Package ‘bayesics’

February 6, 2026

Title Bayesian Analyses for One- and Two-Sample Inference and Regression Methods

Version 2.0.2

Maintainer Daniel K. Sewell `<daniel-sewell@uiowa.edu>`

Description Perform fundamental analyses using Bayesian parametric and non-parametric inference (regression, anova, 1 and 2 sample inference, non-parametric tests, etc.). (Practically) no Markov chain Monte Carlo (MCMC) is used; all exact finite sample inference is completed via closed form solutions or else through posterior sampling automated to ensure precision in interval estimate bounds. Diagnostic plots for model assessment, and key inferential quantities (point and interval estimates, probability of direction, region of practical equivalence, and Bayes factors) and model visualizations are provided. Bayes factors are computed either by the Savage Dickey ratio given in Dickey (1971) <doi:10.1214/aoms/1177693507> or by Chib's method as given in xxx. Interpretations are from Kass and Raftery (1995) <doi:10.1080/01621459.1995.10476572>. ROPE bounds are based on discussions in Kruschke (2018) <doi:10.1177/2515245918771304>. Methods for determining the number of posterior samples required are described in Doss et al. (2014) <doi:10.1214/14-EJS957>. Bayesian model averaging is done in part by Feldkircher and Zeugner (2015) <doi:10.18637/jss.v068.i04>. Methods for contingency table analysis is described in Gunel et al. (1974) <doi:10.1093/biomet/61.3.545>. Variational Bayes (VB) methods are described in Salimans and Knowles (2013) <doi:10.1214/13-BA858>. Mediation analysis uses the framework described in Imai et al. (2010) <doi:10.1037/a0020761>. The loss-likelihood bootstrap used in the non-parametric regression modeling is described in Lyddon et al. (2019) <doi:10.1093/biomet/asz006>. Non-parametric survival methods are described in Qing et al. (2023) <doi:10.1002/pst.2256>. Methods used for the Bayesian Wilcoxon signed-rank analysis is given in Chechile (2018) <doi:10.1080/03610926.2017.1388402> and for the Bayesian Wilcoxon rank sum analysis in Chechile (2020) <doi:10.1080/03610926.2018.1549247>. Correlation analysis methods are carried out by Barch and Chechile (2023) <doi:10.32614/CRAN.package.DFBA>, and described in Lindley and Phillips (1976) <doi:10.1080/00031305.1976.10479154> and Chechile and Barch (2021) <doi:10.1016/j.jmp.2021.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1.0)

RoxygenNote 7.3.3

Suggests datasets, rstanarm, knitr, splines, testthat (>= 3.0.0)

Imports tidyr, dplyr, rlang, janitor, extraDistr, mvtnorm, Matrix, future, future.apply, ggplot2, patchwork, BMS, cluster, DFBA, tibble, survival

Config/testthat/edition 3

URL https://github.com/dksewell/bayesics

BugReports https://github.com/dksewell/bayesics/issues

NeedsCompilation no

Author Daniel K. Sewell [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-9238-4026>), Alan Arakkal [aut] (ORCID: <https://orcid.org/0000-0002-7001-493X>)

Repository CRAN

Date/Publication 2026-02-06 19:10:02 UTC

Contents

aov_b 3
bayes_factors 6
bma_inference 8
case_control_b 10
chisq_test_b 12
coef 14
cor_test_b 15
credint 18
find_beta_parms 19
find_invgamma_parms 20
get_posterior_draws 21
glm_b 22
heteroscedasticity_test 26
IC 28
lm_b 30
mediate_b 33
negbinom 36
np_glm_b 37
plot 39
poisson_test_b 42
predict.aov_b 44
predict.glm_b 45
predict.lm_b 46
predict.lm_b_bma 47
predict.np_glm_b 49
print 50
prop_test_b 51

sign_test_b 53
summary 55
Surv 56
survfit_b 57
t_test_b 59
vcov 61
wilcoxon_test_b 62

Index **66**

`aov_b` *Analysis of Variance using Bayesian methods*

Description

Analysis of Variance using Bayesian methods

Usage

```
aov_b(
  formula,
  data,
  heteroscedastic = TRUE,
  prior_mean_mu,
  prior_mean_nu = 0.001,
  prior_var_shape = 0.001,
  prior_var_rate = 0.001,
  CI_level = 0.95,
  ROPE = 0.1,
  contrasts,
  improper = FALSE,
  seed = 1,
  mc_error = 0.002,
  compute_bayes_factor = TRUE
)
```

Arguments

`formula`	A formula specifying the model.
`data`	A data frame in which the variables specified in the formula will be found. If missing, the variables are searched for in the standard way.
`heteroscedastic`	logical. Set to FALSE to assume all groups have equal variance.
`prior_mean_mu`	numeric. Hyperparameter for the a priori mean of the group means.
`prior_mean_nu`	numeric. Hyperparameter which scales the precision of the group means.

`prior_var_shape`	numeric. Twice the shape parameter for the inverse gamma prior on the residual variance(s). I.e., $\sigma^2 \sim IG$(prior_var_shape/2,prior_var_rate/2).
`prior_var_rate`	numeric. Twice the rate parameter for the inverse gamma prior on the residual variance(s). I.e., $\sigma^2 \sim IG$(prior_var_shape/2,prior_var_rate/2).
`CI_level`	numeric. Credible interval level.
`ROPE`	numeric. Used to compute posterior probability that Cohen's D +/- ROPE
`contrasts`	numeric/matrix. Either vector of length equal to the number of levels in the grouping variable, or else a matrix where each row is a separate contrast, and the number of columns match the number of levels in the grouping variable.
`improper`	logical. Should we use an improper prior that is proportional to the inverse of the variance?
`seed`	integer. Always set your seed!!!
`mc_error`	The number of posterior draws will ensure that with 99% probability the bounds of the credible intervals will be within $\pm$ `mc_error`$\times 4 s_y$, that is, within 100`mc_error`% of the trimmed range of y.
`compute_bayes_factor`	logical. Computing the BF can be done analytically, but it requires an nxn matrix. If this will require more than 1GB of memory, compute_bayes_factor will automatically be set to FALSE. This setting can be overridden by setting `compute_bayes_factor="force"`.

Details

MODEL: The likelihood model is given by

$$y_{gi} \overset{iid}{\sim} N(\mu_g, \sigma_g^2),$$

(although if `heteroscedastic` is set to FALSE, $\sigma_g^2 = \sigma_h^2 \; \forall g, h$).

The prior is given by

$$\mu_g|\sigma_g^2 \overset{iid}{\sim} N\left(\mu, \frac{\sigma_g^2}{\nu}\right), \sigma_g^2 \overset{iid}{\sim} \Gamma^{-1}(a/2, b/2),$$

where mu is set by `prior_mean_mu`, nu is set by `prior_mean_nu`, a is set by `prior_var_shape`, and b is set by `prior_var_rate`.

The posterior is

$$\mu_g|y, \sigma_g^2 \overset{iid}{\sim} N\left(\hat{\mu}_g, \frac{\sigma_g^2}{\nu_g}\right), \sigma_g^2|y \overset{iid}{\sim} \Gamma^{-1}(a_g/2, b_g/2),$$

where $\hat{\mu}_g$, ν_g, a_g, and b_g are all returned by `aov_b` in the named element `posterior_parameters`.

ROPE:

If missing, the ROPE bounds will be given under the principle of "half of a small effect size." Using Cohen's D of 0.2 as a small effect size, the ROPE is defined in terms of $-0.1 <$ Cohen's D < 0.1.

Value

Object of class "aov_b" with the following elements:

- summary - tibble giving the summary of the model parameters
- BF_for_different_vs_same_means - Bayes factor in favor of the full model (each group has their own mean) vs. the null model (all groups have the same mean).
- pairwise summary - tibble giving the summary comparing all factor level means
- contrasts (if provided) - list with named elements L (the contrasts provided by the user) and summary.
- posterior_draws - mcmc object (see coda package) giving the posterior draws
- posterior_parameters -
  - mu_g - the post. means of the group means
  - nu_g - the post. scalars of the precision
  - a_g - (twice) the post. shape of the inv. gamma for the group variances
  - b_g - (twice) the post. rate of the inv. gamma for the group variances.
- hyperparameters -
  - mu - the prior mean of the group means
  - nu - the prior scalar of the precision
  - a - (twice) the prior shape of the inv. gamma for the group variances
  - b - (twice) the prior rate of the inv. gamma for the group variances.
- `fitted` - Posterior mean of $\mu_g := \mathbb{E}(y_{gi})$
- `residuals` - Posterior mean of the residuals
- `standardized_residuals` - Estimated residuals divided by the group standard deviation
- `mc_error` - absolute errors used to determine number of posterior draws for accurate interval estimation

References

Charles R. Doss, James M. Flegal, Galin L. Jones, Ronald C. Neath "Markov chain Monte Carlo estimation of quantiles," Electronic Journal of Statistics, Electron. J. Statist. 8(2), 2448-2478, (2014)

Examples

```
# Create data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rep(letters[1:5],N/5))
test_data$outcome =
  rnorm(N,-1 + 2 * (test_data$x1 %in% c("d","e")) )

# Fit 1-way ANOVA model
fit1 <-
  aov_b(outcome ~ x1,
        test_data,
        prior_mean_mu = 2,
        prior_mean_nu = 0.5,
        prior_var_shape = 0.01,
        prior_var_rate = 0.01)
fit1
summary(fit1)
plot(fit1)
coef(fit1)
credint(fit1)
credint(fit1,
        CI_level = 0.99)
vcov(fit1)
fit1_predictions <-
  predict(fit1,
          CI_level = 0.99,
          PI_level = 0.9)
AIC(fit1)
BIC(fit1)
DIC(fit1)
WAIC(fit1)

# Implement contrasts
## One contrast
fit2 <-
  aov_b(outcome ~ x1,
        test_data,
        mc_error = 0.01,
        contrasts = c(-1/3,-1/3,-1/3,1/2,1/2))
fit2$contrasts
summary(fit2)
## Multiple contrasts
fit3 <-
  aov_b(outcome ~ x1,
        test_data,
        mc_error = 0.01,
        contrasts = rbind(c(-1/3,-1/3,-1/3,1/2,1/2),
                          c(-1/3,-1/3,-1/3,1,0)))
fit3$contrasts
summary(fit3)
```

`bayes_factors` *Bayes factors for lm_b, glm_b, and survfit_b*

Description

Bayes factors for Bayesian regression objects using the Savage-Dickey ratio

Usage

```
bayes_factors(object, ...)

## S3 method for class 'lm_b'
bayes_factors(object, by = "coefficient", ...)

## S3 method for class 'glm_b'
bayes_factors(object, by = "coefficient", ...)

## S3 method for class 'survfit_b'
bayes_factors(object, object2, ...)
```

Arguments

`object`	lm_b, glm_b, or survfit_b object
`...`	Passed to methods.
`by`	character. Either "coefficient" or "variable". If the former, Bayes factors will be computed for each regression coefficient separately. If the latter, Bayes factors will be computed for each covariate separately.
`object2`	a second survfit_b object. Not used for other classes.

Details

Bayes factors are given in terms of favoring the two-tailed alternative hypothesis vs. the null hypothesis that the regression coefficient equals zero. Currently implemented for `lm_b` or `glm_b` objects. Note that for `glm_b` objects, if importance sampling was used, the model will be refit using fixed form variational Bayes to get the multivariate posterior density.

Interpretation is taken from Kass and Raftery.

Value

A tibble with Bayes factors and interpretations.

