted summary.

`verbose` Logical; print progress information.

Value

An object of class `qbrms_bayesfactor`.

bayes_R2 *Bayesian R-squared for qbrms Models*

Description

Compute Bayesian R-squared values for qbrms regression models following the method of Gelman et al. (2019). This corrected version properly handles mixed-effects models to match brms output exactly.

Usage

```
bayes_R2(
  object,
  summary = TRUE,
  robust = FALSE,
  probs = c(0.025, 0.975),
  ndraws = 1000,
  newdata = NULL,
  verbose = TRUE
)
```

Arguments

object	A `qbrms_fit` object.
summary	Logical; if `TRUE` (default), return summary statistics. If `FALSE`, return the posterior draws.
robust	Logical; if `TRUE`, use robust summary statistics.
probs	Numeric vector of quantiles for summary (default: c(0.025, 0.975)).
ndraws	Number of posterior draws to use (default: 1000).
newdata	Optional data frame for predictions. If `NULL`, uses the original data.
verbose	Logical; print progress information.

Details

This implementation handles mixed-effects models by:

1. Using INLA's fitted values that include random effects when available
2. Correctly sampling random effects from their posterior distributions
3. Properly accounting for the variance decomposition in mixed models

Value

If `summary = TRUE`, a matrix with summary statistics. If `summary = FALSE`, a vector of R-squared values from posterior draws.

Beta *Beta Family Constructor (Capital B)*

Description

Beta distribution family for response variables in (0,1)

Usage

```
Beta(link = "logit", link.phi = "log")

Beta(link = "logit", link.phi = "log")
```

Arguments

`link`	Link function for the mean parameter (default: "logit")
`link.phi`	Link function for precision parameter (default: "log")

Value

A family object of class `"family"`.

A family object for use with qbrms()

Examples

```
## Not run:
# Beta regression for proportions
fit <- qbrms(proportion ~ predictor, data = data, family = Beta())

## End(Not run)
```

beta_binomial *Beta Binomial Family for Overdispersed Binary Data*

Description

Beta Binomial Family for Overdispersed Binary Data

Usage

```
beta_binomial(link = "logit")
```

Arguments

`link`	Link function for probability parameter (default: "logit")

Value

A family object of class "`family`".

`beta_prior` *Specify Beta Prior Distribution*

Description

Specify Beta Prior Distribution

Usage

```
beta_prior(alpha = 1, beta = 1)
```

Arguments

`alpha`	First shape parameter
`beta`	Second shape parameter

Value

A prior distribution object

`beta_variants` *Alternative Beta Parameterizations*

Description

Alternative Beta Parameterizations

Usage

```
beta0(link = "logit")

beta1(link = "logit")

logitbeta(link = "logit")
```

Arguments

`link`	Link function (default: "logit")

Value

A family object of class "`family`".

bf *Create a Bayesian Formula*

Description

Function to set up a model formula for use in qbrms, allowing specification of distributional parameters (e.g., sigma) in addition to the mean structure.

Usage

```
bf(formula, ..., flist = NULL, family = NULL, nl = FALSE)
```

Arguments

formula	Main model formula (for the mean/location parameter).
...	Additional formulas for distributional parameters (e.g., sigma ~ x).
flist	Optional list of formulas (for internal use).
family	Same as in qbrms() (optional here).
nl	Logical; indicating if the model is non-linear (not yet fully supported).

Details

This function mimics the brms::bf() syntax to allow users familiar with brms to define distributional models.

Supported distributional parameters depend on the family:

- gaussian: sigma (residual standard deviation)
- student_t: sigma, nu (degrees of freedom)
- lognormal: sigma (shape parameter)
- beta: phi (precision)
- simplex: phi (precision)

Value

An object of class brmsformula (and qbrmsformula) containing the parsed formulas.

Examples

```
## Not run:
# Standard model
f1 <- bf(y ~ x)

# Distributional model (heteroscedasticity)
# Sigma varies by group
f2 <- bf(y ~ x, sigma ~ group)

## End(Not run)
```

`binomial` *Binomial Family*

Description

Binomial Family

Usage

```
binomial()
```

Value

An object of class `"family"` representing the Binomial distribution.

`c.qbrms_prior_spec` *Combine Multiple Prior Specifications*

Description

Combine Multiple Prior Specifications

Usage

```
## S3 method for class 'qbrms_prior_spec'
c(...)
```

Arguments

`...`	Prior specification objects created by prior()

Value

A combined prior object

`cauchy` *Specify Cauchy Prior Distribution*

Description

Specify Cauchy Prior Distribution

Usage

```
cauchy(location = 0, scale = 1)
```

Arguments

`location`	Location parameter (default 0)
`scale`	Scale parameter (default 1)

Value

A prior distribution object

`check_convergence` *Quick model diagnostics*

Description

Quick model diagnostics

Usage

```
check_convergence(object)
```

Arguments

`object`	A qbrms_fit object.

Value

Invisible TRUE if successful.

circular_normal *Circular Normal Family for Directional Data*

Description

Circular Normal Family for Directional Data

Usage

```
circular_normal(link = "tan_half", link.kappa = "log")

von_mises(link = "tan_half", link.kappa = "log")
```

Arguments

`link`	Link function for mean direction
`link.kappa`	Link function for concentration

Value

A family object of class "`family`".

coef.qbrms_fit *Extract Coefficients from qbrms Models*

Description

Extract Coefficients from qbrms Models

Usage

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

Arguments

`object`	A qbrms_fit object
`...`	Additional arguments (unused)

Value

Named vector of coefficients

coef.qbrms_multinomial_fit

Coefficients for multinomial qbrms fits

Description

Extract a concatenated vector of coefficients from a `qbrms_multinomial_fit`, combining the per-category binary submodels if present.

Usage

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

Arguments

`object`	A `qbrms_multinomial_fit`.
`...`	Unused.

Value

A named numeric vector of coefficients. If coefficient information is not available, a minimal intercept-only vector is returned.

coef.tmb_ordinal_qbrms_fit

Coefficients Method for TMB Ordinal Fits

Description

Coefficients Method for TMB Ordinal Fits

Usage

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

Arguments

`object`	A tmb_ordinal_qbrms_fit object
`...`	Additional arguments

Value

Named vector of coefficients

`compare_models` *Compare qbrms models*

Description

Compares multiple fitted models using information criteria and simple predictive metrics. Preference order for criterion = "auto" is: LOO (from CPO) > WAIC > DIC. When information criteria are unavailable it falls back to predictive metrics (RMSE/MAE).

Usage

```
compare_models(
  ...,
  criterion = c("auto", "loo", "waic", "dic", "all"),
  compare_predictions = TRUE,
  weights = TRUE
)
```

Arguments

`...`	Two or more fitted model objects (qbrms_fit or qbrmO_fit)
`criterion`	One of "auto","loo","waic","dic","all"
`compare_predictions`	Logical; if TRUE, include RMSE/MAE comparison
`weights`	Logical; if TRUE, compute weights when a single criterion is used

Value

An object of class "`qbrms_comparison`".

`compare_significance` *Compare Significance Across Multiple Models*

Description

Compare the probability of practical significance for parameters across multiple qbrms models.

Usage

```
compare_significance(
  ...,
  parameters = NULL,
  threshold = "default",
  model_names = NULL
)
```

Arguments

`...`	qbrms_fit objects to compare
`parameters`	Character vector of parameters to compare
`threshold`	Threshold specification (same as p_significance)
`model_names`	Character vector of model names

Value

Data frame with comparison results

`conditional_effects` *Conditional effects (generic)*

Description

Compute one-dimensional conditional effects / marginal fitted values as a predictor varies while other covariates are held fixed (typically at means / modes). Methods should return an object that `plot()` can visualise.

Usage

```
conditional_effects(object, ...)
```

Arguments

`object`	A model object.
`...`	Passed to methods.

Value

An object of class `"qbrms_conditional_effects"` containing conditional effect estimates. The structure is method-dependent.

conditional_effects.qbrms_fit *Conditional effects for qbrms Gaussian models*

Description

Conditional effects for qbrms Gaussian models

Usage

```
## S3 method for class 'qbrms_fit'
conditional_effects(
  object,
  effects = NULL,
  spaghetti = FALSE,
  ndraws = 200L,
  n_points = 100L,
  at = list(),
  seed = NULL,
  prob = 0.95,
  ...
)
```

Arguments

object	A qbrms fit object (Gaussian).
effects	Character vector: names of predictors to vary. Supports simple two-way interactions "num:fac" or "fac:num" where one is numeric and the other factor.
spaghetti	Logical; if TRUE draw per-draw "spaghetti" lines. If FALSE, draw a mean line with a credible-interval ribbon.
ndraws	Number of joint coefficient draws for uncertainty (default 200).
n_points	Size of the x-grid across the observed range (default 100).
at	Optional named list of covariate values to hold constant.
seed	Optional integer seed for reproducibility.
prob	Interval probability for ribbons (default 0.95).
...	Ignored.

