# Package 'bayesnec'

March 29, 2023

**Title** A Bayesian No-Effect- Concentration (NEC) Algorithm

**Version** 2.1.0.3

**Description** Implementation of No-Effect-Concentration estimation that uses 'brms' (see Burkner (2017)<doi:10.18637/jss.v080.i01>; Burkner (2018)<doi:10.32614/RJ-2018-017>; Carpenter 'et al.' (2017)<doi:10.18637/jss.v076.i01> to fit concentration(dose)-response data using Bayesian methods for the purpose of estimating 'ECX' values, but more particularly 'NEC' (see Fox (2010)<doi:10.1016/j.ecoenv.2009.09.012>. This package expands and supersedes an original version implemented in R2jags, see Fisher, Ricardo and Fox (2020)<doi:10.5281/ZENODO.3966864>.

**Depends** R (>= 4.1), brms, ggplot2

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**Biarch** true

**Imports** formula.tools, loo, extraDistr, dplyr, tidyr, purrr, tidyselect, evaluate, rlang, chk (>= 0.7.0)

**Suggests** rstan, knitr, rmarkdown, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**URL** https://open-aims.github.io/bayesnec/

**BugReports** https://github.com/open-aims/bayesnec/issues

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Rebecca Fisher [aut, cre], Diego Barneche [aut], Gerard Ricardo [aut], David Fox [aut]

**Maintainer** Rebecca Fisher <r.fisher@aims.gov.au>

**Repository** CRAN

**Date/Publication** 2023-03-29 13:50:02 UTC

# R topics documented:

bayesnec-package . . . 3
+.bnecfit . . . 3
amend . . . 4
autoplot . . . 5
average_endpoints . . . 7
bayesmanecfit-class . . . 9
bayesnecfit-class . . . 9
bayesnecformula . . . 10
beta_binomial2 . . . 12
beta_binomial2_lpmf . . . 13
beta_binomial2_rng . . . 13
bnec . . . 14
bnecfit-class . . . 17
bnec_newdata . . . 17
c.bnecfit . . . 18
check_chains . . . 19
check_formula . . . 20
check_priors . . . 21
compare_endpoints . . . 22
compare_fitted . . . 24
compare_posterior . . . 25
dispersion . . . 27
ecx . . . 28
expand_manec . . . 30
expand_nec . . . 31
fitted . . . 32
formula . . . 33
ggbnec_data . . . 34
herbicide . . . 35
is_manecsummary . . . 36
is_necsummary . . . 36
log_lik_beta_binomial2 . . . 37
make_brmsformula . . . 37
manecsummary-class . . . 38
manec_example . . . 39
model.frame . . . 39
models . . . 41
nec . . . 42
necsummary-class . . . 43
nec_data . . . 43
nsec . . . 44
plot . . . 45
posterior_epred . . . 47
posterior_epred_beta_binomial2 . . . 48
posterior_predict . . . 49
posterior_predict_beta_binomial2 . . . 50

prebayesnecfit-class . . . . . 50
predict . . . . . 51
print . . . . . 52
pull_brmsfit.bayesnecfit . . . . . 53
pull_out . . . . . 54
pull_prior . . . . . 55
rhat . . . . . 55
sample_priors . . . . . 56
show_params . . . . . 57
summary . . . . . 57
update.bnecfit . . . . . 58

**Index** **60**

---

`bayesnec-package` *The 'bayesnec' package.*

---

## Description

A No-Effect-Concentration (NEC) estimation package that uses brms (https://github.com/paul-buerkner/brms) to fit concentration (dose)-response data using Bayesian methods for the purpose of estimating both Effect Concentration (ECx) values, but more particularly NEC. Please see ?bnec for more details.

## References

Bürkner P-C (2018) Advanced Bayesian Multilevel Modeling with the R Package brms. The R Journal, 10: 395-411. doi:10.32614/RJ-2018-017.

---

`+.bnecfit` *"Add" multiple* `bnecfit` *objects into one single* `bayesmanecfit` *object containing Bayesian model averaging statistics.*

---

## Description

"Add" multiple `bnecfit` objects into one single `bayesmanecfit` object containing Bayesian model averaging statistics.

## Usage

```
## S3 method for class 'bnecfit'
e1 + e2
```

## Arguments

| | |
|---|---|
| `e1` | An object of class `bnecfit`. |
| `e2` | An object of class `bnecfit`. |

## Value

An object of class bayesmanecfit.