References

James M. Dickey. "The Weighted Likelihood Ratio, Linear Hypotheses on Normal Location Parameters." Ann. Math. Statist. 42 (1) 204 - 223, February, 1971. https://doi.org/10.1214/aoms/1177693507

Kass, R. E., & Raftery, A. E. (1995). Bayes Factors. Journal of the American Statistical Association, 90(430), 773–795.

Examples

```
# Generate some binomial data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
                 x3 = letters[1:5])
test_data$outcome =
  rbinom(N,1,1.0 / (1.0 + exp(-(-2 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) ))))


# Fit a GLM
fit <-
  glm_b(outcome ~ x1 + x2 + x3,
        data = test_data,
        family = binomial(),
        seed = 2025)

# Compute the BF for each coefficient
bayes_factors(fit)

# Compute the BF for each variable
bayes_factors(fit,
              by = "variable")
```

`bma_inference` *Bayesian model averaging*

Description

Estimates and CIs from BMA

Usage

```
bma_inference(
  formula,
  data,
  zellner_g = nrow(data),
  CI_level = 0.95,
  ROPE,
  mcmc_draws = 10000,
  n_models = 500,
  mc_error = 0.001,
  seed = 1,
  ...
)
```

Arguments

`formula`	A formula specifying the model.
`data`	Data used in linear regression model
`zellner_g`	numeric. Positive number giving the value of "g" in Zellner's g prior.

`CI_level`	Level for credible interval
`ROPE`	vector of positive values giving ROPE boundaries for each regression coefficient. Optionally, you can not include a ROPE boundary for the intercept. If missing, defaults go to those suggested by Kruchke (2018).
`mcmc_draws`	Integer. Number of draws passed into `bms`
`n_models`	Integer. The number of best models for which information is stored. See `bms` for more details.
`mc_error`	The number of posterior draws will ensure that with 99% probability the bounds of the credible intervals will be within $\pm$ `mc_error`$\times 4 s_y$, that is, within 100`mc_error`% of the trimmed range of y.
`seed`	Integer. Always set your seed!!!
`...`	Other arguments for `bms`.

Details

`bma_inference` leverages the `bms` function from its eponymous R package, and then uses `lm_b` to obtain inference on the regression coefficients for Bayesian model averaging.

Value

A list with the following elements:

- summary Tibble with point and interval estimates
- lm_b_fits A list of lm_b fits using zellner's g prior for all the top models from `bms`
- hyperparameters A named list with the user-specified zellner's g value.
- posterior_draws matrix of posterior draws of the regression parameters, marginalizing out the model

Examples

```
# Create data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5],
             x4 = rnorm(N),
             x5 = rnorm(N),
             x6 = rnorm(N),
             x7 = rnorm(N),
             x8 = rnorm(N),
             x9 = rnorm(N),
             x10 = rnorm(N))
test_data$outcome =
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )

# Fit linear model using Bayesian model averaging
```

```
fit <-
  bma_inference(outcome ~ .,
                test_data,
                user.int = FALSE)
summary(fit)
coef(fit)
credint(fit)
plot(fit)
```

`case_control_b` *Case-Control Analysis*

Description

Bayesian analysis of a case-control study (without covariates).

Usage

```
case_control_b(
  cases,
  controls,
  x,
  large_sample_approx,
  ROPE,
  prior_mean = 0,
  prior_sd = log(10)/1.96,
  plot = TRUE,
  CI_level = 0.95,
  seed = 1,
  mc_error = 0.005
)
```

Arguments

`cases`	vector of length 2, giving the numbers at risk and not at risk, respectively, for cases
`controls`	vector of length 2, giving the numbers at risk and not at risk, respectively, for controls
`x`	2x2 contingency table. The rows should depict the at risk status (first row is at risk, second row is not at risk), and the columns should depict the case control status (first column is case, second column is control).
`large_sample_approx`	If all cell counts of `x` are not too low (≥ 5) then use the approximation that the empirical log odds are normally distributed. (See details for more.) If missing, this will be set to `TRUE` iff all cell counts are greater than or equal to 5.

`ROPE`	ROPE for odds ratio. Provide either a single value or a vector of length two. If the former, the ROPE will be taken as (1/ROPE,ROPE). If the latter, these will be the bounds of the ROPE.
`prior_mean`	numeric. The prior mean on the log odds ratio
`prior_sd`	numeric. The prior sd on the log odds ratio. See details for default values.
`plot`	logical. Should a plot be shown?
`CI_level`	The posterior probability to be contained in the credible interval.
`seed`	integer. Always set your seed!!! (ignored if `large_sample_approx` = `TRUE`.)
`mc_error`	The relative monte carlo error of the quantiles of the CIs. (ignored if `large_sample_approx` = `TRUE`.)

Details

If `large_sample_approx` = `TRUE` (the default if left missing and all cell counts are at least 5), then the likelihood is

$$\log(\hat{\omega}) \sim N\left(\log(\omega), \frac{1}{n_{11}} + \frac{1}{n_{12}} + \frac{1}{n_{21}} + \frac{1}{n_{22}}\right),$$

where ω is the odds ratio, $\hat{\omega}$ is the empirical odds ratio, n_{ij}, $i, j = 1, 2$ are the cells of the 2x2 contingency table. The prior on $\log \omega$ is

$$\log \omega \sim N(a, b^2).$$

If the large sample approximation is not used, then inference is made on the odds ratio by instead putting uniform priors on $\Pr(exposure|outcome)$.

Value

(returned invisible) list including the following:

- `data`: data
- `posterior_mean`: posterior mean of the odds ratio (cases vs. controls)
- `CI`: Credible interval
- `Pr_oddsratio_in_ROPE`: Probability the odds ratio (cases vs. controls) is in the ROPE
- `posterior_draws`: posterior draws of the odds ratio (cases vs. controls)
- `or_plot`: odds ratio (cases vs. controls) posterior plot

Examples

```
case_control_b(matrix(c(8,47,1,26),2,2))

case_control_b(c(8,47),
               c(1,26))
```

chisq_test_b *Test of independence for 2-way contingency tables*

Description

Test of independence for 2-way contingency tables

Usage

```
independence_b(
  x,
  sampling_design = "multinomial",
  ROPE,
  prior = "jeffreys",
  prior_shapes,
  CI_level = 0.95,
  seed = 1,
  mc_error = 0.002
)
```

Arguments

`x`	Either a table or a matrix of counts
`sampling_design`	Either "multinomial", "fixed rows", or "fixed columns"
`ROPE`	vector of positive values giving ROPE boundaries for each regression.
`prior`	Either "jeffreys" (Dirichlet(1/2)) or "uniform" (Dirichlet(1)). This is ignored if prior_shapes is provided.
`prior_shapes`	Either a single positive scalar, in which case a symmetric Dirichlet is used, or else a matrix matching the dimensions of x or a vector of length `prod(dim(x))`.
`CI_level`	The posterior probability to be contained in the credible interval.
`seed`	Always set your seed!
`mc_error`	This is the error in probability from the posterior CDF evaluated at the ROPE bounds. Note that if it is estimated that these probabilities are between 0.11 and 0.89, the more relaxed value of 0.01 is used.

Details

For a 2-way contingency table with R rows and C columns, evaluate the probability that

- the joint probabilities p_{ij} are all within the ROPE of $p_{i.} \times p_{.j}$ for `sampling_design = "multinomial"`
- the probabilities $p_{j|i}$ are all within the ROPE of $p_{.j}$ if `sampling_design = "fixed rows"` or `"fixed columns"`

Value

(returned invisible) A list with the following elements:

- `posterior_shapes`: posterior Dirichlet shape parameters
- `posterior_mean`: posterior mean
- `lower_bound`: lower credible interval bounds
- `upper_bound`: upper credible interval bounds
- `individual_ROPE`: Probability that joint probabilities are in the ROPE around independent probabilities
- `overall_ROPE`: Overall probability of falling in the ROPE (i.e., all probabilities are near the product of the marginal probabilities)
- `prob_pij_less_than_p_i_times_p_j`: (If multinomial sampling design) Probabilities that each joint probability is less than the product of the marginal probabilities
- `prob_p_j_given_i_less_than_p_j`: (If fixed rows or columns sampling design) Probabilities that each conditional probability is less than the marginal probabilities
- `prob_direction`: Probability of direction for the joint or conditional (depending on sampling scheme) probabilities (based on `prob_pij_less_than_p_i_times_p_j` or `prob_p_j_given_i_less_than_p_j`)
- `BF_for_dependence_vs_independence`: Bayes factor testing dependence vs. independence (higher values favor dependence, lower values favor independence)
- `BF_evidence`: Kass and Raftery's interpretation of the level of evidence of the Bayes factor

References

Gunel, Erdogan & Dickey, James (1974). Bayes factors for independence in contingency tables, Biometrika, 61(3), Pages 545–557, https://doi.org/10.1093/biomet/61.3.545

Kass, R. E., & Raftery, A. E. (1995). Bayes Factors. Journal of the American Statistical Association, 90(430), 773–795.

Examples

```
# Generate data
set.seed(2025)
N = 500
nR = 5
nC = 3
dep_probs =
  extraDistr::rdirichlet(1,rep(2,nR*nC)) |>
  matrix(nR,nC)

# Multinomial sampling
## Test independence
independence_b(round(N * dep_probs))

## Use other priors
independence_b(round(N * dep_probs),
               prior = "uniform")
independence_b(round(N * dep_probs),
               prior_shapes = 2)
independence_b(round(N * dep_probs),
               prior_shapes = matrix(1:(nR*nC),nR,nC))

# Fixed marginals
independence_b(round(N * dep_probs),
               sampling_design = "rows")
independence_b(round(N * dep_probs),
               sampling_design = "cols")
```

coef — *Coefficient extraction for bayesics objects*

Description

Coefficient extraction for bayesics objects

Usage

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

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

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

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

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

Arguments

`object`	bayesics object
`...`	optional arguments.

Value

vector of coefficients

Examples

```
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rep(letters[1:5],N/5))
test_data$outcome =
  rnorm(N,-1 + 2 * (test_data$x1 %in% c("d","e")) )

# Fit 1-way ANOVA model
fit1 <-
  aov_b(outcome ~ x1,
        test_data,
        prior_mean_mu = 2,
        prior_mean_nu = 0.5,
        prior_var_shape = 0.01,
        prior_var_rate = 0.01)
coef(fit1)
```

`cor_test_b` *Test for Association/Correlation Between Paired Samples via Kendall's tau*

Description

Test for Association/Correlation Between Paired Samples via Kendall's tau

Usage

```
cor_test_b(x, ...)

## Default S3 method:
cor_test_b(
  x,
  y,
  tau = 0,
  ROPE,
  prior = "centered",
  prior_shapes,
  CI_level = 0.95,
  plot = TRUE,
  ...
)

## S3 method for class 'formula'
cor_test_b(
  formula,
  data,
  tau = 0,
  ROPE,
  prior = "centered",
  prior_shapes,
  CI_level = 0.95,
  plot = TRUE,
  ...
)
```

Arguments

`x, y`	numeric vectors of data values. x and y must have the same length.
`...`	optional arguments.
`tau`	If provided, cor_test_b will return the posterior probability that Kendall's tau is less than this value.
`ROPE`	If a single number, ROPE will be $\tau\pm$ ROPE. If a vector of length 2, these will serve as the ROPE bounds. Defaults to $\pm$0.05.
`prior`	Beta prior used on phi (see details). Either "uniform' (Beta(1,1)), "centered' (Beta(2,2)), "positive" (Beta(3.9,2), putting 80% of the prior mass above 0.5), or "negative" (Beta(2,3.9), putting 80% of the prior mass below 0.5).
`prior_shapes`	Vector of length two, giving the shape parameters for the beta distribution that will act as the prior on ϕ (see details).
`CI_level`	The posterior probability to be contained in the credible interval.
`plot`	logical. Should a plot be shown?
`formula`	ADD description!
`data`	ADD description!