Value

An object of class "qbrms_conditional_effects" containing a list with one element per effect. Each element is a data frame with columns for the predictor values, point estimates (estimate__), and credible interval bounds (lower__, upper__).

conditional_effects.tmb_ordinal_qbrms_fit
Conditional Effects for TMB Ordinal Models

Description

Conditional Effects for TMB Ordinal Models

Usage

```
## S3 method for class 'tmb_ordinal_qbrms_fit'
conditional_effects(
  object,
  effects = NULL,
  prob = 0.95,
  ndraws = 100,
  spaghetti = FALSE,
  n_points = 100,
  plot = TRUE,
  at = list(),
  seed = NULL,
  conditions = NULL,
  categorical = TRUE,
  resolution = NULL,
  ...
)
```

Arguments

`object`	A tmb_ordinal_qbrms_fit object
`effects`	Character vector of effect names (defaults to auto-detected)
`prob`	Confidence level
`ndraws`	Number of draws
`spaghetti`	Logical
`n_points`	Number of points for continuous predictors
`plot`	Logical, whether to return plots
`at`	Named list of conditioning values
`seed`	Random seed
`conditions`	Ordinal-specific conditions (for backwards compatibility)
`categorical`	Whether to show categorical plot (for backwards compatibility)
`resolution`	Grid resolution (for backwards compatibility)
`...`	Additional arguments

Value

List of conditional effects

`conditional_effects_slices`
Discrete-slice conditional effects (brms-style) for qbrms

Description

Build point/interval summaries at a few values of a numeric moderator, plotted against the factor on the x-axis.

Usage

```
conditional_effects_slices(
  object,
  effects,
  slices = NULL,
  nslices = 3L,
  prob = 0.95,
  ndraws = 200L,
  at = list(),
  seed = NULL,
  ...
)
```

Arguments

`object`	A qbrms_fit object.
`effects`	Character vector specifying effects to plot. If NULL, all numeric predictors are used.
`slices`	Named list of variables and values at which to slice the data.
`nslices`	Number of slices to use for each slicing variable.
`prob`	Probability mass to include in uncertainty intervals (default 0.95).
`ndraws`	Number of posterior draws to use for predictions.
`at`	Named list of values at which to fix other predictors.
`seed`	Random seed for reproducibility.
`...`	Additional arguments passed to prediction functions.

Value

An object of class `"qbrms_conditional_effects"` containing a list with one element per effect. Each element is a data frame with columns for the predictor values, estimates, and credible intervals.

convert_family_to_inla *Convert Family Object to INLA-Compatible Specification*

Description

Enhanced family conversion supporting all standard and additional families with automatic routing to specialised implementations when enabled.

Usage

```
convert_family_to_inla(family, quantile = 0.5, allow_ordinal_routing = FALSE)
```

Arguments

`family` A family object, character string, or list specifying the response distribution

`quantile` Numeric value between 0 and 1 for quantile regression

`allow_ordinal_routing`
Logical; if TRUE, enables routing for ordinal families

Value

Character string, list, or routing object specifying family/routing info

create_dummy_data *Create Dummy Data for Testing*

Description

Create dummy data that preserves structure for testing purposes.

Usage

```
create_dummy_data(
  formula,
  data,
  n_dummy = 10,
  family_name = "gaussian",
  verbose = FALSE
)
```

Arguments

formula	Model formula.
data	Original data frame.
n_dummy	Number of dummy observations to create.
family_name	The name of the model family (e.g., "gaussian").
verbose	Logical, whether to print messages.

Value

Data frame with dummy structure.

---

create_prior_object *Create Prior-Only Object for pp_check*

Description

Construct a small qbrms_prior_only object that contains simulated data and prior draws, suitable for passing to pp_check().

Usage

```
create_prior_object(
  formula,
  family = gaussian(),
  prior = NULL,
  n_obs = 100,
  predictor_values = NULL,
  verbose = TRUE
)
```

Arguments

formula	Model formula.
family	Model family (default gaussian()).
prior	Prior specifications (default NULL uses defaults).
n_obs	Number of observations to simulate (default 100).
predictor_values	Named list of fixed predictor values (default NULL).
verbose	Logical; print progress messages (default TRUE).

Value

An object of class qbrms_prior_only.

cumulative *Cumulative Family for Ordinal Regression*

Description

Cumulative Family for Ordinal Regression

Usage

```
cumulative(link = "logit")
```

Arguments

link	Link function (default: "logit").

Value

An object of class "family" representing the Cumulative distribution for ordinal models.

default_priors *Default Priors for qbrms Models*

Description

Default Priors for qbrms Models

Usage

```
default_priors()
```

Value

A default prior list

density_plot — *Density Plot for qbrms Models*

Description

Create density plots of posterior distributions with optional prior and observed-data overlays. Returns a ggplot2 object that can be modified with standard ggplot2 syntax.

Usage

```
density_plot(
  object,
  parameter = NULL,
  show_prior = FALSE,
  show_data = FALSE,
  ndraws = 100,
  prior_ndraws = 100,
  alpha_levels = list(posterior = 0.8, prior = 0.6, data = 1),
  colours = list(posterior = "#1F78B4", prior = "#E31A1C", data = "#000000"),
  seed = NULL,
  verbose = TRUE
)
```

Arguments

object	A qbrms_fit object.
parameter	Parameter name to plot. If NULL, plots the response distribution.
show_prior	Logical; if TRUE, overlay the prior density.
show_data	Logical; if TRUE, overlay the observed data density.
ndraws	Number of posterior draws to use (default 100).
prior_ndraws	Number of prior draws to use (default 100).
alpha_levels	Named list controlling transparency for layers.
colours	Named list of colours for layers.
seed	Optional random seed.
verbose	Logical; print progress messages.

Value

A ggplot2 object.

`diagnose_binomial_mixed` *Diagnose Binomial Mixed Effects Models*

Description

Diagnose potential issues in binomial mixed effects models before fitting

Usage

```
diagnose_binomial_mixed(formula, data, verbose = TRUE)
```

Arguments

`formula`	Model formula with mixed effects
`data`	Data frame containing variables
`verbose`	Logical; print diagnostic information (default: TRUE)

Value

List with diagnostic information

`diagnose_model` *Automated Model Diagnostics and Recommendations*

Description

Comprehensive automated diagnostics for qbrms models with actionable recommendations for model improvement.

Usage

```
diagnose_model(model, checks = "all", verbose = TRUE)
```

Arguments

`model`	A fitted qbrms model object
`checks`	Character vector specifying which checks to perform. Options: "all" (default), "convergence", "fit", "residuals", "posterior", "influential"
`verbose`	Logical; if TRUE, prints detailed diagnostic information (default: TRUE)

Details

This function performs comprehensive model diagnostics including:

- Convergence checks (for MCMC-based inference)
- Goodness-of-fit assessment
- Residual analysis
- Posterior predictive checks
- Influential observation detection
- Prior-posterior overlap assessment

Each check produces a pass/warning/fail status with specific recommendations for addressing any issues detected.

Value

An object of class "qbrms_diagnostics" containing:

- `summary`: Overall assessment (pass/warning/fail)
- `checks`: Detailed results for each diagnostic check
- `recommendations`: Specific suggestions for improvement
- `plots`: List of diagnostic plots

Examples

```
## Not run:
# Fit a model
fit <- qbrms(mpg ~ hp + wt, data = mtcars, family = gaussian())

# Run diagnostics
diag <- diagnose_model(fit)

# View summary
print(diag)

# View specific recommendations
diag$recommendations

# Create diagnostic plots
plot(diag)

## End(Not run)
```

drop_random_effects *Drop Random Effects from Formula*

Description

Remove random effects terms from a model formula.

Usage

```
drop_random_effects(formula)
```

Arguments

formula	A model formula that may contain random effects.

Value

Formula with random effects terms removed.

emmeans *Estimated marginal means (compatibility wrapper)*

Description

This wrapper lets you call emmeans() on a qbrms_fit without attaching the external **emmeans** package. For non-qbrms_fit objects, it forwards to **emmeans** if that package is installed.

Usage

```
emmeans(object, specs, ...)
```

Arguments

object	A model object; if it is a qbrms_fit we dispatch to qbrms_emmeans().
specs	Term(s) for which to compute estimated marginal means. For qbrms_fit, this is passed to qbrms_emmeans() unchanged.
...	Additional arguments forwarded either to qbrms_emmeans() or to emmeans::emmeans() as appropriate.

Value

A data frame for qbrms_fit; otherwise whatever emmeans::emmeans() returns.

exponential *Exponential Distribution (Prior or Family)*

Description

Exponential Distribution (Prior or Family)

Usage

```
exponential(rate_or_link = "log", link = NULL, ...)

prior_exponential(rate_or_link = 1)
```

Arguments

rate_or_link	Rate parameter (numeric) or link function (character).
link	Optional link function (if acting as family).
...	Additional arguments.

Value

A family object or prior object depending on inputs.

export_model *Export Model Specification*

Description

Export model specifications to various formats for sharing, documentation, or reproduction.

Usage

```
export_model(
  model,
  file,
  format = c("R", "markdown", "text", "json"),
  include_data = TRUE,
  include_diagnostics = FALSE
)
```

Arguments

`model` A fitted qbrms model object or qbrms_model_spec object

`file` Character string specifying output file path

`format` Character string specifying export format: "R" (R script), "markdown" (Rmd document), "text" (plain text), or "json" (JSON format)

`include_data` Logical; if TRUE, includes data summary in export (default: TRUE)

`include_diagnostics` Logical; if TRUE and model is fitted, includes diagnostic information (default: FALSE)

Details

This function facilitates model sharing and documentation by exporting:

- Model formula and family specification
- Prior specifications (if any)
- Data summary and structure
- Model fitting code
- Results summary (for fitted models)
- Diagnostic information (if requested)

The exported content can be used to:

- Share analyses with collaborators
- Document modelling decisions
- Create reproducible research reports
- Archive model specifications

Value

Invisibly returns the export content as a character string

Examples

```
## Not run:
# Export model specification
spec <- model_builder(data = mtcars, response = "mpg")
export_model(spec, "my_model_spec.R", format = "R")

# Export fitted model
fit <- qbrms(mpg ~ hp + wt, data = mtcars, family = gaussian())
export_model(fit, "my_model.Rmd", format = "markdown",
             include_diagnostics = TRUE)

# Export as JSON
export_model(spec, "my_model.json", format = "json")

## End(Not run)
```

`family_info` *Get Family Documentation*

Description

Get Family Documentation

Usage

```
family_info(family_name)
```

Arguments

`family_name`	Name of the family

Value

Character string with family information

`family_supports_quantile` *Check if Family Supports Quantile Regression*

Description

Determine whether a given family supports quantile regression.