## Examples

```
## Not run:
library(bayesnec)
nec4param <- pull_out(manec_example, model = "nec4param")
ecx4param <- pull_out(manec_example, model = "ecx4param")
# Go from two bayesnecfit objects to a bayesmanecfit object.
#   In this example case it is redundant because it recovers the original
#   `manec_example`.
nec4param + ecx4param
# Add a bayesnecfit object to an existing bayesmanecfit object
nechorme4 <- nec_data |>
  dplyr::mutate(y = qlogis(y)) |>
  (\(.)bnec(formula = y ~ crf(x, model = "nechorme4"),
            data = ., iter = 200, warmup = 150, chains = 2,
            stan_model_args = list(save_dso = FALSE)))()
nechorme4 + manec_example

## End(Not run)
```

---

amend *Amends an existing* bayesmanecfit *object*

---

## Description

Amends an existing bayesmanecfit object, for example, by adding or removing fitted models.

## Usage

```
amend(
  object,
  drop,
  add,
  loo_controls,
  x_range = NA,
  precision = 1000,
  sig_val = 0.01,
  priors
)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesmanecfit, as returned by bnec. |

| | |
|---|---|
| drop | A character vector containing the names of model types you which to exclude for the modified fit. |
| add | A character vector containing the names of model types you which to include to the modified fit. |
| loo_controls | A named list of two elements ("fitting" and/or "weights"), each being a named list containing the desired arguments to be passed on to loo (via "fitting") or to loo_model_weights (via "weights"). If "fitting" is provided with argument pointwise = TRUE (due to memory issues) and family = "beta_binomial2", the bnec will fail because that is a custom family. If "weights" is not provided by the user, bnec will set the default method argument in loo_model_weights to "pseudobma". See ?loo_model_weights for further info. |
| x_range | A range of predictor values over which to consider extracting ECx. |
| precision | The length of the predictor vector used for posterior predictions, and over which to extract ECx values. Large values will be slower but more precise. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| priors | An object of class brmsprior which specifies user-desired prior distributions of model parameters. If missing, amend will figure out a baseline prior for each parameter. It can also be specified as a named list where each name needs to correspond to the same string as model. See Details. |

## Value

All successfully fitted bayesmanecfit model fits.

## Examples

```
library(bayesnec)
data(manec_example)
exmp <- amend(manec_example, drop = "nec4param")
```

---

# autoplot *bayesnec standard ggplot2 plotting method*

---

## Description

bayesnec standard **ggplot2** plotting method.

## Usage

```
## S3 method for class 'bayesnecfit'
autoplot(object, ..., nec = TRUE, ecx = FALSE, xform = identity)

## S3 method for class 'bayesmanecfit'
autoplot(
  object,
  ...,
  nec = TRUE,
  ecx = FALSE,
  xform = identity,
  all_models = FALSE,
  plot = TRUE,
  ask = TRUE,
  newpage = TRUE,
  multi_facet = TRUE
)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit. |
| ... | Additional arguments to be passed to ggbnec_data. |
| nec | Should NEC values be added to the plot? Defaults to TRUE. |
| ecx | Should ECx values be added to the plot? Defaults to FALSE.. |
| xform | A function to apply to the returned estimated concentration values. |
| all_models | Should all individual models be plotted separately\ (defaults to FALSE) or should model averaged predictions be plotted instead? |
| plot | Should output ggplot output be plotted? Only relevant if all = TRUE and multi_facet = FALSE. |
| ask | Indicates if the user is prompted before a new page is plotted. Only relevant if plot = TRUE and multi_facet = FALSE. |
| newpage | Indicates if the first set of plots should be plotted to a new page. Only relevant if plot = TRUE and multi_facet = FALSE. |
| multi_facet | Should all plots be plotted in one single panel via facets? Defaults to TRUE. |

## Value

A ggplot object.