Details

cor_test_b relies on the robust Kendall's tau, defined to be

$$\tau := \frac{(\#\text{concordant pairs}) - (\#\text{discordant pairs})}{(\#\text{concordant pairs}) - (\#\text{discordant pairs})},$$

where a concordant pair is a pair of points such that if the rank of the x values is higher for the first (second) point of the pair, so too the rank of the y value is higher for the first (second) point of the pair.

The Bayesian approach of Chechile (2020) puts a Beta prior on phi, the proportion of concordance, i.e.,

$$\phi := \frac{(\#\text{concordant pairs})}{(\#\text{concordant pairs}) - (\#\text{discordant pairs})}.$$

The relationship between the two, then, is $\tau = 2\phi - 1$, or equivalently $\phi = (\tau + 1)/2$.

For more information, see dfba_bivariate_concordance and vignette("dfba_bivariate_concordance",package = "DFBA").

Value

(returned invisible) A list with the following:

- `posterior_mean`: posterior mean of Kendall's tau
- `CI`: Credible interval bounds
- `Pr_less_than_tau`: Posterior probability that Kendall's tau is less than provided reference `tau`
- `Pr_in_ROPE`: Posterior probability that Kendall's tau is in the ROPE
- `prob_plot`: Posterior and prior plot
- `posterior_parameters`: The posterior for Kendall's tau is a location shift and scaled beta distribution to fall over the range -1 to 1, i.e., $0.5 * (\tau + 1.0)$ follows a beta with shape parameters given by `posterior_parameters`.
- `dfba_bivariate_concordance_object`: The underlying object from the DFBA package

References

Chechile, R.A. (2020). Bayesian Statistics for Experimental Scientists: A General Introduction Using Distribution_Free Statistics. Cambridge: MIT Press.

Chechile, R.A., & Barch, D.H. (2021). A distribution-free, Bayesian goodness-of-fit method for assessing similar scientific prediction equations. Journal of Mathematical Psychology. https://doi.org/10.1016/j.jmp.2021.1026

Lindley, D. V., & Phillips, L. D. (1976). Inference for a Bernoulli process (a Bayesian view). The American Statistician, 30, 112-119.

Barch DH, Chechile RA (2023). DFBA: Distribution-Free Bayesian Analysis. doi:10.32614/CRAN.package.DFBA

Examples

```
# Generate data
set.seed(2025)
N = 50
x = rnorm(N)
y = x + 4 * rnorm(N)

# Test for non-zero correlation
cor_test_b(x,y)

# Input can be in the form of formula and data
cor_test_b(~ asdf + qwer,
           data = data.frame(asdf = x,
                             qwer = y))

# Other priors can be used, also.  See help for details.
cor_test_b(x,y,
           prior = "uniform")
cor_test_b(x,y,
           prior = "negative")
cor_test_b(x,y,
           prior = "positive")
cor_test_b(x,y,
            prior_shapes = c(10,10))
```

credint — *Credible Intervals for Model Parameters*

Description

Computes credible intervals for one or more parameters in a fitted model.

Usage

```
credint(object, ...)

## S3 method for class 'lm_b'
credint(object, CI_level = 0.95, ...)

## S3 method for class 'aov_b'
credint(object, CI_level = 0.95, which = "means", ...)

## S3 method for class 'glm_b'
credint(object, CI_level = 0.95, ...)

## S3 method for class 'np_glm_b'
credint(object, CI_level = 0.95, ...)

## S3 method for class 'lm_b_bma'
credint(object, CI_level = 0.95, ...)
```

Arguments

`object`	a fitted model object from `bayesics`
`...`	Passed to methods.
`CI_level`	the credible level required
`which`	character. For `aov_b` only. Either "means" (for the group means) or "pairwise" (for pairwise difference in means).

Value

Matrix of credible intervals

Examples

```
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rep(letters[1:5],N/5))
test_data$outcome =
  rnorm(N,-1 + 2 * (test_data$x1 %in% c("d","e")) )

# Fit 1-way ANOVA model
fit1 <-
  aov_b(outcome ~ x1,
        test_data,
        prior_mean_mu = 2,
        prior_mean_nu = 0.5,
        prior_var_shape = 0.01,
        prior_var_rate = 0.01)
credint(fit1)
```

find_beta_parms *Find parameters for Beta prior based on prior mean and one quantile*

Description

Find parameters for Beta prior based on prior mean and one quantile

Usage

```
find_beta_parms(
  mean,
  quantile,
  left_tail_prob,
  plot_results = TRUE,
  search_bounds = c(0.001, 100)
)
```

Arguments

mean	numeric between 0 and 1 giving the prior mean
quantile	numeric between 0 and 1 giving the quantile lying at left_tail_prob
left_tail_prob	numeric between 0 and 1 giving the prior probability of theta being less than or equal to quantile
plot_results	logical. Should the resulting inverse gamma distribution be plotted?
search_bounds	bounds with which to search. Sometimes you need to adjust this to get a good solution.

Value

Vector of beta shape parameters

Examples

```
find_beta_parms(2/5,0.68,0.9)
2/ (2 + 3)
qbeta(0.9,2,3)
```

find_invgamma_parms *Find parameters for Inverse gamma prior based on prior mean and one quantile*

Description

Find parameters for Inverse gamma prior based on prior mean and one quantile

Usage

```
find_invgamma_parms(
  lower_quantile,
  upper_quantile,
  response_variance,
  lower_R2,
  upper_R2,
  probability,
  plot_results = TRUE
)
```

Arguments

`lower_quantile`	lower quantile desired
`upper_quantile`	upper quantile desired
`response_variance`	variance of the response variable of the regression model
`lower_R2, upper_R2`	We are a priori `probability` sure that the coefficient of determination (R^2) falls within these lower and upper bounds.
`probability`	prior probability to be contained within the lower and upper quantiles
`plot_results`	logical. Should the resulting inverse gamma distribution be plotted?

Details

Either provide the lower and upper quantiles that contain `probability` of the inverse gamma distribution, or if this is for linear regression, you can specify that you are a priori `probability` sure that the coefficient of determination (R^2) falls within the two bounds provided, assuming that the residual variance is $1 - R^2$ times the total variance.

Value

twice the shape and rate of the inverse gamma distribution.

Examples

```
# When aimed at linear regression via coefficient of determination...
hypothetical_s2_y = 2.0
lower_R2 = 0.05
upper_R2 = 0.85
find_invgamma_parms(response_variance = hypothetical_s2_y,
                    lower_R2 = lower_R2,
                    upper_R2 = upper_R2,
                    probability = 0.8)

# More arbitrary task...
find_invgamma_parms(0.3, # hypothetical_s2_y * (1.0 - upper_R2)
                    1.9, #hypothetical_s2_y * (1.0 - lower_R2)
                    probability = 0.8)
```

`get_posterior_draws` *Get posterior samples from lm_b object*

Description

Get posterior samples from lm_b object

Usage

```
get_posterior_draws(object, n_draws = 10000, seed = 1)
```

Arguments

`object`	Object of class lm_b
`n_draws`	integer. Number of posterior draws to obtain.
`seed`	integer. Always set your seed!!!

Value

matrix of posterior draws

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
pdraws <-
  get_posterior_draws(fit1)
```

---

glm_b *Bayesian Generalized Linear Models*

Description

glm_b is used to fit linear models. It can be used to carry out regression for gaussian, binomial, and poisson data. Note that if the family is gaussian, this is just a wrapper for `lm_b`.

Usage

```
glm_b(
  formula,
  data,
  family,
  trials,
  prior = c("zellner", "normal", "improper")[1],
  zellner_g,
  prior_beta_mean,
  prior_beta_precision,
  prior_phi_mean = 1,
  ROPE,
  CI_level = 0.95,
  vb_maximum_iterations = 1000,
  algorithm = "VB",
  proposal_df = 5,
  seed = 1,
  mc_error = 0.01,
  save_memory = FALSE
)
```

Arguments

`formula`	A formula specifying the model.
`data`	A data frame in which the variables specified in the formula will be found. If missing, the variables are searched for in the standard way. However, it is strongly recommended that you use this argument so that other generics for bayesics objects work correctly.
`family`	A description of the error distribution and link function to be used in the model. See ?glm for more information. Currently implemented families are `binomial()`, `poisson()`, `negbinom()`, and `gaussian()` (this last acts as a wrapper for `lm_b`. If missing `family`, `glm_b` will try to infer the data type; negative binomial will be used for count data.
`trials`	Either character naming the variable in `data` that corresponds to the number of trials in the binomial observations, or else an integer vector giving the number of trials for each observation.
`prior`	character. One of "zellner", "normal", or "improper", giving the type of prior used on the regression coefficients.
`zellner_g`	numeric. Positive number giving the value of "g" in Zellner's g prior. Ignored unless prior = "zellner". Default is the number of observations.
`prior_beta_mean`	numeric vector of same length as regression coefficients (denoted p). Unless otherwise specified, automatically set to rep(0,p). Ignored unless prior = "normal".
`prior_beta_precision`	pxp matrix giving a priori precision matrix to be scaled by the residual precision. Ignored unless prior = "normal".
`prior_phi_mean`	For negative binomial distributed outcomes, an exponential distribution is used for the prior of the dispersion parameter phi, parameterized such that $\text{Var}(y) = \mu + \frac{\mu^2}{\phi}$, so that the prior on ϕ is $\lambda e^{-\lambda\phi}$, where λ equals 1/`prior_phi_mean`.
`ROPE`	vector of positive values giving ROPE boundaries for each regression coefficient. Optionally, you can not include a ROPE boundary for the intercept. If missing, defaults go to those suggested by Kruchke (2018).
`CI_level`	numeric. Credible interval level.
`vb_maximum_iterations`	if `algorithm` = `"VB"`, the number of iterations used in the fixed-form VB algorithm.
`algorithm`	Either "VB" (default) for fixed-form variational Bayes, "IS" for importance sampling, or "LSA" for large sample approximation.
`proposal_df`	degrees of freedom used in the multivariate t proposal distribution if `algorithm = "IS"`.
`seed`	integer. Always set your seed!!! Not used for `algorithm = LSA`.

`mc_error`	If importance sampling is used, the number of posterior draws will ensure that with 99% probability the bounds of the credible intervals will be within $\pm$ `mc_error`.
`save_memory`	logical. If TRUE, a more memory efficient approach will be taken at the expense of computataional time (for important sampling only. But if memory is an issue, it's probably because you have a large sample size, in which case the normal approximation sans IS should probably work.)

Value

glm_b() returns an object of class "glm_b", which behaves as a list with the following elements:

- `summary` - tibble giving results for regression coefficients
- `posterior_draws`
- `ROPE`
- `hyperparameters` - list giving the user input or default hyperparameters used
- `fitted` - posterior mean of the individuals' means
- `residuals` - posterior mean of the residuals
- If `algorithm = "IS"`, the following:
  - `proposal_draws` - draws from the importance sampling proposal distribution (i.e., multivariate t centered at the posterior mode with precision equal to the negative hessian, and degrees of freedom set to the user input `proposal_df`.
  - `importance_sampling_weights` - importance sampling weights that match the rows of the returned `proposal_draws`.
  - `effective_sample_size`
  - `mc_error`
- other inputs into `glm_b`

Importance sampling:

`glm_b` will, unless `use_importance_sampling = FALSE`, perform importance sampling. The proposal will use a multivariate t distribution, centered at the posterior mode, with the negative hessian as its precision matrix. Do NOT treat the proposal_draws as posterior draws.

Priors:

If the prior is set to be either "zellner" or "normal", a normal distribution will be used as the prior of β, specifically

$$\beta \sim N(\mu, V)$$

where μ is the prior_beta_mean and V is the prior_beta_precision (not covariance) matrix.

- `zellner`: `glm_b` sets $\mu = 0$ and $V = \frac{1}{g}X^\top X$.
- `normal`: If missing `prior_beta_mean`, `glm_b` sets $\mu = 0$, and if missing `prior_beta_precision` V will be a diagonal matrix. The first element, corresponding to the intercept, will be $(2.5 \times \max \tilde{s}_y, 1)^{-2}$, where $\tilde{s}_y$ is max of 1 and the standard deviation of y. Remaining diagonal elements will equal $(2.5s_y/s_x)^{-2}$, where s_x is the standard deviations of the covariates. This equates to being 95% certain a priori that a change in x by one standard deviation (s_x) would not lead to a change in the linear predictor of more than 5 standard deviations ($5s_y$). This imposes weak regularization that adapts to the scale of the data elements.

ROPE:

If missing, the ROPE bounds will be given under the principle of "half of a small effect size."

- Gaussian. Using Cohen's D of 0.2 as a small effect size, the ROPE is built under the principle that moving the full range of X (i.e., $\pm 2s_x$) will not move the mean of y by more than the overall mean of y minus $0.1s_y$ to the overall mean of y plus $0.1s_y$. The result is a ROPE equal to $|\beta_j| < 0.05s_y/s_j$. If the covariate is binary, then this is simply $|\beta_j| < 0.2s_y$.
- Poisson. FDA guidance suggests a small effect is a rate ratio less than 1.25. We use half this effect: 1.125, and consider ROPE to indicate that a moving the full range of X ($\pm 2s_x$ will not change the rate ratio by more than this amount. Thus the ROPE for the regression coefficient equals $|\beta| < \frac{\log(1.125)}{4s_x}$. For binary covariates, this is simply $|\beta| < \log(1.125)$.

References

Kruschke JK. Rejecting or Accepting Parameter Values in Bayesian Estimation. Advances in Methods and Practices in Psychological Science. 2018;1(2):270-280. doi:10.1177/2515245918771304

Tim Salimans. David A. Knowles. "Fixed-Form Variational Posterior Approximation through Stochastic Linear Regression." Bayesian Anal. 8 (4) 837 - 882, December 2013. https://doi.org/10.1214/13-BA858

Examples