Usage

```
family_supports_quantile(family_obj)
```

Arguments

`family_obj`	Family object or name.

Value

Logical indicating whether the family supports quantile regression.

fitted.qbrms_fit *Extract fitted values from qbrms models*

Description

Extract fitted values from qbrms models

Usage

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

Arguments

`object`	A qbrms_fit object
`...`	Additional arguments (currently unused)

Value

Numeric vector of fitted values

Examples

```
## Not run:
fit <- qbrms(mpg ~ hp, data = mtcars, family = gaussian())
fitted_values <- fitted(fit)

## End(Not run)
```

fitted.tmb_ordinal_qbrms_fit *Fitted Values Method for TMB Ordinal Fits*

Description

Fitted Values Method for TMB Ordinal Fits

Usage

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

Arguments

`object`	A tmb_ordinal_qbrms_fit object
`...`	Additional arguments

Value

Fitted values

`gamma` *Gamma Distribution (Prior or Family)*

Description

Gamma Distribution (Prior or Family)

Usage

```
gamma(shape_or_link = "log", rate = 1, link = NULL, ...)

gamma_prior(shape_or_link = 2, rate = 1)
```

Arguments

`shape_or_link`	Shape parameter (numeric) or link function (character).
`rate`	Rate parameter.
`link`	Optional link function (if acting as family).
`...`	Additional arguments.

Value

A family object or prior object depending on inputs.

`Gamma_family` *Gamma family (GLM-style)*

Description

Gamma family constructor to avoid conflict with base::gamma.

Usage

```
Gamma(link = "log")
```

Arguments

`link`	Link function (default: "log")

Value

A family object of class `"family"`.

gaussian *Gaussian Family*

Description

Gaussian Family

Usage

```
gaussian()
```

Value

An object of class "`family`" representing the Gaussian distribution.

gen_student_t *Generalized t Family*

Description

Generalized t Family

Usage

```
gen_student_t(link = "identity", link.sigma = "log", link.nu = "log")
```

Arguments

`link`	Link function for location
`link.sigma`	Link function for scale
`link.nu`	Link function for degrees of freedom

Value

A family object of class "`family`".

get_default_prior *Get Default Prior for Parameter Class*

Description

Get Default Prior for Parameter Class

Usage

```
get_default_prior(class)
```

Arguments

`class`	Parameter class

Value

A default prior for that class

gev *Generalized Extreme Value Family*

Description

Generalized Extreme Value Family

Usage

```
gev(link = "identity", link.sigma = "log", link.xi = "identity")

gumbel(link = "identity", link.sigma = "log")
```

Arguments

`link`	Link function for location
`link.sigma`	Link function for scale
`link.xi`	Link function for shape

Value

A family object of class `"family"`.

hdi *Highest Density Interval (HDI)*

Description

Compute highest density intervals for parameters based on simulated posterior draws from a `qbrms_fit`.

Usage

```
hdi(object, parameters = NULL, prob = 0.95, nsim = 4000)
```

Arguments

`object`	A `qbrms_fit`.
`parameters`	Optional character vector; default uses all fixed effects.
`prob`	Probability mass for the interval (default 0.95).
`nsim`	Number of draws to simulate.

Value

A data frame of class `qbrms_hdi`.

hurdle_families *Hurdle Families for Two-Part Models*

Description

Hurdle Families for Two-Part Models

Usage

```
hurdle_poisson(link = "log", link.hu = "logit")

hurdle_negbinomial(link = "log", link.hu = "logit")
```

Arguments

`link`	Link function for count component
`link.hu`	Link function for hurdle component

Value

A family object of class `"family"`.

`iid` *IID Random Effects*

Description

IID Random Effects

Usage

```
iid(scale.model = "log", diagonal = 1e-06)
```

Arguments

`scale.model`	Scaling model for precision
`diagonal`	Diagonal precision matrix structure

Value

A family object of class `"family"`.

`import_model` *Import Model Specification from JSON*

Description

Import a previously exported model specification from JSON format.

Usage

```
import_model(file)
```

Arguments

`file`	Character string specifying JSON file path

Value

A list containing the model specification components

Examples

```
## Not run:
# Import model
spec <- import_model("my_model.json")

# Recreate model
fit <- qbrms(
  formula = as.formula(spec$model$formula),
  data = my_data,
  family = get(spec$model$family)()
)

## End(Not run)
```

---

kfold_cv *K-fold cross-validation for qbrms models (ordinal and standard families)*

Description

Performs K-fold cross-validation either from a fitted model or from `formula` + `data`. For **ordinal (cumulative/ordinal)** families, you can choose the re-fit engine used inside CV: TMB (`qbrmO`) or a robust fallback using `MASS::polr` that avoids TMB compilation in each fold. Your original fitted model is unchanged.

Usage

```
kfold_cv(
  object,
  data = NULL,
  family = gaussian(),
  K = 10,
  folds = NULL,
  seed = NULL,
  stratify = TRUE,
  parallel = FALSE,
  workers = NULL,
  keep_fits = FALSE,
  engine = c("auto", "tmb", "polr"),
  verbose = TRUE,
  ...
)
```

Arguments

`object` Either a fitted qbrms/qbrmO object **or** a formula.

`data` Required only if `object` is a formula. Ignored if `object` is a fit.

`family` Optional family override (used if `object` is a formula; fits use their own).

`K` Number of folds (default 10).

`folds` Optional integer vector of length `nrow(data)` giving fold IDs.

`seed` Optional seed for stratified folds.

`stratify` Logical; stratify on response if factor/ordered (default TRUE).

`parallel` Logical; use `future.apply` if available (default FALSE).

`workers` Optional workers when parallel and no plan is set.

`keep_fits` Logical; keep per-fold fits (default FALSE).

`engine` Ordinal CV engine: `"auto"` (default), `"tmb"`, or `"polr"`. Only used for **ordinal** families during CV refits. `"auto"` uses `getOption("qbrms.kfold.ordinal_engine", "polr")`.

`verbose` Logical; brief progress (default TRUE).

`...` Passed to `qbrms()` when refitting folds (non-ordinal or `engine="tmb"`).

Value

An object of class `qbrms_kfold` with ELPD, pointwise elpd, SE, etc.

`laplace` *Laplace (Double Exponential) Family*

Description

Laplace (Double Exponential) Family

Usage

```
laplace(link = "identity", link.sigma = "log")

double_exponential(link = "identity", link.sigma = "log")
```

Arguments

`link` Link function for location

`link.sigma` Link function for scale

Value

A family object of class `"family"`.

`list_extended_families` *List Available Extended Families*

Description

List Available Extended Families

Usage

```
list_extended_families()
```

Value

Data frame with family names, categories, and brief descriptions

`lognormal` *Lognormal Family Constructor*

Description

Lognormal distribution family for positive continuous responses

Usage

```
lognormal(meanlog_or_link = "identity", sdlog = 1, link = NULL, ...)

lognormal(meanlog_or_link = "identity", sdlog = 1, link = NULL, ...)

lognormal_prior(meanlog_or_link = 0, sdlog = 1)
```

Arguments

`meanlog_or_link`	Mean on log scale (numeric) or link function (character).
`sdlog`	SD on log scale (numeric).
`link`	Optional link function (if acting as family).
`...`	Additional arguments.

Value

A family object for use with qbrms()

A family object or prior object depending on inputs.

Examples

```
## Not run:
# Lognormal regression
fit <- qbrms(response ~ predictor, data = data, family = lognormal())

## End(Not run)
```

loo_compare *Compare models by LOO (default) or WAIC*

Description

Compare multiple fitted models and rank them by out-of-sample fit. If you pass qbrms/qbrmO fit objects, this uses the package's `loo()` / `waic()` wrappers under the hood. If you pass actual `loo` objects (from the **loo** package), it will delegate to `loo::loo_compare()` automatically.

Usage

```
loo_compare(..., criterion = c("loo", "waic"), sort = TRUE)
```

Arguments

`...`	One or more fitted models (qbrms/qbrmO), or `loo` objects; you can also pass a single named list of models.
`criterion`	Character, "loo" (default) or "waic".
`sort`	Logical; if TRUE (default) the best model is first.

Value

A data.frame with model names, estimate on the ELPD scale (higher is better), standard error (if available), differences vs best, and ranks.

model_builder *Interactive Model Builder for qbrms (console)*

Description

An interactive assistant that guides users through model specification by asking questions about their data, suggesting appropriate families, helping with prior selection, and building qbrms model code.

Usage

```
model_builder(data = NULL, response = NULL, predictors = NULL, quiet = FALSE)
```

Arguments

`data`	A data frame containing the variables to be modelled (optional). If not provided, the user will be prompted to specify it.
`response`	Character string specifying the response variable name (optional).
`predictors`	Character vector of predictor variable names (optional).
`quiet`	Logical; if TRUE, suppresses welcome messages (default: FALSE).