## Examples

```
## Not run:
library(brms)
nec4param <- pull_out(manec_example, "nec4param")
autoplot(nec4param)
autoplot(nec4param, nec = FALSE)
```

```
autoplot(nec4param, ecx = TRUE, ecx_val = 50)

# plot model averaged predictions
autoplot(manec_example)
# plot all panels together
autoplot(manec_example, ecx = TRUE, ecx_val = 50, all_models = TRUE)

## End(Not run)
## Not run:
# plots multiple models, one at a time, with interactive prompt
autoplot(manec_example, ecx = TRUE, ecx_val = 50, all_models = TRUE,
         multi_facet = FALSE)

## End(Not run)
```

---

`average_endpoints` *average_endpoints*

---

## Description

Extracts posterior predicted endpoint values from a list of class `bayesnecfit` or `bayesmanecfit` model fits and calculates a geometric mean.

## Usage

```
average_endpoints(
  x,
  endpoint = "nec",
  ecx_val = 10,
  posterior = FALSE,
  type = "absolute",
  hormesis_def = "control",
  sig_val = 0.01,
  precision = 1000,
  x_range = NA,
  xform = identity,
  prob_vals = c(0.5, 0.025, 0.975)
)
```

## Arguments

| | |
|---|---|
| `x` | A named `list` of objects of class `bayesnecfit` or `bayesmanecfit` returned by `bnec`. |
| `endpoint` | The type of endpoint to use in the mean. Takes values "nec", "ecx" or "nsec". |
| `ecx_val` | The desired percentage effect value. This must be a value between 1 and 99 (for type = "relative" and "absolute"), defaults to 10. |
| `posterior` | A `logical` value indicating if the full posterior sample of calculated ECx values should be returned instead of just the median and 95 credible intervals. |

| | |
|---|---|
| type | A character vector, taking values of "relative", "absolute" (the default) or "direct". See Details. |
| hormesis_def | A character vector, taking values of "max" or "control". See Details. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values. against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| precision | The number of unique x values over which to find ECx – large values will make the ECx estimate more precise. |
| x_range | A range of x values over which to consider extracting ECx. |
| xform | A function to apply to the returned estimated concentration values. |
| prob_vals | A vector indicating the probability values over which to return the estimated ECx value. Defaults to 0.5 (median) and 0.025 and 0.975 (95 percent credible intervals). |

## Details

The geometric mean of values are simply the mean calculated on a log scale and back transformed through exp, although we have added the capacity to accommodate zero values. Note that the function assumes that x has been modelled on the natural scale. Often CR models are more stable on a log-transformed or sqrt scaling. If the input bayesnecfit or bayesmanecfit model fits are already based on a re-scaling of the x (concentration) axis, it is important to pass an appropriate xform argument to ensure these are back transformed before the the geometric mean calculation is applied.

## Value

The geometric mean of the endpoints estimate values of the bayesnecfit or bayesmanecfit model fits contained in x. See Details.

## See Also

bnec

## Examples

```
## Not run:
library(brms)
library(bayesnec)
data(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
ecx4param <- pull_out(manec_example, model = "ecx4param")
average_endpoints(list("nec" = ecx4param, "ecx" = nec4param), ecx_val = 50)

## End(Not run)
```

bayesmanecfit-class *Class* bayesmanecfit *of models fitted with the* **brms** *package*

## Description

Multiple models fitted with the bayesnec package are represented as a bayesmanecfit object, which contains the original brmsfit fitted objects, names of non-linear models that were fitted, model averaging WAIC stats, sample size, mean posterior NEC values, mean model averaged predictions on the data scale, model averaged residuals, full posterior distribution of predicated values, and summary statistics of NEC statistics.

## Details

See methods(class = "bayesmanecfit") for an overview of available methods.

## Slots

mod_fits A list of fitted model outputs of class prebayesnecfit for each of the fitted models.

success_models A character vector indicating the name of the successfully fitted models.

mod_stats A data.frame of model fit statistics.

sample_size The size of the posterior sample. Information on the priors used in the model.

w_nec_posterior The model-weighted posterior estimate of the NEC.

w_predicted_y The model-weighted predicted values for the observed data.

w_residuals Model-weighted residual values (i.e. observed - w_predicted_y).

w_pred_vals A list containing model-weighted posterior predicted values based on the supplied precision and x_range.

w_nec The summary stats (median and 95% credibility intervals) of w_nec_posterior.

## See Also

bayesnec, bnec, bayesnecfit

bayesnecfit-class *Class* bayesnecfit *of models fitted with the* **brms** *package*

## Description

Models fitted with the bayesnec package are represented as a bayesnecfit object, which contain the original brmsfit fitted object, list of initialisation values used, the validated bayesnecformula, name of non-linear model that was fitted, posterior predictions, posterior parameter estimates and a series of other statistics.