```
# Generate some negative-binomial data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5],
             time = rexp(N))
test_data$outcome =
  rnbinom(N,
        mu = exp(-2 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e"))) * test_data$time,
          size = 0.7)

# Fit using variational Bayes (default)
fit_vb1 <-
  glm_b(outcome ~ x1 + x2 + x3 + offset(log(time)),
        data = test_data,
        family = negbinom(),
        seed = 2025)
fit_vb1
summary(fit_vb1,
        CI_level = 0.9)
plot(fit_vb1)
coef(fit_vb1)
credint(fit_vb1,
        CI_level = 0.99)
bayes_factors(fit_vb1,
              by = "v")
preds =
  predict(fit_vb1)

# Try different priors
fit_vb2 <-
  glm_b(outcome ~ x1 + x2 + x3 + offset(log(time)),
        data = test_data,
        family = negbinom(),
        seed = 2025,
        prior = "normal")
fit_vb2
fit_vb3 <-
  glm_b(outcome ~ x1 + x2 + x3 + offset(log(time)),
        data = test_data,
        family = negbinom(),
        seed = 2025,
        prior = "improper")
fit_vb3

# Use Importance sampling instead of VB
fit_is <-
  glm_b(outcome ~ x1 + x2 + x3 + offset(log(time)),
        data = test_data,
        family = negbinom(),
        algorithm = "IS",
        seed = 2025)
summary(fit_is)

# Use large sample approximation instead of VB
fit_lsa <-
  glm_b(outcome ~ x1 + x2 + x3 + offset(log(time)),
        data = test_data,
        family = negbinom(),
        algorithm = "LSA",
        seed = 2025)
summary(fit_lsa)
```

---

heteroscedasticity_test

Test for heteroscedasticity in AOV models

Description

Use Chib's method to compute the Bayes factor to test for heteroscedasticity in analysis of variance models.

Usage

```
heteroscedasticity_test(hetero_model, homo_model)
```

Arguments

`hetero_model`	aov_b object where the heteroscedastic argument has been set to TRUE (the default)
`homo_model`	aov_b object where the heteroscedastic argument has been set to FALSE

Value

(returned invisible) A tibble with Bayes factors and interpretations.

References

Kass, R. E., & Raftery, A. E. (1995). Bayes Factors. Journal of the American Statistical Association, 90(430), 773–795.

Examples

```
# Test homoscedastic case
## Generate some data
set.seed(2025)
N = 200
test_data =
  data.frame(x1 = rep(letters[1:5],N/5))
test_data$outcome =
  rnorm(N,-1 + 2 * (test_data$x1 %in% c("d","e")) )

## Fit the anova models
hetero_model =
  aov_b(outcome ~ x1,
        test_data)
homo_model =
  aov_b(outcome ~ x1,
        test_data,
        heteroscedastic = FALSE)

## Perform test for heteroscedasticity using Bayes factors
heteroscedasticity_test(hetero_model,
                        homo_model)

# Test heteroscedastic case
## Generate some data
set.seed(2025)
N = 200
test_data =
  data.frame(x1 = rep(letters[1:5],N/5))
test_data$outcome =
  rnorm(N,
        -1 + 2 * (test_data$x1 %in% c("d","e")),
        sd = 3 - 2 * (test_data$x1 %in% c("d","e")))

## Fit the anova models
hetero_model =
  aov_b(outcome ~ x1,
        test_data)
homo_model =
  aov_b(outcome ~ x1,
        test_data,
        heteroscedastic = FALSE)

## Perform test for heteroscedasticity using Bayes factors
heteroscedasticity_test(hetero_model,
                        homo_model)
```

`IC` *Compute AIC, BIC, DIC, or WAIC for aov_b or lm_b objects. (Lower is better.)*

Description

Compute AIC, BIC, DIC, or WAIC for aov_b or lm_b objects. (Lower is better.)

Usage

```
DIC(object, ...)

## S3 method for class 'lm_b'
BIC(object, ...)

## S3 method for class 'glm_b'
BIC(object, ...)

## S3 method for class 'aov_b'
BIC(object, ...)

## S3 method for class 'lm_b'
AIC(object, ...)

## S3 method for class 'glm_b'
AIC(object, ...)

## S3 method for class 'aov_b'
AIC(object, ...)

## S3 method for class 'lm_b'
DIC(object, seed = 1, mc_error = 0.01, ...)

## S3 method for class 'glm_b'
```

```
DIC(object, seed = 1, ...)

## S3 method for class 'aov_b'
DIC(object, ...)

## S3 method for class 'lm_b'
WAIC(object, seed = 1, ...)

## S3 method for class 'aov_b'
WAIC(object, ...)

## S3 method for class 'glm_b'
WAIC(object, seed = 1, ...)
```

Arguments

`object`	aov_b, lm_b, or glm_b object
`...`	Passed to methods.
`seed`	integer. Always set your seed!!!
`mc_error`	The number of posterior draws will ensure that with 99% probability the posterior mean of the deviance for DIC will be within $\pm$`mc_error`E(deviance).

Value

Numeric (or in the case of DIC, a numeric vector)

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
AIC(fit1)
BIC(fit1)
DIC(fit1)
WAIC(fit1)
```

lm_b *Bayesian Linear Models*

Description

lm_b is used to fit linear models. It can be used to carry out regression, single stratum analysis of variance and analysis of covariance (although aov_b may provide a more convenient interface for ANOVA.)

Usage

```
lm_b(
  formula,
  data,
  weights,
  prior = c("zellner", "conjugate", "improper")[1],
  zellner_g,
  prior_beta_mean,
  prior_beta_precision,
  prior_var_shape,
  prior_var_rate,
  ROPE,
  CI_level = 0.95
)
```

Arguments

`formula`	A formula specifying the model.
`data`	A data frame in which the variables specified in the formula will be found. If missing, the variables are searched for in the standard way. However, it is strongly recommended that you use this argument so that other generics for bayesics objects work correctly.
`weights`	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If non-NULL, it is assumed that the variance of y_i can be written as $Var(y_i) = \sigma^2/w_i$. While the estimands remain the same, the estimation is done by performing unweighted lm_b on $W^{\frac{1}{2}}y$ and $W^{\frac{1}{2}}X$, where W is the diagonal matrix of weights. Note that this then affects the zellner prior.
`prior`	character. One of "zellner", "conjugate", or "improper", giving the type of prior used on the regression coefficients.
`zellner_g`	numeric. Positive number giving the value of "g" in Zellner's g prior. Ignored unless prior = "zellner".
`prior_beta_mean`	numeric vector of same length as regression coefficients (denoted p). Unless otherwise specified, automatically set to rep(0,p). Ignored unless prior = "conjugate".

`prior_beta_precision`	pxp matrix giving a priori precision matrix to be scaled by the residual precision.
`prior_var_shape`	numeric. Twice the shape parameter for the inverse gamma prior on the residual variance(s). I.e., $\sigma^2 \sim IG$(prior_var_shape/2,prior_var_rate/2).
`prior_var_rate`	numeric. Twice the rate parameter for the inverse gamma prior on the residual variance(s). I.e., $\sigma^2 \sim IG$(prior_var_shape/2,prior_var_rate/2).
`ROPE`	vector of positive values giving ROPE boundaries for each regression coefficient. Optionally, you can not include a ROPE boundary for the intercept. If missing, defaults go to those suggested by Kruchke (2018).
`CI_level`	numeric. Credible interval level.

Details

MODEL:

The likelihood is given by

$$y_i \overset{ind}{\sim} N(x_i'\beta, \sigma^2).$$

The prior is given by

$$\beta|\sigma^2 \sim N\left(\mu, \sigma^2 V^{-1}\right) \sigma^2 \sim \Gamma^{-1}(a/2, b/2).$$

- For Zellner's g prior, $\frac{1}{g}X'X$.
- The default for the conjugate prior is based on arguments from standardized regression. The default V is set according to the following: "a priori, we are 95% certain that a standard deviation increase in X will not lead to more than a 5 standard deviation in the mean of y." If we then set the prior on the intercept to be flat, this leads to

  $$V^{1/2} = diag(1, s_{x_1}, \ldots, s_{x_p})/2.5,$$

  where s_y is the standard deviation of y, and s_{x_j} is the standard deviation of the j^{th} covariate.
- Unless `prior_var_shape` AND `prior_var_rate` are provided, the inverse gamma prior on the residual variance will place 50% prior probability that the correlation between the response and the fitted values is between 0.1 and 0.9.
- If `prior = "improper"`, then the prior is

  $$\pi(\beta, \sigma^2) \propto \frac{1}{\sigma^2}.$$

ROPE:

If missing, the ROPE bounds will be given under the principle of "half of a small effect size." Using Cohen's D of 0.2 as a small effect size, the ROPE is built under the principle that moving the full range of X (i.e., $\pm 2s_x$) will not move the mean of y by more than the overall mean of y minus $0.1s_y$ to the overall mean of y plus $0.1s_y$. The result is a ROPE equal to $|\beta_j| < 0.05s_y/s_j$. If the covariate is binary, then this is simply $|\beta_j| < 0.2s_y$.