Value

An object with class `"qbrms_model_spec"` containing:

- `formula`: The model formula
- `family`: A list with `name` and the constructed family `object`
- `prior`: Prior specifications (if provided)
- `data`: The data frame
- `data_name`: The symbol used for data in the emitted code
- `model_code`: Character string with executable qbrms code
- `response_info`: Summary information about the response

Examples

```
## Not run:
spec <- model_builder()
fit <- eval(parse(text = spec$model_code))

## End(Not run)
```

`model_lab_addin` *qbrms Model Lab (RStudio Add-in)*

Description

Compare plausible families, run prior/posterior checks, plot conditional effects, compute diagnostics, and emit reproducible code. Does not load 'brms'.

Usage

```
model_lab_addin()
```

Value

This function is called for its side effects (launching a Shiny gadget in RStudio). It returns `NULL` invisibly.

`model_workflow_addin` *Launch Guided Bayesian Workflow (RStudio Add-in)*

Description

A comprehensive, step-by-step assistant for Bayesian model building with qbrms.

Usage

```
model_workflow_addin()
```

Value

No return value. This function launches an interactive Shiny gadget for model building and code generation.

`multinomial` *Multinomial Family*

Description

Multinomial Family

Usage

```
multinomial()
```

Value

An object of class `"family"` representing the Multinomial distribution.

`negbinomial` *Negative Binomial Family (Alias)*

Description

Negative Binomial Family (Alias)

Usage

```
negbinomial()
```

Value

An object of class `"family"` representing the Negative Binomial distribution.

`neg_binomial` *Negative Binomial Family*

Description

Negative Binomial Family

Usage

```
neg_binomial()
```

Value

An object of class `"family"` representing the Negative Binomial distribution.

`normal` *Specify Normal Prior Distribution*

Description

Specify Normal Prior Distribution

Usage

```
normal(mean = 0, sd = 1)
```

Arguments

`mean`	Mean of the normal distribution (default 0)
`sd`	Standard deviation of the normal distribution (default 1)

Value

A prior distribution object

plot.qbrms_conditional_effects

Plot conditional effects for qbrms models

Description

Plot method for objects returned by `conditional_effects` and related helpers. For a single effect, this produces either a spaghetti plot of draws or a ribbon / slice plot of summary statistics. For multiple effects it can combine the plots using **patchwork** if available.

Usage

```
## S3 method for class 'qbrms_conditional_effects'
plot(x, ...)
```

Arguments

`x`	An object of class `"qbrms_conditional_effects"`, typically the result of `conditional_effects` or `conditional_effects_slices`.
`...`	Currently ignored. Included for future extensions and method compatibility.

Value

For a single effect, a `ggplot2` object. For multiple effects, either

- a **patchwork** object combining the individual plots (if the **patchwork** package is installed), or
- a named list of `ggplot2` objects otherwise.

plot.qbrms_diagnostics

Plot Method for Diagnostics

Description

Plot Method for Diagnostics

Usage

```
## S3 method for class 'qbrms_diagnostics'
plot(x, which = "all", ...)
```

Arguments

x	A qbrms_diagnostics object
which	Character vector specifying which plots to show
...	Additional arguments (unused)

Value

Invisibly returns the input object x.

plot.qbrms_p_significance

Plot Method for Enhanced p_significance

Description

Create a visual plot of probability of practical significance results.

Create a visual plot of probability of practical significance results.

Usage

```
## S3 method for class 'qbrms_p_significance'
plot(x, ...)
```

```
## S3 method for class 'qbrms_p_significance'
plot(x, ...)
```

Arguments

x	A qbrms_p_significance object from p_significance().
...	Additional arguments passed to ggplot2 functions.

Value

A ggplot2 object.

A ggplot2 object.

`plot_parameters` *Plot Parameters with Prior/Posterior Comparison*

Description

Create density plots for multiple model parameters, optionally comparing posterior estimates with their priors. Returns a ggplot2 object with faceted parameter plots.

Create density plots for multiple model parameters, optionally comparing posterior estimates with their priors. Returns a ggplot2 object with faceted parameter plots.

Usage

```
plot_parameters(
  object,
  pars = NULL,
  show_prior = FALSE,
  ndraws = 200,
  prior_ndraws = 200,
  ncol = 2,
  alpha_levels = c(0.8, 0.5),
  colours = c("#1F78B4", "#E31A1C"),
  verbose = TRUE,
  ...
)
```

```
plot_parameters(
  object,
  pars = NULL,
  show_prior = FALSE,
  ndraws = 200,
  prior_ndraws = 200,
  ncol = 2,
  alpha_levels = c(0.8, 0.5),
  colours = c("#1F78B4", "#E31A1C"),
  verbose = TRUE,
  ...
)
```

Arguments

`object` A `qbrms_fit` object.

`pars` Optional character vector of parameter names to plot. If `NULL`, plots all fixed-effect parameters.

`show_prior` Logical; if `TRUE`, overlay prior distributions.

`ndraws` Number of posterior draws to use for plotting.

`prior_ndraws` Number of prior draws to use if `show_prior` = `TRUE`.

`ncol` Number of columns for faceting (default 2).

`alpha_levels` Numeric vector of length 2 giving alpha levels for c(posterior, prior). Default c(0.8, 0.5).

`colours` Character vector of length 2 giving colours for c(posterior, prior). Default c("#1F78B4", "#E31A1C").

`verbose` Logical; print progress information.

`...` Additional arguments (currently unused).

Value

A ggplot2 object with faceted parameter density plots.

A ggplot2 object with faceted parameter density plots.

Examples

```
## Not run:
fit <- qbrms(y ~ x1 + x2, data = my_data, sample_prior = "yes")

# Plot all parameters
plot_parameters(fit)

# Plot specific parameters with priors
plot_parameters(fit, pars = c("x1", "x2"), show_prior = TRUE)

# Customize appearance
plot_parameters(fit, show_prior = TRUE) +
  theme_bw() +
  labs(title = "My Parameter Estimates")

## End(Not run)
```

```
## Not run:
fit <- qbrms(y ~ x1 + x2, data = my_data, sample_prior = "yes")

# Plot all parameters
plot_parameters(fit)

# Plot specific parameters with priors
plot_parameters(fit, pars = c("x1", "x2"), show_prior = TRUE)

# Customize appearance
plot_parameters(fit, show_prior = TRUE) +
  theme_bw() +
  labs(title = "My Parameter Estimates")

## End(Not run)
```

`poisson` *Poisson Family*

Description

Poisson Family

Usage

```
poisson()
```

Value

An object of class `"family"` representing the Poisson distribution.

`pp_check` *Posterior and prior predictive checks*

Description

Create posterior or prior predictive diagnostic plots for fitted qbrms models.

Usage

```
pp_check(object, ...)

## S3 method for class 'qbrms_fit'
pp_check(
  object,
  type = "dens_overlay",
  ndraws = 5000,
  seed = NULL,
  show_observed = FALSE,
  ...
)

## S3 method for class 'qbrms_prior'
pp_check(
  object,
  type = "dens_overlay",
  ndraws = 5000,
  seed = NULL,
  show_observed = FALSE,
  ...
)
```

Arguments

`object`	A model object.
`...`	Additional arguments passed to methods.
`type`	Character string indicating the check type: one of `"dens_overlay"`, `"hist"`, `"scatter"`, `"scatter_avg"`.
`ndraws`	Integer number of draws to use.
`seed`	Optional RNG seed.
`show_observed`	Logical; show observed data where applicable.

pp_check.tmb_ordinal_qbrms_fit

Posterior predictive checks for TMB ordinal models

Description

Posterior predictive checks for TMB ordinal models

Usage

```
## S3 method for class 'tmb_ordinal_qbrms_fit'
pp_check(
  object,
  type = "bars",
  ndraws = 100,
  seed = NULL,
  newdata = NULL,
  prob = 0.9,
  ...
)
```

Arguments

`object`	A fitted TMB ordinal qbrms model object
`type`	Character; type of posterior predictive check
`ndraws`	Integer; number of posterior draws to use
`seed`	Random seed for reproducibility.
`newdata`	Optional data frame for predictions. If NULL, uses original data.
`prob`	Probability mass for credible intervals (default 0.95).
`...`	Additional arguments passed to methods.

Value

A ggplot object showing the posterior predictive check

print.qbrmb_fit *Print a qbrmb model fit*

Description

Nicely formatted one-line summary plus key diagnostics for a `qbrmb_fit`.

Prints a summary of a regularised binomial qbrms model.

Usage

```
## S3 method for class 'qbrmb_fit'
print(x, digits = 2, ...)

## S3 method for class 'qbrmb_fit'
print(x, digits = 2, ...)
```

Arguments

`x`	A `qbrmb_fit` object.
`digits`	Number of decimal places for output (default: 2).
`...`	Additional arguments (currently unused).

Value

Invisibly returns the input object `x`.

print.qbrmsformula *Print method for qbrms formulas*

Description

Print method for qbrms formulas

Usage

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

Arguments

`x`	A qbrmsformula object
`...`	Unused

Value

Invisibly returns the input object `x`.

print.qbrms_diagnostics *Print Method for Diagnostics*

Description

Print Method for Diagnostics

Usage

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

Arguments

`x`	A qbrms_diagnostics object
`...`	Additional arguments (unused)

Value

Invisibly returns the input object `x`.

print.qbrms_fit *Print Method for qbrms_fit Objects*

Description

Prints a summary of a fitted qbrms model object.