## Details

See `methods(class = "bayesnecfit")` for an overview of available methods.

## Slots

`fit` The fitted Bayesian model of class `brmsfit`.

`model` A `character` string indicating the name of the fitted model.

`init` A `list` containing the initialisation values for to fit the model.

`bayesnecformula` An object of class `bayesnecformula` and `formula`.

`pred_vals` A `list` containing a `data.frame` of summary posterior predicted values and a vector containing based on the supplied `precision` and `x_range`.

`top` The estimate for parameter "top" in the fitted model.

`beta` The estimate for parameter "beta" in the fitted model.

`nec` The estimated NEC.

`f` The estimate for parameter "f" in the fitted model, NA if absent for the fitted model type.

`bot` The estimate for parameter "bot" in the fitted model, NA if absent for the fitted model type.

`d` The estimate for parameter "d" in the fitted model, NA if absent for the fitted model type.

`slope` The estimate for parameter "slope" in the fitted model, NA if absent for the fitted model type.

`ec50` The estimate for parameter "ec50" in the fitted model, NA if absent for the fitted model type.

`dispersion` An estimate of dispersion.

`predicted_y` The predicted values for the observed data.

`residuals` Residual values of the observed data from the fitted model.

`nec_posterior` A full posterior estimate of the NEC.

## See Also

`bayesnec`, `bnec`, `bayesmanecfit`, `bayesnecformula`

---

# bayesnecformula *Set up a model formula for use in* `bayesnec`

---

## Description

Set up a model formula for use in the `bayesnec` package, allowing linear and non-linear (potentially multi-level) concentration-response models to be defined.

## Usage

```
bayesnecformula(formula, ...)
```

## Arguments

| | |
|---|---|
| formula | Either a character string defining an R formula or an actual formula object. See details. |
| ... | Unused. |

## Details

See methods(class = "bayesnecformula") for an overview of available methods.

### General formula syntax

The formula argument accepts formulas of the following syntax:

response | aterms ~ crf(x, model) + glterms

### The population-level term: crf

bayesnec uses a special internal term called crf, which sets the concentration-response equation to be evaluated based on some x predictor. The equation itself is defined by the argument "model": a character vector containing a specific model, a concatenation of specific models, or a single string defining a particular group of models (or group of equations, see models). Internally this argument is substituted by an actual brmsformula, which is then passed onto brm for model fitting.

### Group-level terms: glterms

The user has three options to define group-level effects in a bayesnecformula: 1) a general "offset" group-level effect defined by the term ogl (as in e.g. ogl(group_variable)). This adds an additional population-level parameter ogl to the model defined by crf, analogously to an intercept-only group-level effect in a classic linear model. 2) A group-level effect applied to all parameters in a model at once. This is done by the special term pgl, (as in e.g. pgl(group_variable)), which comes in handy so the user does not need to know the internal syntax and name of each parameter in the model. 3) A more classic approach where the user can specify which specific parameters — NB: that requires prior knowledge on the model structure and parameter names — to vary according to a grouping variable (as in e.g. (bot | group_variable)). bayesnecformula will ignore this term should the parameter not exist in the specified model or model suite. For example, the parameter bot exists in model "nec4param" but not in "nec3param", so if the user specifies model = "nec" in crf, the term (bot | group_variable) will be dropped in models where that parameter does not exist.

### Further brms terms (largely untested)

Currently bayesnecformula is quite agnostic about additional terms that are valid for a brmsformula. These are aterms and pterms (see ?brmsformula). The only capability that bayesnecformula does not allow is the addition of pterms outside of the term crf. Although pterms can be passed to predictor x within crf, we strongly discourage their use because those functionalities have not been tested yet. If this is extremely important to your work, please raise an issue on bayesnec GitHub, and we will consider further testing and development. Currently, the only two aterms that have validated behaviour are:

1. trials(), which is essential in binomially-distributed data, e.g. y | trials(trials_variable), and 2) weights, e.g. y | weights(weights_variable), following **brms** formula syntax. Please note that **brms** does not implement design weights as in other standard **base** functions. From their help page, **brms** "takes the weights literally, which means that an observation with weight 2 receives 2 times more weight than an observation with weight 1. It also means that

using a weight of 2 is equivalent to adding the corresponding observation twice to the data frame". Other aterms might be added, though we cannot attest to their functionality within bayesnec, i.e. checks will be done outside via brm.