Value

lm_b() returns an object of class "lm_b", which behaves as a list with the following elements:

- summary - tibble giving results for regression coefficients
- posterior_parameters - list giving the posterior parameters
- hyperparameters - list giving the user input (or default) hyperparameters used
- fitted - posterior mean of the individuals' means
- residuals - posterior mean of the residuals
- formula, data - input by user

Examples

```
# Generate some data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome =
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )

# Find good hyperparameters for the residual variance
s2_hyperparameters =
  find_invgamma_parms(lower_R2 = 0.05,
                      upper_R2 = 0.95,
                      probability = 0.8,
                      response_variance = var(test_data$outcome))
## Check (should equal ~ 0.8)
extraDistr::pinvgamma((1.0 - 0.05) * var(test_data$outcome),
                      0.5 * s2_hyperparameters[1],
                      0.5 * s2_hyperparameters[2]) -
  extraDistr::pinvgamma((1.0 - 0.95) * var(test_data$outcome),
                        0.5 * s2_hyperparameters[1],
                        0.5 * s2_hyperparameters[2])

# Fit the linear model using a conjugate prior
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data,
       prior = "conj",
       prior_var_shape = s2_hyperparameters["shape"],
       prior_var_rate = s2_hyperparameters["rate"])
fit1
summary(fit1,
        CI_level = 0.99)
plot(fit1)
coef(fit1)
credint(fit1,
        CI_level = 0.9)
```

```
bayes_factors(fit1)
bayes_factors(fit1,
              by = "var")
AIC(fit1)
BIC(fit1)
DIC(fit1)
WAIC(fit1)
vcov(fit1)
preds = predict(fit1)

# Try other priors
fit2 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data) # Default is prior = "zellner"
fit3 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data,
       prior = "improper")
```

`mediate_b` *Mediation using Bayesian methods*

Description

Mediation analysis done in the framework of Imai et al. (2010). Currently only applicable to linear models.

Usage

```
mediate_b(
  model_m,
  model_y,
  treat,
  control_value,
  treat_value,
  n_draws = 500,
  ask_before_full_sampling = TRUE,
  CI_level = 0.95,
  seed = 1,
  mc_error = ifelse("glm_b" %in% model_y, 0.01, 0.002),
  batch_size = 500
)
```

Arguments

`model_m`	a fitted model object of class lm_b for mediator.
`model_y`	a fitted model object of class lm_b for outcome.
`treat`	a character string indicating the name of the treatment variable used in the models. NOTE: Treatment variable must be numeric (even if it's 1's and 0's).
`control_value`	value of the treatment variable used as the control condition. Default is the 1st quintile of the treat variable.
`treat_value`	value of the treatment variable used as the treatment condition. Default is the 4th quintile of the treat variable.
`n_draws`	Number of preliminary posterior draws to assess final number of posterior draws required for accurate interval estimation
`ask_before_full_sampling`	logical. If FALSE, the user will not be asked if they want to complete the full sampling. Defaults to TRUE, as this can be a computationally intensive procedure.
`CI_level`	numeric. Credible interval level.
`seed`	integer. Always set your seed!!!
`mc_error`	positive scalar. The number of posterior samples will, with high probability, estimate the CI bounds up to $\pm$`mc_error`$\times$`sd(y)`.
`batch_size`	positive integer. Number of posterior draws to be taken at once. Higher values are more computationally intensive, but values which are too high might take up significant memory (allocates on the order of `batch_size`$\times$`nrow(model_y$data)`).

Details

The model is the same as that of Imai et al. (2010):

$$M_i(X) = w_i'\alpha_m + X\beta_m + \epsilon_{m,i}, y_i(X, M(\tilde{X})) = w_i'\alpha_y + X\beta_y + M(\tilde{X})\gamma + \epsilon_{y,i}, \epsilon_{m,i} \overset{iid}{\sim} N(0, \sigma_m^2), \epsilon_{y,i} \overset{iid}{\sim} N(0, \sigma_y^2),$$

where $M_i(X)$ is the mediator as a function of the treatment variable X, and w_i are confounder covariates.

Unlike the `mediation` R package, the estimation in `mediate_b` is fully Bayesian (as opposed to "quasi-Bayesian").

Value

A list with the following elements:

- `summary` - tibble giving results for causal mediation quantities
- `posterior_draws` (of counterfactual expectations)
- `mc_error` absolute error used, including any rescaling to match the scale of the outcome
- other inputs to `mediate_b`

References

Imai, Kosuke, et al. “A General Approach to Causal Mediation Analysis.” Psychological Methods, vol. 15, no. 4, 2010, pp. 309–34, https://doi.org/10.1037/a0020761.

Examples

```
# Simplest case
## Generate some data
set.seed(2025)
N = 500
test_data =
  data.frame(tr = rnorm(N),
             x1 = rnorm(N))
test_data$m =
  rnorm(N, 0.4 * test_data$tr - 0.25 * test_data$x1)
test_data$outcome =
  rnorm(N,-1 + 0.6 * test_data$tr + 1.5 * test_data$m + 0.25 * test_data$x1)

## Fit the mediator and outcome models
m1 =
  lm_b(m ~ tr + x1,
       data = test_data)
m2 =
  lm_b(outcome ~ m + tr + x1,
       data = test_data)
## Estimate the causal mediation quantities
m3 <-
  mediate_b(m1,m2,
            treat = "tr",
            control_value = -2,
            treat_value = 2,
            n_draws = 500,
            mc_error = 0.05,
            ask_before_full_sampling = FALSE)
m3
summary(m3,
        CI_level = 0.9)

# More complicated scenario
## Generate some data
set.seed(2025)
N = 500
test_data =
  data.frame(tr = rep(0:1,N/2),
             x1 = rnorm(N))
test_data$m =
  rnorm(N, 0.4 * test_data$tr - 0.25 * test_data$x1)
test_data$outcome =
  rpois(N,exp(-1 + 0.6 * test_data$tr + 1.5 * test_data$m + 0.25 * test_data$x1))

## Fit the mediator and outcome models
m1 =
  lm_b(m ~ tr + x1,
       data = test_data)
m2 =
  glm_b(outcome ~ m + tr + x1,
        data = test_data,
        family = poisson())

##  Estimate the causal mediation quantities
m3 <-
  mediate_b(m1,m2,
            treat = "tr",
            control_value = 0,
            treat_value = 1,
            n_draws = 500,
            mc_error = 0.05,
            ask_before_full_sampling = FALSE)
summary(m3)
```

negbinom *Negative-binomial family*

Description

The `negbinom()` is an additional family to be considered alongside others documented under `stats::family`.

Usage

```
negbinom()
```

Value

an object of class "family". See `stats::family`.

Examples

```
negbinom()
```

np_glm_b *Non-parametric linear models*

Description

np_glm_b uses general Bayesian inference with loss-likelihood bootstrap. This is, as implemented here, a Bayesian non-parametric linear models inferential engine. Applicable data types are continuous (use family = gaussian()), count (use family = poisson()), or binomial (use family = binomial()).

Usage

```
np_glm_b(
  formula,
  data,
  family,
  loss = "selfinformation",
  loss_gradient,
  trials,
  n_draws,
  ask_before_full_sampling = TRUE,
  CI_level = 0.95,
  ROPE,
  seed = 1,
  mc_error = 0.01
)
```

Arguments

`formula`	A formula specifying the model.
`data`	A data frame in which the variables specified in the formula will be found. If missing, the variables are searched for in the standard way. However, it is strongly recommended that you use this argument so that other generics for bayesics objects work correctly.
`family`	A description of the error distribution and link function to be used in the model. See ?glm for more information. Currently implemented families are `binomial()`, `poisson()`, `negbinom()`, and `gaussian()` (this last acts as a wrapper for
`loss`	Either "selfinformation", or a function that takes in two arguments, the first of which should be the vector of outcomes and the second should be the expected value of y; The outcome of the function should be the loss evaluated for each observation. By default, the self-information loss is used (i.e., the negative log-likelihood). Note: I really do mean the expected value of y, even for binomial (i.e., n*p). If `family = negbinom()`, then a user-supplied loss function should take three arguments: y, mu, and phi, where phi is the dispersion parameter (i.e., Var$(y) = \mu + \mu^2/\phi$).
`loss_gradient`	If loss is a user-defined function (as opposed to "selfinformation"), supplying the gradient to the loss will speed up the algorithm.

`trials`	Integer vector giving the number of trials for each observation if family = binomial().
`n_draws`	integer. Number of posterior draws to obtain. If left missing, the large sample approximation will be used.
`ask_before_full_sampling`	logical. If TRUE, the user will be asked to specify whether they wish to commit to getting the full number of posterior draws to obtain precise credible interval bounds. Defaults to TRUE because the bootstrap is computationally intensive. Also, parallelization via future::plan is highly recommended for full sample.
`CI_level`	numeric. Credible interval level.
`ROPE`	vector of positive values giving ROPE boundaries for each regression coefficient. Optionally, you can not include a ROPE boundary for the intercept. If missing, defaults go to those suggested by Kruchke (2018).
`seed`	integer. Always set your seed!!!
`mc_error`	If large sample approximation is not used, the number of posterior draws will ensure that with 99% probability the bounds of the credible intervals will be within $\pm$ `mc_error`.

Details

Consider a population parameter of interest defined in terms of minimizing a loss function ℓ wrt the population distribution:

$$\theta(F_y) := \underset{\theta \in \Theta}{\text{argmax}} \int \ell(\theta, y) dF_y$$

If we use a non-parametric Dirichlet process prior on the distribution of y, F_y, and let the concentration parameter go to zero, we have the Bayesian bootstrap applied to a general Bayesian updating framework dictated by the loss function.

By default, the loss function is the self-information loss, i.e., the negative log likelihood. This then resembles a typical `glm_b` implementation, but is more robust to model misspecification.

Value

np_glm_b() returns an object of class "np_glm_b", which behaves as a list with the following elements:

- summary - a tibble giving results for regression coefficients.

References

S P Lyddon, C C Holmes, S G Walker, General Bayesian updating and the loss-likelihood bootstrap, Biometrika, Volume 106, Issue 2, June 2019, Pages 465–478, https://doi.org/10.1093/biomet/asz006

Examples

```
# Generate some data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome =
  rbinom(N,1,1.0 / (1.0 + exp(-(-2 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) ))))

# Fit the GLM via the (non-parametric) loss-likelihood bootstrap.
fit1 <-
  np_glm_b(outcome ~ x1 + x2 + x3,
           data = test_data,
           family = binomial())
fit1
summary(fit1,
        CI_level = 0.99)
plot(fit1)
coef(fit1)
credint(fit1)
predict(fit1,
        newdata = fit1$data[1,])
vcov(fit1)
```

plot *Plots bayesics objects.*

Description

Plots bayesics objects.