Usage

```
## S3 method for class 'qbrms_fit'
print(x, digits = 2, ...)
```

Arguments

`x`	A `qbrms_fit` object.
`digits`	Number of decimal places for output (default: 2).
`...`	Additional arguments (currently unused).

Value

Invisibly returns the input object `x`.

Examples

```
if (requireNamespace("INLA", quietly = TRUE)) {
  fit <- qbrms(mpg ~ hp, data = mtcars, family = gaussian(), verbose = FALSE)
  print(fit)
}
```

print.qbrms_kfold *Print Method for qbrms_kfold Objects*

Description

Print Method for qbrms_kfold Objects

Usage

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

Arguments

x	A qbrms_kfold object
...	Additional arguments (unused)

Value

Invisibly returns the input object x.

print.qbrms_loo_compare *Print Method for qbrms_loo_compare Objects*

Description

Print Method for qbrms_loo_compare Objects

Usage

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

Arguments

x	A qbrms_loo_compare object
...	Additional arguments (unused)

Value

Invisibly returns the input object `x`.

print.qbrms_model_spec

Print Method for qbrms_model_spec

Description

Print Method for qbrms_model_spec

Usage

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

Arguments

`x`	A qbrms_model_spec object
`...`	Additional arguments (unused)

Value

Invisibly returns the input object `x`.

print.qbrms_multinomial_fit

Print method for multinomial qbrms fits

Description

Shorthand print method that delegates to `summary.qbrms_multinomial_fit()`.

Usage

```
## S3 method for class 'qbrms_multinomial_fit'
print(x, digits = 2, ...)
```

Arguments

`x`	A `qbrms_multinomial_fit`.
`digits`	Integer; number of decimal places to display.
`...`	Unused.

Value

The input `x`, returned invisibly.

print.qbrms_prior_build

Print method for qbrms_prior_build objects

Description

Nicely formats the result of `prior_build_from_beliefs()`, showing the elicited beliefs, implied prior distributions, and (optionally) the corresponding prior code.

Usage

```
## S3 method for class 'qbrms_prior_build'
print(
  x,
  digits = 3,
  show_data = FALSE,
  show_code = TRUE,
  code_object_name = "priors",
  max_terms = 12,
  ...
)
```

Arguments

`x` An object of class `"qbrms_prior_build"` as returned by `prior_build_from_beliefs`.

`digits` Integer scalar giving the number of decimal places to display for numeric summaries (default: `3`).

`show_data` Logical; if `TRUE`, print a compact summary of the elicitation data used to construct the priors.

`show_code` Logical; if `TRUE`, print the corresponding prior code that can be copied into a modelling script.

`code_object_name` Character string giving the name that will be used for the prior object in the displayed code (default: `"priors"`).

`max_terms` Integer scalar giving the maximum number of individual terms to display before truncating the printed output (default: `12`).

`...` Currently ignored. Included for method compatibility.

Value

Invisibly returns the input object `x`.

print.qbrms_prior_code

Print method for qbrms_prior_code objects

Description

Print method for qbrms_prior_code objects

Usage

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

Arguments

x	A qbrms_prior_code object
...	Additional arguments passed to cat

Value

Invisibly returns the input object x.

print.qbrms_prior_diagnostics

Print method for qbrms_prior_diagnostics objects

Description

Print method for qbrms_prior_diagnostics objects

Usage

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

Arguments

x	A qbrms_prior_diagnostics object
digits	Number of decimal places to display (default 3) # <– ADD THIS LINE
...	Additional arguments (unused)

Value

The input object, returned invisibly

print.qbrms_prior_dist

Print Prior Distribution Objects

Description

Print Prior Distribution Objects

Usage

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

Arguments

`x`	A qbrms_prior_dist object
`...`	Unused

Value

Invisibly returns `x`.

print.qbrms_prior_list

Print Prior List Objects

Description

Print Prior List Objects

Usage

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

Arguments

`x`	A qbrms_prior_list object
`...`	Unused

Value

Invisibly returns `x`.

print.qbrms_prior_spec *Print Prior Specification Objects*

Description

Print Prior Specification Objects

Usage

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

Arguments

x	A qbrms_prior_spec object
...	Unused

Value

Invisibly returns x.

print.qbrms_p_significance *Print Method for Enhanced p_significance*

Description

Print results from probability of practical significance analysis.

Usage

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

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

Arguments

x	A qbrms_p_significance object from p_significance().
digits	Number of decimal places to display (default 3).
...	Additional arguments passed to print.data.frame().

Value

Invisibly returns the input object.

Invisibly returns the input object.

`print.summary.qbrms_fit` *Print Method for summary.qbrms_fit Objects*

Description

Print Method for summary.qbrms_fit Objects

Usage

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

Arguments

`x`	A `summary.qbrms_fit` object.
`...`	Additional arguments (currently unused).

Value

Invisibly returns the input object `x`.

`print.tmb_ordinal_qbrms_fit` *Print Method for TMB Ordinal Fits*

Description

Print Method for TMB Ordinal Fits

Usage

```
## S3 method for class 'tmb_ordinal_qbrms_fit'
print(x, digits = 2, ...)
```

Arguments

`x`	A tmb_ordinal_qbrms_fit object
`digits`	Number of decimal places for output
`...`	Additional arguments

Value

Invisibly returns the object

`prior` *Specify Prior for Model Parameters*

Description

Specify Prior for Model Parameters

Usage

```
prior(prior, class = "b", coef = NULL, group = NULL)
```

Arguments

`prior`	A prior distribution object.
`class`	Parameter class ("Intercept", "b", "sd", etc.)
`coef`	Specific coefficient name (optional)
`group`	Specific group name for random effects (optional)

Value

A prior specification object

`priors` *Prior Distribution Specifications*

Description

Functions to specify prior distributions for qbrms models `prior_build_from_beliefs`

Prior Build from Beliefs

Description

Build priors from elicited beliefs (GLM-aware)

Usage

```
prior_build_from_beliefs(
  formula,
  data,
  family = gaussian(),
  beliefs = list(),
  outcome_location = NULL,
  outcome_interval = NULL,
  outcome_level = 0.95,
  outcome_sd = NULL,
  standardise = TRUE,
  plausible_range = NULL,
  target_coverage = 0.8,
  tune = FALSE,
  seed = NULL
)
```

Arguments

`formula`	Model formula
`data`	Data frame
`family`	Model family
`beliefs`	List of beliefs about parameters
`outcome_location`	Expected outcome location
`outcome_interval`	Expected outcome interval
`outcome_level`	Confidence level for outcome interval
`outcome_sd`	Outcome standard deviation
`standardise`	Whether to standardise predictors
`plausible_range`	Plausible range for outcomes
`target_coverage`	Target coverage probability
`tune`	Whether to tune priors
`seed`	Random seed

Value

An object of class "qbrms_prior_build" containing:

- priors: A list of prior specifications.
- prior_code: Character string of R code to reproduce the priors.
- formula: The (possibly modified) model formula.
- data: The (possibly standardised) data.
- diagnostics: Prior predictive diagnostic information.

---

prior_code *Format priors as qbrms prior() code*

Description

Format priors as qbrms prior() code

Usage

```
prior_code(build, object_name = "priors", digits = 3, include_comments = TRUE)
```

Arguments

build	A ’qbrms_prior_build’ returned by prior_build_from_beliefs()
object_name	Name of the object on the left-hand side (default "priors")
digits	Number of decimal places to print
include_comments	Logical; if TRUE, prepend a short comment header

Value

A single character string containing formatted R code

prior_pp_diagnostics *Prior predictive diagnostics and sensibility report*

Description

Summarise prior predictive draws to check basic support, scale and shape, and (optionally) how simple statistics of the observed data compare with the prior-predictive distribution. Returns an object with a concise verdict.

Usage

```
prior_pp_diagnostics(
  object,
  level = 0.95,
  support = NULL,
  lower = NULL,
  upper = NULL,
  trials = NULL,
  plausible_lower = NULL,
  plausible_upper = NULL,
  include_observed = TRUE,
  seed = NULL
)
```

Arguments

`object` A qbrms prior object: qbrms_prior_fit, qbrms_prior_only, or a qbrms_fit that contains `prior_samples`.

`level` Credible level for central intervals (default 0.95). Reserved.

`support` Optional override of the implied support: one of "real", "positive", "proportion", or "bounded". If `NULL`, an attempt is made to infer from the family.

`lower, upper` Optional numeric bounds used when `support = "bounded"`. If `support = "proportion"`, the default is `c(0, 1)`.

`trials` Optional integer vector for binomial data (bounds helper).

`plausible_lower, plausible_upper`
Optional numeric bounds defining a user-declared “plausible range” for the outcome on the response scale. When both are supplied, the function reports the fraction of prior-predictive mass that falls in `[plausible_lower, plausible_upper]` and incorporates this into the verdict.

`include_observed`
Logical; if `TRUE` and the object contains data, the report compares simple statistics of `y` to their prior-predictive reference distributions.

`seed` Optional seed for reproducibility.

Value

An object of class `qbrms_prior_diagnostics`.

`prior_pp_summary` *A convenience wrapper mirroring pp_check's show_observed flag*

Description

A convenience wrapper mirroring pp_check's show_observed flag

Usage

```
prior_pp_summary(
  object,
  show_observed = FALSE,
  plausible_lower = NULL,
  plausible_upper = NULL,
  ...
)
```

Arguments

`object`	A qbrms prior object.
`show_observed`	Logical; compare observed summaries when available.
`plausible_lower, plausible_upper`	Optional plausible range bounds to score coverage.
`...`	Passed to `prior_pp_diagnostics()`.