**NB:** aterms other than trials() and weights() are currently omitted from model.frame output. If you need other aterms as part of that output please raise an issue on our GitHub page.

**Validation of formula** Please note that the function only checks for the input nature of the formula argument and adds a new class. This function **does not** perform any validation on the model nor checks on its adequacy to work with other functions in the package. For that please refer to the function check_formula which requires the dataset associated with the formula.

## Value

An object of class bayesnecformula and formula.

## See Also

check_formula, model.frame, models, show_params, make_brmsformula

## Examples

```
library(bayesnec)

bayesnecformula(y ~ crf(x, "nec3param"))
# or use shot alias bnf
bayesnecformula(y ~ crf(x, "nec3param")) == bnf(y ~ crf(x, "nec3param"))
bnf(y | trials(tr) ~ crf(sqrt(x), "nec3param"))
bnf(y | trials(tr) ~ crf(x, "nec3param") + ogl(group_1) + pgl(group_2))
bnf(y | trials(tr) ~ crf(x, "nec3param") + (nec + top | group_1))


# complex transformations are not advisable because
# they are passed directly to Stan via brms
# and are likely to fail -- transform your variable beforehand!
try(bnf(y | trials(tr) ~ crf(scale(x, scale = TRUE), "nec3param")))
```

---

# beta_binomial2 *Custom beta-binomial family*

---

## Description

Custom beta-binomial family

## Format

An object of class customfamily

`beta_binomial2_lpmf` *beta_binomial2_lpmf*

## Description

Beta-binomial wrapper LPMF

## Usage

```
beta_binomial2_lpmf(y, mu, phi, trials)
```

## Arguments

| | |
|---|---|
| `y` | vector of observation successes. |
| `mu` | posterior mu. |
| `phi` | posterior phi. |
| `trials` | vector of observation trials. |

## Value

A `numeric` value or vector containing the probability density of the beta binomial distribution

`beta_binomial2_rng` *beta_binomial2_rng*

## Description

Beta-binomial wrapper RNG

## Usage

```
beta_binomial2_rng(mu, phi, trials)
```

## Arguments

| | |
|---|---|
| `mu` | posterior mu. |
| `phi` | posterior phi. |
| `trials` | vector of observation trials. |

## Value

A `numeric` value or vector containing random values of the beta binomial distribution

bnec *bnec*

## Description

Fits a variety of NEC models using Bayesian analysis and provides a model averaged predictions based on WAIC model weights