Usage

```
## S3 method for class 'lm_b'
plot(
  x,
  type,
  variable,
  exemplar_covariates,
  combine_pred_cred = TRUE,
  variable_seq_length = 30,
  return_as_list = FALSE,
  CI_level = 0.95,
  PI_level = 0.95,
  backtransformation = function(x) {
     x
 },
  ...
)
```

```
## S3 method for class 'aov_b'
plot(
  x,
  type = c("diagnostics", "cred band", "pred band"),
  combine_pred_cred = TRUE,
  return_as_list = FALSE,
  CI_level = 0.95,
  PI_level = 0.95,
  ...
)

## S3 method for class 'lm_b_bma'
plot(
  x,
  type = c("diagnostics", "cred band", "pred band"),
  variable,
  exemplar_covariates,
  combine_pred_cred = TRUE,
  bayes_pvalues_quantiles = c(0.01, 1:19/20, 0.99),
  variable_seq_length = 30,
  return_as_list = FALSE,
  CI_level = 0.95,
  PI_level = 0.95,
  seed = 1,
  backtransformation = function(x) {
     x
 },
  ...
)

## S3 method for class 'glm_b'
plot(
  x,
  type,
  variable,
  exemplar_covariates,
  combine_pred_cred = TRUE,
  variable_seq_length = 30,
  return_as_list = FALSE,
  CI_level = 0.95,
  PI_level = 0.95,
  seed = 1,
  ...
)

## S3 method for class 'np_glm_b'
plot(
  x,
  type,
  variable,
  exemplar_covariates,
  variable_seq_length = 30,
  return_as_list = FALSE,
  CI_level = 0.95,
  seed = 1,
  backtransformation = function(x) {
     x
 },
  ...
)

## S3 method for class 'mediate_b'
plot(x, type, return_as_list = FALSE, ...)

## S3 method for class 'survfit_b'
plot(x, n_draws = 10000, seed = 1, CI_level = 0.95, ...)
```

Arguments

`x`	A bayesics object
`type`	character. Select any of "diagnostics" ("dx" is also allowed), "pdp" (partial dependence plot), "cred band", and/or "pred band". NOTE: the credible and prediction bands only work for numeric variables. If plotting a `mediate_b` object, the valid values for `type` are "diagnostics" (or "dx"), "acme", or "ade".
`variable`	character. If type = "pdp" , which variable should be plotted?
`exemplar_covariates`	data.frame or tibble with exactly one row. Used to fix other covariates while varying the variable of interest for the plot.
`combine_pred_cred`	logical. If type includes both "cred band" and "pred band", should the credible band be superimposed on the prediction band or plotted separately?
`variable_seq_length`	integer. Number of points used to draw pdp.
`return_as_list`	logical. If TRUE, a list of ggplots will be returned, rather than a single plot produced by the patchwork package.
`CI_level`	Posterior probability covered by credible interval
`PI_level`	Posterior probability covered by prediction interval
`backtransformation`	function. If a transformation of the response variable was used, `backtransformation` should be the inverse of this transformation function. E.g., if you fit lm_b(log(y) ~ x), then set `backtransformation=exp`.
`...`	optional arguments.

`bayes_pvalues_quantiles`	ADD description!
`seed`	ADD description!
`n_draws`	integer. Number of posterior draws used for visualization of survival curves. Ignored if `x` is not a `survfit_b` object.

Value

If `return_as_list=TRUE`, a list of requested ggplots.

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
plot(fit1)
```

`poisson_test_b` *Poisson tests*

Description

Make inference on one or two populations using Poisson distributed count data

Usage

```
poisson_test_b(
  x,
  offset,
  r,
  ROPE,
  prior = "jeffreys",
  prior_shape_rate,
  CI_level = 0.95,
  plot = TRUE,
  seed = 1,
  mc_error = 0.002
)
```

Arguments

`x`	Number of events. A vector of length one or two.
`offset`	Time, area, etc. measured in the Poisson process. NOTE: Do not take the log!
`r`	optional. If provided and inference is being made for a single population, `poisson_test_b` will return the posterior probability that the population rate is less than this value.
`ROPE`	ROPE for rate ratio if inference is being made for two populations. Provide either a single value or a vector of length two. If the former, the ROPE will be taken as (1/ROPE,ROPE). If the latter, these will be the bounds of the ROPE.
`prior`	Either "jeffreys" (Gamma(1/2,0)) or "flat" (Gamma(0.001,0.001)). This is ignored if prior_shape_rate is provided.
`prior_shape_rate`	Vector of length two, giving the shape and rate parameters for the gamma distribution that will act as the prior on the population rates.
`CI_level`	The posterior probability to be contained in the credible intervals.
`plot`	logical. Should a plot be shown?
`seed`	Always set your seed! (Unused for a single population rate)
`mc_error`	The number of posterior draws will ensure that with 99% probability the bounds of the credible intervals of λ_1/λ_2 will be within $\pm$ `mc_error`. (Ignored for a single population rate.)

Details

The likelihood is

$$y \sim Poi(\lambda t),$$

where λ is the rate, and t is the time or area observed and is given by the argument `offset`.

The prior is given by

$$\lambda \sim \Gamma(a, b),$$

where a and b are given by the argument `prior_shape_rate`. If `prior_shape_rate` is missing and `prior = "jeffreys"`, then a Jeffrey's prior will be used, i.e., $\Gamma(0.5, 0)$ (improper), while if `prior = "flat"`, $\Gamma(0.001, 0.001)$ will be used.

Value

(returned invisible) A list with the following:

- `x`, `offset`: data and offset(s)
- `posterior_mean`, `posterior_mean_pop1`, `posterior_mean_pop2`: posterior means of the Poisson rates
- `CI`, `CI_pop1`, `CI_pop2`: Credible interval bounds for the rates
- `CI_lambda1_over_lambda2`: Credible interval bounds for the rate ratio (rate of population 1 over the rate of population 2)
- `Pr_less_than_r`: (1 sample analysis only) If `r` was supplied, the posterior probability that the rate is less than `r`.

- Pr_rate_ratio_lt_one: (2 sample analysis only) Posterior probability that the rate ratio is less than 1
- Pr_rateratio_in_ROPE: (2 sample analysis only) Posterior probability that the rate ratio is in the ROPE (based on Pr_rate_ratio_lt_one)
- rate_plot: Posterior and prior plots for the rates
- posterior_parameters: Posterior parameters for rates for the gamma posterior distribution

Examples

```
# One sample
poisson_test_b(x = 12)
## You can compute the posterior probability that the rate is less than r
poisson_test_b(x = 12,
               r = 8)

# Two samples
poisson_test_b(x = c(12,20))

# Offsets can be included:
poisson_test_b(x = c(12,20),
               offset = c(10,9))

# Different priors can be used
poisson_test_b(x = c(12,20),
               prior = "flat")
poisson_test_b(x = c(12,20),
               prior_shape_rate = c(20,1.5))
```

---

predict.aov_b *Predict method for aov_b model fits*

Description

Predict method for aov_b model fits

Usage

```
## S3 method for class 'aov_b'
predict(object, CI_level = 0.95, PI_level = 0.95, ...)
```

Arguments

object	Object of class aov_b
CI_level	Posterior probability covered by credible interval
PI_level	Posterior probability covered by prediction interval
...	optional arguments.

Value

tibble with estimate (posterior mean), prediction intervals, and credible intervals for the mean.

Examples

```
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rep(letters[1:5],N/5))
test_data$outcome =
  rnorm(N,-1 + 2 * (test_data$x1 %in% c("d","e")) )

# Fit 1-way ANOVA model
fit1 <-
  aov_b(outcome ~ x1,
        test_data,
        prior_mean_mu = 2,
        prior_mean_nu = 0.5,
        prior_var_shape = 0.01,
        prior_var_rate = 0.01)
predict(fit1)
```

`predict.glm_b` *Predict method for glm_b model fits*

Description

Predict method for glm_b model fits

Usage

```
## S3 method for class 'glm_b'
predict(
  object,
  newdata,
  trials,
  CI_level = 0.95,
  PI_level = 0.95,
  seed = 1,
  n_draws = 5000,
  ...
)
```

Arguments

`object`	Object of class glm_b
`newdata`	An optional data.frame in which to look for variables with which to predict.
`trials`	Integer vector giving the number of trials for each observation if family = binomial().
`CI_level`	Posterior probability covered by credible interval
`PI_level`	Posterior probability covered by prediction interval
`seed`	integer. Always set your seed!!!
`n_draws`	integer. Number of posterior draws used for prediction
`...`	optional arguments.

Value

tibble with estimate (posterior mean), prediction intervals, and credible intervals for the mean.

Examples

```
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5],
             time = rexp(N))
test_data$outcome =
  rnbinom(N,
        mu = exp(-2 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e"))) * test_data$time,
          size = 0.7)

# Fit using variational Bayes (default)
fit_vb1 <-
  glm_b(outcome ~ x1 + x2 + x3 + offset(log(time)),
        data = test_data,
        family = negbinom(),
        seed = 2025)
# Predict
predict(fit_vb1)
```

`predict.lm_b` *Predict method for lm_b model fits*

Description

Predict method for lm_b model fits

Usage

```
## S3 method for class 'lm_b'
predict(object, newdata, CI_level = 0.95, PI_level = 0.95, n_draws = 0, ...)
```

Arguments

`object`	Object of class lm_b
`newdata`	An optional data.frame in which to look for variables with which to predict.
`CI_level`	Posterior probability covered by credible interval
`PI_level`	Posterior probability covered by prediction interval
`n_draws`	If desired, the number of posterior samples drawn.
`...`	optional arguments.

Value

tibble with estimate (posterior mean), prediction intervals, and credible intervals for the mean.

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
predict(fit1)
```

`predict.lm_b_bma` *Predict method for bma model fits*

Description

Predict method for bma model fits

Usage

```
## S3 method for class 'lm_b_bma'
predict(object, newdata, CI_level = 0.95, PI_level = 0.95, seed = 1, ...)
```

Arguments

`object`	Object of class bma
`newdata`	An optional data.frame in which to look for variables with which to predict.
`CI_level`	Posterior probability covered by credible interval
`PI_level`	Posterior probability covered by prediction interval
`seed`	integer. Always set your seed!!!
`...`	optional arguments.

Value

list.

- newdata tibble with estimate, prediction intervals, and credible intervals for the mean.
- posterior_draws
  - mean_of_ynew draws of $E(y)$, marginalizing out the model
  - posterior draws of ynew

Examples

```
# Create data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5],
             x4 = rnorm(N),
             x5 = rnorm(N),
             x6 = rnorm(N),
             x7 = rnorm(N),
             x8 = rnorm(N),
             x9 = rnorm(N),
             x10 = rnorm(N))
test_data$outcome =
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )

# Fit linear model using Bayesian model averaging
fit <-
  bma_inference(outcome ~ .,
                test_data,
                user.int = FALSE)
predict(fit)
```

predict.np_glm_b *Predict method for lm_b model fits*

Description

Predict method for lm_b model fits

Usage

```
## S3 method for class 'np_glm_b'
predict(object, newdata, trials, CI_level = 0.95, ...)
```

Arguments

`object`	Object of class lm_b
`newdata`	An optional data.frame in which to look for variables with which to predict.
`trials`	Integer vector giving the number of trials for each observation if family = binomial().
`CI_level`	numeric. Credible interval level.
`...`	optional arguments.

Value

tibble with estimate, prediction intervals, and credible intervals for the mean.

Examples

```
# Generate some data
set.seed(2025)
N = 500
test_data =
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome =
 rbinom(N,1,1.0 / (1.0 + exp(-(-2 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) ))))

# Fit the GLM via the (non-parametric) loss-likelihood bootstrap.
fit1 <-
  np_glm_b(outcome ~ x1 + x2 + x3,
           data = test_data,
           family = binomial())
predict(fit1)
```

print — *Print bayesics objects.*

Description

Print bayesics objects.