Value

The diagnostics object, invisibly, after printing a summary.

`prior_predictive_check` *Create Prior Predictive Distribution Plot*

Description

Generate predictions from the prior distribution to assess whether priors are reasonable before seeing the data.

Usage

```
prior_predictive_check(formula, data, family, prior, n_samples = 1000)
```

Arguments

formula	Model formula
data	Data frame (used for structure, not values)
family	Model family
prior	Prior specification
n_samples	Number of prior predictive samples (default: 1000)

Value

A ggplot object showing the prior predictive distribution

Examples

```
## Not run:
prior_predictive_check(
  mpg ~ hp + wt,
  data = mtcars,
  family = gaussian(),
  prior = prior(normal(0, 10), class = "b")
)

## End(Not run)
```

---

prior_to_posterior_workflow

Complete Prior-to-Posterior Workflow

Description

Fit a model with priors sampled, then produce a comparison density plot that overlays posterior, prior, and observed distributions where available.

Usage

```
prior_to_posterior_workflow(
  formula,
  data,
  family = gaussian(),
  prior = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

`formula`	Model formula.
`data`	Data frame.
`family`	Model family (default `gaussian()`).
`prior`	Prior specification (default `NULL`).
`verbose`	Logical; print progress messages.
`...`	Additional arguments forwarded to `qbrms()`.

Value

A list of class `qbrms_workflow_result` with elements `fit` and `plot`.

`p_direction` *Probability of Direction (pd)*

Description

Estimate the probability that a parameter is strictly positive (or strictly negative) under the posterior, based on simulated draws from a `qbrms_fit`.

Usage

```
p_direction(object, parameters = NULL, nsim = 4000, null = 0)
```

Arguments

`object`	A `qbrms_fit` object.
`parameters`	Optional character vector of parameter names. If `NULL`, all fixed-effect coefficients are used.
`nsim`	Number of draws to simulate from the fitted summary.
`null`	Numeric value defining the reference for direction (default 0).

Value

A data frame of class `qbrms_p_direction`.

p_significance *Probability of Practical Significance (Enhanced bayestestR-style)*

Description

Compute the probability that each parameter is above a threshold in the median's direction, similar to bayestestR::p_significance(). This represents the proportion of the posterior distribution that indicates a "significant" effect in the median's direction.

Usage

```
p_significance(
  object,
  parameters = NULL,
  threshold = "default",
  nsim = 1000,
  verbose = TRUE
)
```

Arguments

`object`	A `qbrms_fit` object.
`parameters`	Optional character vector of parameter names; if `NULL`, all fixed-effect coefficients are used.
`threshold`	The threshold value that separates significant from negligible effect: • `"default"`: Uses 0.1 as threshold range around zero • A single numeric value (e.g., 0.1): Creates symmetric range around zero (-0.1, 0.1) • A numeric vector of length two (e.g., c(-0.2, 0.1)): Asymmetric threshold • A list of numeric vectors: Each vector corresponds to a parameter • A named list: Names correspond to parameter names
`nsim`	Number of draws to simulate for the approximation.
`verbose`	Logical; print progress information.

Value

A data frame of class `qbrms_p_significance` with columns `Parameter`, `ps`, `Median`, `CI_low`, `CI_high`, `Threshold_low`, `Threshold_high`, and `Interpretation`.

qbrm *Alias for* `qbrms()`

Description

`qbrm()` is a shorter alias for `qbrms()` with identical functionality.

Enhanced interface to qbrms with all required parameters and built-in diagnostics.

Usage

```
qbrm(
  formula,
  data,
  family = gaussian(),
  prior = NULL,
  sample_prior = "no",
  verbose = TRUE,
  ...
)

qbrm(
  formula,
  data,
  family = gaussian(),
  prior = NULL,
  sample_prior = "no",
  verbose = TRUE,
  ...
)
```

Arguments

`formula`	Model formula in lme4/brms style.
`data`	Data frame containing the variables in the model.
`family`	Model family (default: gaussian()).
`prior`	Prior specifications (default: NULL).
`sample_prior`	Whether to sample from priors ("no", "yes", "only"). Default: "no".
`verbose`	Logical; print diagnostic information (default: TRUE).
`...`	Additional arguments passed to qbrms().

Value

An object of class `"qbrms_fit"`, same as `qbrms()`.

A qbrms_fit object with model results.

See Also

qbrms

---

qbrmb *Enhanced binomial mixed-effects modelling*

Description

Fits a regularised binomial (or Bernoulli) mixed-effects model using INLA, with enhanced diagnostics, stability checks and strategy selection.

Usage

```
qbrmb(
  formula,
  data,
  family = "binomial",
  strategy = "auto",
  regularisation_strength = 0.1,
  use_data_augmentation = TRUE,
  min_group_size = 5,
  verbose = FALSE,
  diagnostics = FALSE,
  silent = FALSE,
  ...
)
```

Arguments

`formula` Model formula with random effects in lme4-style syntax.

`data` Data frame containing the variables in the model.

`family` Model family (currently `"binomial"` or `"bernoulli"`; default `"binomial"`).

`strategy` Fitting strategy: `"auto"`, `"enhanced"`, `"aggressive"`, or `"minimal"`.

`regularisation_strength` Regularisation strength in the interval $[0, 1]$ (default `0.1`).

`use_data_augmentation` Logical; if `TRUE`, add pseudo-observations for additional numerical stability.

`min_group_size` Minimum group size before triggering diagnostic warnings.

`verbose` Logical; if `TRUE`, show detailed progress and diagnostics while fitting.

`diagnostics` Logical; if `TRUE`, compute and store extended diagnostics in the returned object.

`silent` Logical; if `TRUE`, suppress printed output except errors.

`...` Additional arguments passed to `INLA::inla()`.

Value

An object of class c("qbrmb_fit", "qbrms_fit", "list") containing the fitted model, diagnostics and metadata.

qbrmb_aggressive *Aggressively regularised binomial mixed-effects model*

Description

Convenience wrapper around qbrmb using the "aggressive" strategy with a higher default regularisation strength.

Usage

```
qbrmb_aggressive(formula, data, verbose = FALSE, ...)
```

Arguments

formula	Model formula with random effects (lme4-style).
data	Data frame containing the variables in the model.
verbose	Logical; if TRUE, show detailed diagnostics.
...	Additional arguments passed to qbrmb.

Value

An object of class c("qbrmb_fit", "qbrms_fit", "list").

Examples

```
if (requireNamespace("INLA", quietly = TRUE)) {
  set.seed(123)
  data <- data.frame(
    y     = rbinom(100, 1, 0.2),
    x     = rnorm(100),
    group = factor(rep(1:10, each = 10))
  )
  # qbrmb_aggressive requires a mixed model with random intercepts
  fit <- qbrmb_aggressive(y ~ x + (1 | group), data = data, verbose = FALSE)
}
```

qbrmb_regularised *Regularised binomial mixed-effects (enhanced strategy)*

Description

Convenience wrapper around qbrmb using the "enhanced" regularisation strategy and a stronger default regularisation strength.

Usage

```
qbrmb_regularised(formula, data, verbose = FALSE, ...)
```

Arguments

formula	Model formula with random effects (lme4-style).
data	Data frame containing the variables in the model.
verbose	Logical; if TRUE, show detailed diagnostics.
...	Additional arguments passed to qbrmb.

Value

An object of class c("qbrmb_fit", "qbrms_fit", "list").

qbrmO *Quick Bayesian Ordinal Regression Models with Adaptive Centering*

Description

Fits ordinal regression models using Template Model Builder (TMB) with Laplace approximation and adaptive threshold centering to match brms output.

Usage

```
qbrmO(
  formula,
  data,
  family = cumulative(),
  prior = NULL,
  verbose = FALSE,
  threshold_method = "quantile",
  control = list(),
  ...
)
```

Arguments

`formula` Model formula with ordinal response on the left-hand side.

`data` Data frame containing the variables in the model.

`family` Ordinal family specification. Currently supports cumulative().

`prior` Prior specifications using qbrms prior syntax.

`verbose` Logical; print progress messages during fitting.

`threshold_method` Method for threshold centering ("quantile" or "cumulative_mean").

`control` List of control parameters for TMB optimization.

`...` Additional arguments passed to TMB functions.

Value

An object of class c("tmb_ordinal_qbrms_fit", "qbrms_fit")

qbrms *Quick Bayesian Regression Models with Automatic Routing*

Description

Enhanced qbrms interface with automatic routing to specialised implementations. Supports ordinal regression via TMB, quantile regression, and all standard INLA families.

Usage

```
qbrms(
  formula,
  data,
  family = gaussian(),
  prior = NULL,
  sample_prior = "no",
  quantile = 0.5,
  control.compute = list(dic = TRUE, waic = TRUE, cpo = TRUE),
  verbose = getOption("qbrms.verbose", FALSE),
  ...
)
```

Arguments

`formula` Model formula in lme4/brms style

`data` Data frame containing the variables in the model

`family` Model family (default: gaussian()). Ordinal families automatically route to qbrmO()

`prior` Prior specifications (default: NULL) `sample_prior` Whether to sample from priors ("no", "yes", "only"). Default: "no"

`quantile` For asymmetric_laplace family, which quantile to estimate (default: 0.5)

`control.compute` INLA control settings for model information criteria

`verbose` Logical; print diagnostic information (default: getOption("qbrms.verbose", FALSE))

`...` Additional arguments passed to fitting functions

Value

An object of class `"qbrms_fit"` (or `"qbrmO_fit"` for ordinal models). The object is a list containing:

- `fit`: The internal model fit (INLA or TMB object).
- `model_type`: String indicating the type of model fitted.
- `data`: The data used for fitting.
- `original_formula`: The formula used.
- `prior_samples`: Prior predictive samples (if requested).

See Also

`qbrmO` for direct ordinal model fitting

`qbrms-model-criteria` *Model comparison criteria for qbrms models*

Description

Compute approximate DIC, LOO and WAIC for qbrms model fits.

Usage

```
waic(object, ...)

loo(object, ...)

dic(object, ...)