## Usage

```
bnec(
  formula,
  data,
  x_range = NA,
  precision = 1000,
  sig_val = 0.01,
  loo_controls,
  x_var = NULL,
  y_var = NULL,
  trials_var = NULL,
  model = NULL,
  random = NULL,
  random_vars = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| formula | Either a character string defining an R formula or an actual formula object. See bayesnecformula and check_formula. |
| data | A data.frame containing the data to use with the formula. |
| x_range | A range of predictor values over which to consider extracting ECx. |
| precision | The length of the predictor vector used for posterior predictions, and over which to extract ECx values. Large values will be slower but more precise. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| loo_controls | A named list of two elements ("fitting" and/or "weights"), each being a named list containing the desired arguments to be passed on to loo (via "fitting") or to loo_model_weights (via "weights"). If "fitting" is provided with argument pointwise = TRUE (due to memory issues) and family = "beta_binomial2", the bnec will fail because that is a custom family. If "weights" is not provided by the user, bnec will set the default method argument in loo_model_weights to "pseudobma". See ?loo_model_weights for further info. |

| | |
|---|---|
| x_var | Removed in version 2.0. Use formula instead. Used to be a character indicating the column heading containing the predictor (concentration) variable. |
| y_var | Removed in version 2.0. Use formula instead. Used to be a character indicating the column heading containing the response variable. |
| trials_var | Removed in version 2.0. Use formula instead. Used to be a character indicating the column heading for the number of "trials" for binomial or beta_binomial2 response data, as it appears in "data" (if data is supplied). |
| model | Removed in version 2.0. Use formula instead. Used to be a character vector indicating the model(s) to fit. See Details for more information. |
| random | Removed in version 2.0. Use formula instead. Used to be a named list containing the random model formula to apply to model parameters. |
| random_vars | Removed in version 2.0. Use formula instead. Used to be a character vector containing the names of the columns containing the variables used in the random model formula. |
| ... | Further arguments to brm. |

## Details

### Overview

bnec serves as a wrapper for (currently) 23 (mostly) non-linear equations that are classically applied to concentration(dose)-response problems. The primary goal of these equations is to provide the user with estimates of No-Effect-Concentration (NEC), No-Significant-Effect-Concentration (NSEC), and Effect-Concentration (of specified percentage 'x', *ECx*) thresholds. These in turn are fitted through the brm function from package **brms** and therefore further arguments to brm are allowed. In the absence of those arguments, bnec makes its best attempt to calculate distribution family, priors and initialisation values for the user based on the characteristics of the data. Moreover, in the absence of user-specified values, bnec sets the number of iterations to 1e4 and warmup period to `floor(iterations / 5) * 4`. The chosen models can be extended by the addition of **brms** special "aterms" as well as group-level effects. See ?bayesnecformula for details.

### The available models/equations/formulas

The available equations (or models) can be found via the models function. Since version 2.0, bnec requires a specific formula structure which is fully explained in the help file of bayesnecformula. This formula incorporates the information regarding the chosen model(s). If one single model is specified, bnec will return an object of class bayesnecfit; otherwise if model is either a concatenation of multiple models and/or a string indicating a family of models, bnec will return an object of class bayesmanecfit, providing they were successfully fitted. The major difference being that the output of the latter includes Bayesian model averaging statistics. These classes come with multiple associated methods such as plot, autoplot, summary, print, model.frame and formula.

model may also be one of "all", meaning all of the available models will be fit; "ecx" meaning only models excluding a specific NEC step parameter will be fit; "nec" meaning only models with a specific NEC step parameter will be fit; "bot_free" meaning only models without a "bot" parameter (without a bottom plateau) will be fit; "zero_bounded" are models that are bounded to be zero; or "decline" excludes all hormesis models, i.e., only allows a strict decline in response across the whole predictor range. Notice that if one of these group strings is provided together with a user-specified named list for the brm's argument prior, the list names need to contain the actual model names, and not the group string , e.g. if `model = "ecx"` and `prior = my_priors` then `names(my_priors)` must

contain `models("ecx")`. To check available models and associated parameters for each group, use the function `models` or to check the parameters of a specific model use the function `show_params`.

All models provide an estimate for NEC. For model types with "nec" as a prefix, NEC is directly estimated as parameter "nec" in the model. Models with "ecx" as a prefix are continuous curve models, typically used for extracting ECx values from concentration response data. In this instance the NEC value is defined as the concentration at which there is a user supplied (see argument `sig_val`) percentage certainty (based on the Bayesian posterior estimate) that the response falls below the estimated value of the upper asymptote (top) of the response (i.e. the response value is significantly lower than that expected in the case of no exposure). The default value for `sig_val` is 0.01, which corresponds to an alpha value of 0.01 for a one-sided test of significance.

**Further argument to `brm`**

If not supplied via the `brm` argument `family`, the appropriate distribution will be guessed based on the characteristics of the input data. Guesses include: "bernoulli" / bernoulli / bernoulli(), "Beta" / Beta / Beta(), "binomial" / binomial / binomial(), "beta_binomial2" / beta_binomial2, "Gamma" / Gamma / Gamma(), "gaussian" / gaussian / gaussian(), "negbinomial" / negbinomial / negbinomial(), or "poisson" / poisson / poisson(). Note, however, that "negbinomial" and "betabinomimal2" require knowledge on whether the data is over-dispersed. As explained below in the Return section, the user can extract the dispersion parameter from a `bnec` call, and if they so wish, can refit the model using the "negbinomial" family.

Other families can be considered as required, please raise an issue on the GitHub development site if your required family is not currently available.

As a default, `bnec` sets the `brm` argument `sample_prior` to "yes" in order to sample draws from the priors in addition to the posterior distributions. Among others, these samples can be used to calculate Bayes factors for point hypotheses via `hypothesis`.

Model averaging is achieved through a weighted sample of each fitted models posterior predictions, with weights derived using functions `loo` and `loo_model_weights` from **brms**. Argument to both these functions can be passed via the `loo_controls` argument. Individual model fits can be pulled out for examination using function `pull_out`.

**Additional technical notes**

As some concentration-response data will use zero concentration which can cause numerical estimation issues, a small offset is added (1 / 10th of the next lowest value) to zero values of concentration where `x_var` are distributed on a continuous scale from 0 to infinity, or are bounded to 0, or 1.

**NAs are thrown away**

Stan's default behaviour is to fail when the input data contains NAs. For that reason **brms** excludes any NAs from input data prior to fitting, and does not allow them back in as is the case with e.g. `stats::lm` and `na.action = exclude`. So we advise that you exclude any NAs in your data prior to fitting because if you so wish that should facilitate merging predictions back onto your original dataset.

## Value

If argument model is a single string, then an object of class `bayesnecfit`; if many strings or a set, an object of class `bayesmanecfit`.

## See Also

bayesnecformula, check_formula, models, show_params

## Examples