Usage

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

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

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

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

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

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

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

Arguments

`x`	an object used to select a method.
`...`	optional arguments.

Value

None

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
```

```
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
print(fit1)
```

`prop_test_b` *Bayesian test of Equal or Given Proportions*

Description

`prop_test_b` either makes inference on a single population proportion, or else compares two population proportions. `binom_test_b` is the same as `prop_test_b`.

Usage

```
prop_test_b(
  n_successes,
  n_failures,
  n_total,
  p,
  predict_for_n,
  ROPE,
  prior = "jeffreys",
  prior_shapes,
  CI_level = 0.95,
  PI_level = 0.95,
  plot = TRUE,
  seed = 1,
  mc_error = 0.002
)
```

Arguments

`n_successes`	integer/numeric vector of length 1 (for 1 population) or 2 (for 2 populations) providing the number of "successes"
`n_failures`	Similar to n_successes, but for failures. Only provide this OR n_total.
`n_total`	Similar to n_successes, but for total number of trials. Only provide this OR n_failures.
`p`	optional. If provided and inference is being made for a single population, `prop_test_b` will return the posterior probability that the population proportion is less than this value.
`predict_for_n`	Number in a future trial. If missing, `prop_test_b` will use the observed number of trials.

`ROPE`	ROPE for odds ratio if inference is being made for two populations. Provide either a single value or a vector of length two. If the former, the ROPE will be taken as (1/ROPE,ROPE). If the latter, these will be the bounds of the ROPE.
`prior`	Either "jeffreys" (Beta(1/2,1/2)) or "uniform" (Beta(1,1)). This is ignored if prior_shapes is provided.
`prior_shapes`	Vector of length two, giving the shape parameters for the beta distribution that will act as the prior on the population proportions.
`CI_level, PI_level`	The posterior probability to be contained in the credible and prediction intervals respectively.
`plot`	logical. Should a plot be shown?
`seed`	Always set your seed! (Unused for a single population proportion.)
`mc_error`	The number of posterior draws will ensure that with 99% probability the bounds of the credible intervals of $p_1 - p_2$ will be within ± `mc_error`. (Ignored for a single population proportion.)

Details

The likelihood is given by

$$y \sim \text{Binom}(n, p),$$

and the prior on p is

$$p \sim Beta(a, b),$$

where a and b are given by the argument `prior_shapes`. If `prior_shapes` is missing and `prior = "jeffreys"`, then a Jeffreys prior will be used ($Beta(1/2, 1/2)$), and if `prior = "uniform"`, then a uniform prior will be used ($Beta(1, 1)$).

Value

(returned invisible) A list with the following:

- `successes`, `failures`: Number of successes and failures
- `posterior_mean`, `posterior_mean_pop1`, `posterior_mean_pop2`: posterior means for the population proportion
- `CI`, `CI_pop1`, `CI_pop2`: Credible interval for the population proportion
- `Pr_oddsratio_in_ROPE`: (2 sample analysis only) Posterior probability that the odds ratio is in the ROPE
- `PI`, `PI_pop1`, `PI_pop2`: Prediction interval for the number of trials given in `predict_for_n`
- `Pr_less_than_p`: (1 sample analysis only) If `p` was supplied, the posterior probability that the population proportion is less than `p`
- `prop_plot`: Prior and posterior plot of population proportion(s)
- `posterior_parameters`: Posterior beta shape parameters for the population proportion(s)

Examples

```
# Single population
prop_test_b(14,
            19)
# or another way of the same thing;
prop_test_b(14,
            n_total = 14 + 19)

# A null value compared against can be added:
prop_test_b(14,
            19,
            p = 0.5)

# Two populations
prop_test_b(c(14,22),
            c(19,45))
# or equivalently
prop_test_b(c(14,22),
            n_total = c(14,22) + c(19,45))
```

---

sign_test_b *Paired sign test*

Description

Sign test for paired data.

Usage

```
sign_test_b(
  x,
  y,
  p0 = 0.5,
  prior = "jeffreys",
  prior_shapes,
  ROPE,
  CI_level = 0.95,
  plot = TRUE
)
```

Arguments

x	Either numeric vector or binary vector. If the former, $z_i = 1_{[x_i > y_i]}$ if y is supplied, else $z_i = 1_{[x_i > 0]}$. If the latter, then $z_i = x_i$.
y	Optional numeric vector to pair with x.

`p0`	`sign_test_b` will return the posterior probability that $p < p0$. Defaults to 0.5, as is most typical in the sign test.
`prior`	Either "jeffreys" (Beta(1/2,1/2)) or "uniform" (Beta(1,1)). This is ignored if prior_shapes is provided.
`prior_shapes`	Vector of length two, giving the shape parameters for the beta distribution that will act as the prior on the probability that $z_i = 1$.
`ROPE`	positive numeric of length 1 or 2. If of length 1, then ROPE is taken to be $p0\pm$ROPE. Defaults to ± 0.05.
`CI_level`	The posterior probability to be contained in the credible interval for p.
`plot`	logical. Should a plot be shown?

Details

The sign test looks at $z_i := 1_{[x_i > y_i]}$ rather than trying to model the distribution of (x_i, y_i). `sign_test_b` then uses the fact that

$$z_i \overset{iid}{\sim} Bernoulli(p)$$

and then makes inference on p. The prior on p is

$$p \sim Beta(a, b),$$

where a and b are given by the argument `prior_shapes`. If `prior_shapes` is missing and `prior = "jeffreys"`, then a Jeffreys prior will be used ($Beta(1/2, 1/2)$), and if `prior = "uniform"`, then a uniform prior will be used ($Beta(1, 1)$).

Value

(returned invisible) A list with the following:

- `posterior_mean`: Posterior mean of the median difference
- `CI`: Credible interval for the median difference
- `Pr_less_than_p`: Posterior probability that the proportion of differences that are positive is less than the argument `p0`.
- `ROPE_bounds`: ROPE bounds for the proportion of differences that are positive
- `ROPE`: Posterior probability that the proportion of differences which are positive falls in the ROPE
- `prop_plot`: Prior and posterior plot
- `posterior_parameters`: Posterior beta shape parameters for the proportion of differences which are positive

Examples

```
# Single population
sign_test_b(x = rnorm(50))

## Change ROPE
sign_test_b(x = rnorm(50),
            ROPE = 0.1)
```

```
## Change the prior
sign_test_b(x = rnorm(50),
            prior = "uniform")
sign_test_b(x = rnorm(50),
            prior_shapes = c(2,2))

## Two populations
sign_test_b(x = rnorm(50,1),
            y = rnorm(50,0))

## Change reference probability
sign_test_b(x = rnorm(50,1),
            y = rnorm(50,0),
            p0 = 0.7)
```

---

summary *Summary functions for bayesics objects*

Description

Summary functions for bayesics objects

Usage

```
## S3 method for class 'lm_b'
summary(object, CI_level = 0.95, ...)

## S3 method for class 'aov_b'
summary(object, CI_level = 0.95, ...)

## S3 method for class 'np_glm_b'
summary(object, CI_level = 0.95, interpretable_scale = TRUE, ...)

## S3 method for class 'lm_b_bma'
summary(object, CI_level = 0.95, ...)

## S3 method for class 'glm_b'
summary(object, CI_level = 0.95, interpretable_scale = TRUE, ...)

## S3 method for class 'mediate_b'
summary(object, CI_level = 0.95, ...)
```

Arguments

`object`	bayesics object

`CI_level`	Posterior probability covered by credible interval
`...`	optional arguments.
`interpretable_scale`	ADD description!

Value

tibble with summary values

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
summary(fit1)
```

`Surv` *Create a Survival Object*

Description

Create a survival object, usually used as a response variable in a model formula. Argument matching is special for this function, see Details under `Surv`. This is a restricted wrapper around `Surv` and currently supports only right-censored data.

Usage

```
Surv(...)
```

Arguments

`...`	arguments to be passed into `survival::Surv`. Currently, the input must be of the form `Surv(time,event)` for right censored data.

Value

An object of class "`Surv`".

References

Therneau T (2024). *A Package for Survival Analysis in R*. R package version 3.8-3, https://CRAN.R-project.org/package=survival.

Examples

```
set.seed(2025)
N = 300
test_data =
  data.frame(outcome =
               rweibull(N,2,5))
test_data$observed =
  ifelse(test_data$outcome >= 7, 0, 1)
test_data$outcome =
  ifelse(dplyr::near(test_data$observed,1), test_data$outcome, 7)
Surv(test_data$outcome,
     test_data$observed)
```

`survfit_b` *Create survival curves*

Description

Use the semi-parametric piecewise exponential survival model to fit a survival curve to one or more samples

Usage

```
survfit_b(formula, data, prior_shape, prior_rate, max_n_time_bins, n_time_bins)
```

Arguments

`formula`	Either `Surv(time,event)` ~ group for multiple groups, or else `Surv(time,event)` ~ 1 to make inference on a single population. The `event` variable must equal 1 if the event occurred and 0 if right censored. Currently right censoring is the only type of censoring allowed.
`data`	A data frame in which the variables specified in the formula will be found.
`prior_shape`	The shape parameter used in the gamma priors for the hazard rates
`prior_rate`	The rate parameter used in the gamma priors for the hazard rates
`max_n_time_bins`	integer. Maximum number of time bins, or "pieces", of the hazard function to be evaluated via Bayes factors. Ignored if `n_time_bins` is provided.
`n_time_bins`	Number of time bins used for hazard ratio. For a more data-driven approach, leave this argument missing and provide `max_n_time_bins`.

Details

The approach proposed by Qing et al. (2023) models the survival curve by way of piecewise exponential curves. That is, the hazard function is a piecewise function. The prior on the hazard within each "piece", or equivalently the rate of the exponential distribution, is a conjugate gamma distribution. Unless specified, the prior shape and rate for each piece is the posterior under the assumption that the data follow a single exponential distribution.

Unless prespecified by the user, the number of breaks in the hazard function is determined by Bayes factors, which can be quickly computed analytically.

If more than one population is being compared, then as before Bayes factors will be used to determine the number of breaks in each group's hazard function, and then Bayes factors will be used to compare the hypothesis that each group has a separate survival function vs. the null hypothesis that all groups share the same survival function.

Value

Object of class `survfit_b` with the following:

- `posterior_parameters` An `n_time_bins`x2 matrix whose columns provide shapes and rates of the gamma posterior distribution of each of the piecewise hazard rates.
- `intervals` An `n_time_bins`x2 matrix whose columns provide the start and endpoints of each time bin. If comparing multiple samples, a list of such matrices will be provided.
- `marginal_likelihood`
- `data`

If comparing multiple samples, each group will have a list of `posterior_parameters` and `intervals`.