## S3 method for class 'qbrms_fit'
waic(object, ...)

## S3 method for class 'qbrms_fit'
loo(object, ...)

## S3 method for class 'qbrms_fit'
dic(object, ...)
```

Arguments

`object`	A `qbrms_fit` object.
`...`	Additional arguments passed to internal methods or underlying tools.

Details

These functions provide generic interfaces (`dic()`, `loo()`, `waic()`) and S3 methods for `qbrms_fit` objects that extract the corresponding criteria from the underlying INLA fit where available.

Value

For `dic()`, `loo()` and `waic()` methods on `qbrms_fit` objects, a list containing the corresponding criterion (for example, `list(dic = ...)`, `list(looic = ..., elpd_loo = ...)`, or `list(waic = ...)`). If the criterion cannot be computed, `NA_real_` is returned.

---

qbrms_bayesian_analysis

Bayesian Analysis Functions (qbrms)

Description

Posterior analysis tools for qbrms models: Bayes factors, probability of direction, ROPE, HDI, and estimated marginal means.

---

qbrms_binomial_regularised

Fixed Regularised Binomial Mixed Effects Fitting

Description

Fits binomial mixed effects models with regularisation, with all parameters handled correctly.

Usage

```
qbrms_binomial_regularised(
  formula,
  data,
  regularise = TRUE,
  sample_prior = "no",
  verbose = TRUE,
  ...
)
```

Arguments

`formula` Model formula with mixed effects structure.

`data` Data frame containing the variables.

`regularise` Logical; if TRUE, apply regularisation techniques.

`sample_prior` Whether to sample from priors ("no", "yes", "only"). Default: "no".

`verbose` Logical; print progress information.

`...` Additional arguments passed to qbrms().

Value

A qbrms_fit object with additional regularisation metadata.

`qbrms_emmeans` *Estimated Marginal Means for qbrms models*

Description

Compute estimated marginal means (least-squares means) for factor terms and their combinations for a `qbrms_fit`, using a multivariate-normal approximation to the posterior of the fixed effects.

Usage

```
qbrms_emmeans(object, specs, at = NULL, nsim = 1000, prob = 0.95, ...)
```

Arguments

`object` A `qbrms_fit`.

`specs` Character vector naming factor(s) for EMMs, or a string containing a formula with a right-hand side (for example, `"~ group"` or `"y ~ group"`). If multiple terms are provided, a full grid is used.

`at` Optional named list giving values at which to hold other predictors. Numerics are fixed at their means if not supplied; factors at their modal level.

`nsim` Number of posterior draws for uncertainty.

`prob` Interval mass (default 0.95).

`...` Additional arguments (currently not used).

Value

A data frame of class `qbrms_emmeans`.

qbrms_fit_log *Get captured fit log from a qbrms object (if available)*

Description

Get captured fit log from a qbrms object (if available)

Usage

```
qbrms_fit_log(x)
```

Arguments

x	A qbrms_fit / qbrmO_fit object returned by qbrm()/qbrms()

Value

A character vector of captured console lines, or NULL if none

qbrms_ordinal_binary *Ordinal regression via binary decomposition (fallback)*

Description

Splits an ordered response with K levels into K-1 binary problems (thresholds y > c_j) and fits a simple binomial GLM for each split.

Usage

```
qbrms_ordinal_binary(formula, data, verbose = FALSE, ...)
```

Arguments

formula	Model formula with an ordered response on the LHS.
data	Data frame.
verbose	Logical; print progress messages.
...	Ignored (compat).

Value

An object of class `c("ordinal_binary_qbrms_fit","qbrms_fit")` with:

- `binary_models`: list of length K-1 of fitted `glm` objects
- `thresholds`: character vector of thresholds used
- `response`: response variable name
- `levels`: factor levels of the ordered response
- `ordinal_levels`: factor levels (for test compatibility)

qbrms_set_verbosity *Set qbrms verbosity for the current session*

Description

Set qbrms verbosity for the current session

Usage

```
qbrms_set_verbosity(verbose = FALSE)
```

Arguments

`verbose`	Logical. If TRUE, fitting prints progress; if FALSE, fitting is silent.

Value

Invisibly returns the previous value.

quick_density_comparison *Quick Density Comparison*

Description

Quick Density Comparison

Usage

```
quick_density_comparison(object, parameter = NULL, ...)
```

Arguments

`object`	A `qbrms_fit` object.
`parameter`	Optional parameter name to focus the comparison.
`...`	Additional arguments forwarded to `density_plot()`.

Value

A `ggplot2` object.

`random_walk_families` *Random Walk Families*

Description

Random Walk Families

Usage

```
rw1(scale.model = "log", diagonal = 1e-06)

rw2(scale.model = "log", diagonal = 1e-06)
```

Arguments

`scale.model`	Scaling model for precision
`diagonal`	Diagonal precision matrix structure

Value

A family object of class `"family"`.

`residuals.qbrms_fit` *Extract residuals from qbrms models*

Description

Extract residuals from qbrms models

Usage

```
## S3 method for class 'qbrms_fit'
residuals(object, type = "response", ...)
```

Arguments

`object`	A qbrms_fit object
`type`	Character string indicating type of residuals (default: "response")
`...`	Additional arguments (currently unused)

Value

Numeric vector of residuals

Examples

```
## Not run:
fit <- qbrms(mpg ~ hp, data = mtcars, family = gaussian())
resid_values <- residuals(fit, type = "response")

## End(Not run)
```

rope_analysis *ROPE analysis*

Description

Compute the posterior mass inside a Region Of Practical Equivalence (ROPE) for selected parameters.

Usage

```
rope_analysis(object, parameters = NULL, rope = c(-0.1, 0.1), nsim = 4000)
```

Arguments

`object`	A `qbrms_fit`.
`parameters`	Optional character vector of parameter names. If `NULL`, all fixed-effect coefficients are used.
`rope`	Numeric length-2 vector `c(lower, upper)`.
`nsim`	Number of posterior draws to simulate.

Value

A data frame of class `qbrms_rope`.

simplex *Simplex Family for Compositional Data*

Description

Simplex Family for Compositional Data

Usage

```
simplex(link = "logit", link.precision = "log")
```

Arguments

`link` Link function for the mean (default: "logit")

`link.precision` Link function for precision parameter (default: "log")

Value

A family object of class `"family"`.

`skew_normal` *Skew Normal Family*

Description

Skew Normal Family

Usage

```
skew_normal()
```

Value

An object of class `"family"` representing the Skew Normal distribution.

`student_t` *Student's t Family for Robust Regression*

Description

Student's t-distribution family for robust regression with heavier tails than Gaussian to handle outliers.

Functions that act as both family constructors (for `qbrm`) and prior specifications (for `prior`), depending on arguments.

Usage

```
student_t(
  link_or_df = "identity",
  location = 0,
  scale = 1,
  link = NULL,
  link.sigma = "log",
  link.nu = "log",
  ...
)
```

```
student()

student_t(
  link_or_df = "identity",
  location = 0,
  scale = 1,
  link = NULL,
  link.sigma = "log",
  link.nu = "log",
  ...
)

student_t_prior(
  link_or_df = 3,
  location = 0,
  scale = 1,
  link.sigma = "log",
  link.nu = "log"
)
```

Arguments

`link_or_df`	For family: link function (character). For prior: degrees of freedom (numeric).
`location`	Location parameter (prior only).
`scale`	Scale parameter (prior only).
`link`	Optional link function (if acting as family).
`link.sigma`	Link for sigma (family only).
`link.nu`	Link for nu (family only).
`...`	Additional arguments.

Value

An object of class `"family"` specifying the Student-t distribution.

An object of class `"family"` specifying the Student-t distribution.

A family object or prior object depending on inputs.

Examples

```
# Create a Student-t family object
fam <- student_t()
print(fam$family)
```

summary.qbrmb_fit *Summary Method for qbrmb_fit Objects*

Description

Provides a detailed summary of a regularised binomial qbrms model.

Usage

```
## S3 method for class 'qbrmb_fit'
summary(object, digits = 2, ...)
```

Arguments

`object`	A `qbrmb_fit` object.
`digits`	Number of decimal places for output (default: 2).
`...`	Additional arguments (currently unused).

Value

An object of class `"summary.qbrmb_fit"` containing model summary information.

summary.qbrms_fit *Summary Method for qbrms_fit Objects*

Description

Provides a detailed summary of a fitted qbrms model.

Usage

```
## S3 method for class 'qbrms_fit'
summary(object, ..., digits = 2)

## S3 method for class 'qbrms_fit'
summary(object, ..., digits = 2)
```

Arguments

`object`	A qbrms_fit object
`...`	Additional arguments
`digits`	Number of digits for output (default 2)

Value

An object of class "summary.qbrms_fit" containing:

- formula: The model formula.
- family: The distribution family.
- nobs: Number of observations.
- fixed: Data frame of fixed effects estimates.
- random: Random effects summary (if applicable).

Examples