```
## Not run:
library(bayesnec)
data(nec_data)

# A single model
exmp_a <- bnec(y ~ crf(x, model = "nec4param"), data = nec_data, chains = 2)
# Two models model
exmp_b <- bnec(y ~ crf(x, model = c("nec4param", "ecx4param")),
               data = nec_data, chains = 2)

## End(Not run)
```

---

# bnecfit-class *Class* bnecfit *of models fitted with function* bnec

---

## Description

This is a wrapper class which will be attached to both bayesnecfit and bayesmanecfit classes. Useful for methods which might need to take either class as an input simultaneously.

## Details

See methods(class = "bnecfit") for an overview of available methods.

## See Also

bayesnec, bnec, bayesnecfit, bayesmanecfit

---

# bnec_newdata *bnec_newdata*

---

## Description

Create a dataset for predictions

## Usage

```
bnec_newdata(x, precision = 100, x_range = NA)
```

### Arguments

| | |
|---|---|
| x | An object of class bayesnecfit or bayesmanecfit as returned by bnec. |
| precision | A numeric vector of length 1 indicating the number of x values over which to predict values. |
| x_range | A numeric vector of length 2 indicating the range of x values over which to make predictions. |

### Value

A data.frame to be used in predictions.

### Examples

```
## Not run:
library(bayesnec)
nec4param <- pull_out(manec_example, model = "nec4param")
# Make fine precision, predict out of range
newdata <- bnec_newdata(nec4param, precision = 200, x_range = c(0, 4))
nrow(newdata) == 200
all(range(newdata$x) == c(0, 4))
newdata2 <- bnec_newdata(manec_example) # default size
nrow(newdata2) == 100

## End(Not run)
```

---

## c.bnecfit *Concatenate multiple* bnecfit *objects into one single* bayesmanecfit *object containing Bayesian model averaging statistics.*

---

### Description

Concatenate multiple bnecfit objects into one single bayesmanecfit object containing Bayesian model averaging statistics.

### Usage

```
## S3 method for class 'bnecfit'
c(x, ...)
```

### Arguments

| | |
|---|---|
| x | An object of class bnecfit. |
| ... | Additional objects of class bnecfit. |

### Value

An object of class bayesmanecfit.

## Examples

```
## Not run:
library(bayesnec)
nec4param <- pull_out(manec_example, model = "nec4param")
ecx4param <- pull_out(manec_example, model = "ecx4param")
# Go from two bayesnecfit objects to a bayesmanecfit object.
#   In this example case it is redundant because it recovers the original
#   `manec_example`.
c(nec4param, ecx4param)
# Add a bayesnecfit object to an existing bayesmanecfit object
nechorme4 <- nec_data |>
  dplyr::mutate(y = qlogis(y)) |>
  (\(.)bnec(formula = y ~ crf(x, model = "nechorme4"),
            data = ., iter = 200, warmup = 150, chains = 2,
            stan_model_args = list(save_dso = FALSE)))()
c(nechorme4, manec_example)

## End(Not run)
```

---

# check_chains *Checking chain convergence*

---

## Description

Plots HMC chains for a bayesnecfit or bayesmanecfit model fit as returned by bnec.

## Usage

```
check_chains(x, ...)