References

Qing Y, Thall PF, Yuan Y. A Bayesian piecewise exponential phase II design for monitoring a time-to-event endpoint. Pharm Stat. 2023 Jan;22(1):34-44. doi: 10.1002/pst.2256. Epub 2022

Examples

```
# Single population
set.seed(2025)
N = 300
test_data =
  data.frame(outcome =
               rweibull(N,2,5))
test_data$observed =
  ifelse(test_data$outcome >= 7, 0, 1)
test_data$outcome =
  ifelse(dplyr::near(test_data$observed,1), test_data$outcome, 7)
fit1 =
  survfit_b(Surv(test_data$outcome,
                 test_data$observed) ~ 1)
fit1
plot(fit1)
```

```
# Multiple populations
set.seed(2025)
N = 300
test_data =
  data.frame(outcome =
               c(rweibull(2*N/3,2,5),
                 rweibull(N/3,2,10)),
             x1 = rep(letters[1:3],each = N/3))
test_data$observed =
  ifelse(test_data$outcome >= 9, 0, 1)
test_data$outcome =
  ifelse(dplyr::near(test_data$observed,1), test_data$outcome, 9)
fit2 =
  survfit_b(Surv(outcome,
                 observed) ~ x1,
            data = test_data)
fit2
plot(fit2)
```

---

t_test_b *t-test*

Description

One and two sample t-tests on vectors of data

Usage

```
t_test_b(
  x,
  y,
  mu,
  paired = FALSE,
  data,
  heteroscedastic = TRUE,
  prior_mean_mu,
  prior_mean_nu = 0.001,
  prior_var_shape = 0.001,
  prior_var_rate = 0.001,
  CI_level = 0.95,
  ROPE = 0.1,
  improper = FALSE,
  plot = TRUE,
  seed = 1,
  mc_error = 0.002
)
```

Arguments

`x`	Either a (non-empty) numeric vector of data values, or a formula of the form outcome ~ grouping variable.
`y`	an optional (non-empty) numeric vector of data values
`mu`	optional. If supplied, `t_test_b` will return the posterior probabilty that the population mean (ignored in 2 sample inference) is less than this value.
`paired`	logical. If TRUE, provide both x and y as vectors.
`data`	logical. Only used if x is a formula.
`heteroscedastic`	logical. Set to FALSE to assume all groups have equal variance.
`prior_mean_mu`	numeric. Hyperparameter for the a priori mean of the group means.
`prior_mean_nu`	numeric. Hyperparameter which scales the precision of the group means.
`prior_var_shape`	numeric. Twice the shape parameter for the inverse gamma prior on the residual variance(s). I.e., $\sigma^2 \sim IG$(prior_var_shape/2,prior_var_rate/2).
`prior_var_rate`	numeric. Twice the rate parameter for the inverse gamma prior on the residual variance(s). I.e., $\sigma^2 \sim IG$(prior_var_shape/2,prior_var_rate/2).
`CI_level`	numeric. Credible interval level.
`ROPE`	numeric. Used to compute posterior probability that Cohen's D +/- ROPE
`improper`	logical. Should we use an improper prior that is proportional to the inverse of the variance?
`plot`	logical. Should the resulting inverse gamma distribution be plotted?
`seed`	integer. Always set your seed!!!
`mc_error`	The number of posterior draws will ensure that with 99% probability the bounds of the credible intervals will be within $\pm$ `mc_error`$\times 4 s_y$, that is, within 100`mc_error`% of the trimmed range of y. (Ignored for single population inference.)

Details

A one and two sample t-test is nothing more than a special case of one-way anova. See `aov_b` for details.

Value

Either an aov_b object, if two samples are being compared, or a list with the following elements:

- Variable
- Post Mean
- Lower (bound of credible interval)
- Upper (bound of credible interval)
- Prob Dir (Probability of Direction)

Examples

```
# Single population
t_test_b(rnorm(50))
# or an alternative input format
t_test_b(outcome ~ 1,
         data = data.frame(outcome = rnorm(50)))

# Two populations
t_test_b(rnorm(50),
         rnorm(15,1))

# or an alternative input format
t_test_b(outcome ~ group_variable,
         data =
           data.frame(outcome = c(rnorm(50),
                                  rnorm(15,1)),
                      group_variable = rep(c("a","b"),
                                           c(50,15))))
```

vcov *Calculate Posterior Variance-Covariance Matrix for a Bayesian Fitted Model Object*

Description

Returns the posterior covariance matrix of the main parameters of a fitted `bayesics` object

Usage

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

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

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

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

Arguments

`object`	a fitted model object from `bayesics`.
`...`	Passed to methods.

Value

A matrix of the covariance matrix for the regression coefficients. If the posterior is a multivariate t distribution (or consists of independent t's in the case of heteroscedastic 1-way ANOVA), the degrees of freedom are returned as the `df` attribute of the matrix. Note that for `lm_b` and `aov_b` objects, this function already takes into account the uncertainty around the residual variance.

Examples

```
set.seed(2025)
N = 500
test_data <-
  data.frame(x1 = rnorm(N),
             x2 = rnorm(N),
             x3 = letters[1:5])
test_data$outcome <-
  rnorm(N,-1 + test_data$x1 + 2 * (test_data$x3 %in% c("d","e")) )
fit1 <-
  lm_b(outcome ~ x1 + x2 + x3,
       data = test_data)
vcov(fit1)
```

`wilcoxon_test_b` *Bayesian Wilcoxon Rank Sum (aka Mann-Whitney U) and Signed Rank Analyses*

Description

Bayesian Wilcoxon Rank Sum (aka Mann-Whitney U) and Signed Rank Analyses

Usage

```
wilcoxon_test_b(
  x,
  y,
  paired = FALSE,
  p = 0.5,
  ROPE,
  prior = "centered",
  prior_shapes,
  CI_level = 0.95,
  plot = TRUE,
  seed = 1
)
```

Arguments

`x`	numeric vector of data values. Non-finite (e.g., infinite or missing) values will be omitted.
`y`	an optional numeric vector of data values: as with x non-finite values will be omitted.
`paired`	if `TRUE` and `y` is supplied, x-y will be the input of the Bayesian Wilcoxon signed rank test.
`p`	numeric. • Signed rank: `wilcox_test_b` will return the posterior probability that the population proportion of positive values (i.e., $x > y$) is greater than this value. • Rank sum/Mann-Whitney U: `wilcox_test_b` will return the posterior probability that the Ω_x (see details) is greater than this value.
`ROPE`	If a single number, ROPE will be `p`$\pm$`ROPE`. If a vector of length 2, these will serve as the ROPE bounds. Defaults to ± 0.05.
`prior`	Prior used on the probability that x > y. Either "uniform" (Beta(1,1)), or "centered" (Beta(2,2)). This is ignored if prior_shapes is provided.
`prior_shapes`	Vector of length two, giving the shape parameters for the beta distribution that will act as the prior on the population proportions.
`CI_level`	The posterior probability to be contained in the credible interval.
`plot`	logical. Should a plot be shown?
`seed`	Always set your seed! (Unused for ≥ 20 observations.)

Details

Bayesian Wilcoxon signed rank analysis For a single input vector or paired data, the Bayesian signed rank analysis will be performed. The estimand is the proportion of (differenced) values that are positive. For more information, see dfba_wilcoxon and vignette("dfba_wilcoxon",package = "DFBA").

Bayesian Wilcoxon rank sum/Mann-Whitney analysis For unpaired x and y inputs, the Bayesian rank sum analysis will be performed. The estimand is $\Omega_x := \lim_{n\to\infty} \frac{U_x}{U_x+U_y}$, where U_x is the number of pairs (i,j) such that $x_i > y_j$, and vice versa for U_y. That is, it is the population proportion of all untied pairs for which $x > y$. Larger values imply that x is stochastically larger than y. For more information, see dfba_mann_whitney and vignette("dfba_mann_whitney",package = "DFBA").

Value

(returned invisible) If signed rank analysis is implemented, a list with the following:

- `posterior_mean`: Posterior mean of the proportion of differences that are positive
- `CI`: Credible interval of the proportion of differences that are positive
- `Pr_less_than_p`: Probability proportion of differences that are positive is less than the argument `p`

- `Pr_in_ROPE`: Probability proportion of differences that are positive is in the ROPE
- `prob_plot`: Prior and posterior plot of differences that are positive
- `posterior_parameters`: Posterior beta shape parameters for the proportion of differences that are positive
- `BF_for_phi_gr_onehalf_vs_phi_less_onehalf`: Bayes factor giving evidence in favor of the proportion of differences that are positive being greater than one half vs. less than one half
- `dfba_wilcoxon_object`: Underlying DFBA object

If rank sum analysis is implemented, a list with the following:

- `posterior_mean`: Posterior mean of Ω_x (see details)
- `CI`: Credible interval for Ω_x
- `Pr_less_than_p`: Posterior probability Ω_x is less than the argument `p`
- `Pr_in_ROPE`: Probability Ω_x is in the ROPE
- `prob_plot`: Prior and posterior plot of Ω_x
- `posterior_parameters`: Posterior beta shape parameters for Ω_x
- `BF_for_Omegax_gr_onehalf_vs_Omegax_less_onehalf`: Bayes factor in favor of Ω_x being greater than one half vs. less than one half
- `dfba_wilcoxon_object`: Underlying DFBA object

References

Chechile, R.A. (2020). Bayesian Statistics for Experimental Scientists: A General Introduction to Distribution-Free Methods. Cambridge: MIT Press.

Chechile, R. A. (2018) A Bayesian analysis for the Wilcoxon signed-rank statistic. Communications in Statistics - Theory and Methods, https://doi.org/10.1080/03610926.2017.1388402

Chechile, R.A. (2020). A Bayesian analysis for the Mann-Whitney statistic. Communications in Statistics – Theory and Methods 49(3): 670-696. https://doi.org/10.1080/03610926.2018.1549247.

Barch DH, Chechile RA (2023). DFBA: Distribution-Free Bayesian Analysis. doi:10.32614/CRAN.package.DFBA

Examples

```
# Signed rank analysis
## Generate data
N = 150
set.seed(2025)
test_data =
  data.frame(x = rbeta(N,2,10),
             y = rbeta(N,5,10))

## input differenced data
wilcoxon_test_b(test_data$x - test_data$y)
## input paired data vectors individually
wilcoxon_test_b(test_data$x,
                test_data$y,
                paired = TRUE)
```

```
## Use different priors
wilcoxon_test_b(test_data$x - test_data$y,
                prior = "uniform")
wilcoxon_test_b(test_data$x - test_data$y,
                prior_shapes = c(5,5))

## Change ROPE bounds
wilcoxon_test_b(test_data$x - test_data$y,
                ROPE = 0.1)

# Rank sum analysis
## Generate data
set.seed(2025)
N = 150
x = rbeta(N,2,10)
y = rbeta(N + 1,5,10)

## Perform analysis
wilcoxon_test_b(x,y)
```

Index

AIC *(IC)*, 28
aov_b, 3, *60*

bayes_factors, 6
BIC *(IC)*, 28
binom_test_b *(prop_test_b)*, 51
bma_inference, 8
bms, *9*

case_control_b, 10
chisq_test_b, 12
coef, 14
cor_test_b, 15
credint, 18

dfba_bivariate_concordance, *16*
dfba_mann_whitney, *63*
dfba_wilcoxon, *63*
DIC *(IC)*, 28

find_beta_parms, 19
find_invgamma_parms, 20

get_posterior_draws, 21
glm, *23*, *37*
glm_b, 22

heteroscedasticity_test, 26

IC, 28
independence_b *(chisq_test_b)*, 12

lm_b, 30

mediate_b, 33

negbinom, 36
np_glm_b, 37

plot, 39
poisson_test_b, 42
predict.aov_b, 44
predict.glm_b, 45
predict.lm_b, 46
predict.lm_b_bma, 47
predict.np_glm_b, 49
print, 50
prop_test_b, 51

sign_test_b, 53
summary, 55
Surv, *56*, 56
survfit_b, 57

t_test_b, 59

var_test_b *(heteroscedasticity_test)*, 26
vcov, 61

WAIC *(IC)*, 28
wilcoxon_test_b, 62