```
if (requireNamespace("INLA", quietly = TRUE)) {
  fit <- qbrms(mpg ~ hp, data = mtcars, family = gaussian(), verbose = FALSE)
  summary(fit)
}
```

---

summary.qbrms_multinomial_fit

Summary method for multinomial qbrms fits

Description

Print a readable summary of a qbrms_multinomial_fit, including its reference category, the list of categories, and per-category fixed-effect summaries when available.

Usage

```
## S3 method for class 'qbrms_multinomial_fit'
summary(object, digits = 2, ...)
```

Arguments

object	A qbrms_multinomial_fit.
digits	Integer; number of decimal places to display.
...	Unused.

Value

The input object, returned invisibly.

summary.qbrms_p_significance
Summary Method for Enhanced p_significance

Description

Provide a summary of probability of practical significance results.

Usage

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

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

Arguments

`object`	A `qbrms_p_significance` object from `p_significance()`.
`...`	Additional arguments (currently unused).

Value

Invisibly returns the input object.

Invisibly returns the input object.

summary.tmb_ordinal_qbrms_fit
Summary Method for TMB Ordinal Fits

Description

Summary Method for TMB Ordinal Fits

Usage

```
## S3 method for class 'tmb_ordinal_qbrms_fit'
summary(object, digits = 2, ...)
```

Arguments

`object`	A tmb_ordinal_qbrms_fit object
`digits`	Number of decimal places for output
`...`	Additional arguments

Value

Invisibly returns the object

tab_model *Create HTML Table for qbrms Models with Enhanced Styling*

Description

Generate APA-style HTML tables for qbrms model outputs with customizable styling options.

Usage

```
tab_model(
  ...,
  show.ci = TRUE,
  ci.lvl = 0.95,
  show.rope = FALSE,
  rope = c(-0.1, 0.1),
  show.p_sig = FALSE,
  show.pd = FALSE,
  show.bf = FALSE,
  digits = 2,
  title = "Model Results",
  file = NULL,
  CSS = NULL,
  dv.labels = NULL,
  pred.labels = NULL,
  show.intercept = TRUE,
  show.r2 = FALSE,
  show.icc = FALSE,
  show.nobs = TRUE,
  bootstrap = TRUE,
  table.style = "default",
  font.family = "system-ui, -apple-system, sans-serif",
  font.size = "14px",
  header.bg = "#f8f9fa",
  stripe.bg = "#f9f9f9",
  verbose = FALSE
)
```

Arguments

`...`	One or more qbrms_fit objects to display in the table
`show.ci`	Logical; show credible intervals (default TRUE)
`ci.lvl`	Credible interval level (default 0.95)
`show.rope`	Logical; show ROPE analysis (default FALSE)

`rope`	Numeric vector c(lower, upper) for ROPE bounds
`show.p_sig`	Logical; show probability of practical significance (default FALSE)
`show.pd`	Logical; show probability of direction (default FALSE)
`show.bf`	Logical; show Bayes factors (default FALSE)
`digits`	Number of decimal places (default 2)
`title`	Character; table title
`file`	Character; file path to save HTML output (optional)
`CSS`	Character; custom CSS styling (optional)
`dv.labels`	Character vector of dependent variable labels
`pred.labels`	Named character vector for predictor labels
`show.intercept`	Logical; show intercept row (default TRUE)
`show.r2`	Logical; show R-squared if available (default FALSE)
`show.icc`	Logical; show ICC for mixed models (default FALSE)
`show.nobs`	Logical; show number of observations (default TRUE)
`bootstrap`	Logical; use Bootstrap CSS framework (default TRUE)
`table.style`	Character; table style theme. Options: "default", "minimal", "academic", "modern"
`font.family`	Character; CSS font family (default "system-ui")
`font.size`	Character; base font size (default "14px")
`header.bg`	Character; header background colour (default "#f8f9fa")
`stripe.bg`	Character; striped row background colour (default "#f9f9f9")
`verbose`	Logical; print progress (default FALSE)

Value

An object of class "qbrms_html_table" containing the HTML code

`test_corrected_bayes_R2` *Test the corrected implementation with a mixed-effects example*

Description

Test the corrected implementation with a mixed-effects example

Usage

```
test_corrected_bayes_R2()
```

Examples

```
## Not run:
# Test with mixed-effects model
library(qbrms)

# Create sample data with strong group effects
set.seed(123)
n_groups <- 10
n_per_group <- 20
n_total <- n_groups * n_per_group

data <- data.frame(
  group = factor(rep(1:n_groups, each = n_per_group)),
  x = rnorm(n_total),
  group_effect = rep(rnorm(n_groups, 0, 2), each = n_per_group)
)

# Generate response with strong group effects
data$y <- 2 + 0.5 * data$x + data$group_effect + rnorm(n_total, 0, 0.5)

# Fit mixed-effects model
fit_mixed <- qbrms(y ~ x + (1|group), data = data, family = gaussian())

# Compute Bayesian R-squared (should now match brms closely)
r2_corrected <- bayes_R2(fit_mixed, verbose = TRUE)
print(r2_corrected)

# Should show high R-squared due to strong group effects

## End(Not run)
```

uniform *Specify Uniform Prior Distribution*

Description

Specify Uniform Prior Distribution

Usage

```
uniform(min = -Inf, max = Inf)
```

Arguments

min	Lower bound (default -Inf)
max	Upper bound (default Inf)

Value

A prior distribution object

validate_family_quantile *Validate Family Quantile Combination*

Description

Check if a family supports quantile regression with a given quantile value. Throws informative errors for invalid combinations.

Usage

```
validate_family_quantile(family_name, quantile)
```

Arguments

`family_name`	Character string specifying the family name.
`quantile`	Numeric quantile value (or NULL).

Value

TRUE if the combination is valid (invisibly), throws error otherwise.

vcov.qbrms_fit *Extract Variance-Covariance Matrix from qbrms Models*

Description

Extract Variance-Covariance Matrix from qbrms Models

Usage

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

Arguments

`object`	A qbrms_fit object
`...`	Additional arguments (unused)

Value

Variance-covariance matrix

`vcov.tmb_ordinal_qbrms_fit` *Variance-Covariance Matrix Method for TMB Ordinal Fits*

Description

Variance-Covariance Matrix Method for TMB Ordinal Fits

Usage

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

Arguments

`object`	A tmb_ordinal_qbrms_fit object
`...`	Additional arguments

Value

Variance-covariance matrix

`view_table` *Display HTML Table in Viewer*

Description

Display HTML Table in Viewer

Usage

```
view_table(x)
```

Arguments

`x`	A `qbrms_html_table` created by `tab_model()`.

visualise_prior *Visualise Prior Distributions*

Description

Create visual representations of prior distributions to aid in prior specification and sensitivity analysis.

Usage

```
visualise_prior(
  prior,
  parameter = NULL,
  xlim = NULL,
  add_reference = TRUE,
  samples = 10000
)
```

Arguments

`prior`	Prior specification in qbrms format, or a list of prior specifications to compare
`parameter`	Character string specifying which parameter to visualise (e.g., "b", "sd", "sigma"). If NULL, visualises all priors.
`xlim`	Numeric vector of length 2 specifying x-axis limits. If NULL, automatically determined.
`add_reference`	Logical; if TRUE, adds reference distributions for comparison (default: TRUE)
`samples`	Number of samples to draw for visualisation (default: 10000)

Details

This function helps users:

- Visualise the implications of their prior choices
- Compare different prior specifications
- Identify overly informative or vague priors
- Understand prior-data conflict potential

Supported prior distributions include:

- Normal: normal(mean, sd)
- Student t: student_t(df, mean, scale)
- Cauchy: cauchy(location, scale)
- Exponential: exponential(rate)
- Gamma: gamma(shape, rate)
- Uniform: uniform(lower, upper)

Value

A ggplot object showing the prior distribution(s)

Examples

```
## Not run:
# Visualise a single prior
prior <- prior(normal(0, 10), class = "b")
visualise_prior(prior)

# Compare different priors
prior_list <- list(
  "Weak" = prior(normal(0, 10), class = "b"),
  "Medium" = prior(normal(0, 5), class = "b"),
  "Strong" = prior(normal(0, 1), class = "b")
)
visualise_prior(prior_list)

# Visualise with custom limits
visualise_prior(prior, xlim = c(-20, 20))

## End(Not run)
```

weibull *Weibull Survival Family*

Description

Weibull Survival Family

Usage

```
weibull(link = "log", link.shape = "log")
```

Arguments

`link`	Link function for scale (default: "log")
`link.shape`	Link function for shape (default: "log")

Value

A family object of class "`family`".

zero_inflated_negbinomial *Zero-Inflated Negative Binomial Family*

Description

Zero-Inflated Negative Binomial Family

Usage

```
zero_inflated_negbinomial(link = "log", link.zi = "logit")

zinb(link = "log", link.zi = "logit")
```

Arguments

`link`	Link function for mean (default: "log")
`link.zi`	Link function for zero-inflation (default: "logit")

Value

A family object of class `"family"`.

zero_inflated_poisson *Zero-Inflated Poisson Family*

Description

Zero-Inflated Poisson Family

Usage

```
zero_inflated_poisson(link = "log", link.zi = "logit")

zip()
```

Arguments

`link`	Link function for mean (default: "log")
`link.zi`	Link function for zero-inflation (default: "logit")

Value

A family object of class `"family"`.

Index