## S3 method for class 'bayesnecfit'
check_chains(x, ...)

## S3 method for class 'bayesmanecfit'
check_chains(x, filename = NA, ...)
```

## Arguments

| | |
|---|---|
| x | An object of class bayesnecfit or bayesmanecfit as returned by bnec. |
| ... | Unused. |
| filename | An optional character vector to be used as a pdf filename in the case of a bayesmanecfit. Any non empty character string will indicate the user wants to save the plots. |

## Value

No return value, generates a plot or writes a pdf to file.

## Examples

```
## Not run:
library(bayesnec)
check_chains(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
check_chains(nec4param)

## End(Not run)
```

---

`check_formula` *Check if input model formula is appropriate to use with* `bayesnec`

---

## Description

Perform a series of checks to ensure that the input formula is valid for usage within `bayesnec`.

## Usage

```
check_formula(formula, data, run_par_checks = FALSE)
```

## Arguments

| | |
|---|---|
| `formula` | An object of class `bayesnecformula` as returned by function `bayesnecformula`. |
| `data` | A `data.frame` containing the variables specified in `formula`. |
| `run_par_checks` | See details. A `logical` defining whether random terms for specific parameters should be checked against the underlying concentration-response model defined in `formula`. Defaults to `FALSE`. |

## Details

This function allows the user to make sure that the input formula will allow for a successful model fit with the function `bnec`. Should all checks pass, the function returns the original formula. Otherwise it will fail and requires that the user fixes it until they're able to use it with `bnec`.

The argument `run_par_checks` is irrelevant for most usages of this package because it only applies if three conditions are met: 1) the user has specified a group-level effect; 2) the group-level effects is parameter specific (e.g. `(par | group_variable)` rather than `pgl/ogl(group_variable)`); and 3) The user is keen to learn if the specified parameter is found in the specified model (via argument `model` in the `crf` term – see details in ?bayesnecformula).

**NB:** `aterms` other than `trials()` and `weights()` are currently omitted from `model.frame` output. If you need other `aterms` as part of that output please raise an issue on our GitHub page. See details about `aterms` in ?bayesnecformula.

## Value

A validated object of class `bayesnecformula` and `formula`.

## See Also

bnec, bayesnecformula

## Examples

```
library(bayesnec)
nec3param <- function(beta, nec, top, x) {
  top * exp(-exp(beta) * (x - nec) *
    ifelse(x - nec < 0, 0, 1))
}

data <- data.frame(x = seq(1, 20, length.out = 10), tr = 100, wght = c(1, 2),
                   group_1 = sample(c("a", "b"), 10, replace = TRUE),
                   group_2 = sample(c("c", "d"), 10, replace = TRUE))
data$y <- nec3param(beta = -0.2, nec = 4, top = 100, data$x)

# returns error
# > f_1 <- y ~ crf(x, "nec3param") + z
# regular formula not allowed, wrap it with function bnf
# > check_formula(f_1, data)
# population-level covariates are not allowed
# > check_formula(bnf(f_1), data)


# expect a series of messages for because not all
# nec models have the "bot" parameter
f_2 <- y | trials(tr) ~ crf(x, "nec") + (nec + bot | group_1)
check_formula(bnf(f_2), data, run_par_checks = TRUE)

# runs fine
f_3 <- "y | trials(tr) ~ crf(sqrt(x), \"nec3param\")"
check_formula(bnf(f_3), data)
f_4 <- y | trials(tr) ~ crf(x, "nec3param") + ogl(group_1) + pgl(group_2)
inherits(check_formula(bnf(f_4), data), "bayesnecformula")
```

---

# check_priors *Plots the prior and posterior parameter probability densities from an object of class* bayesnecfit *or* bayesmanecfit.

---

## Description

Plots the prior and posterior parameter probability densities from an object of class bayesnecfit or bayesmanecfit.

## Usage

```
check_priors(object, filename = NA)
```

### Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit returned by bnec. |
| filename | An optional character vector to be used as a pdf filename in the case of a bayesmanecfit. Any non empty character string will indicate the user wants to save the plots. |

### Value

A plot of the prior and posterior parameter probability densities.

### See Also

bnec

### Examples

```
## Not run:
library(bayesnec)
data(manec_example)
check_priors(manec_example)

## End(Not run)
```

---

## compare_endpoints *compare_endpoints*

---

### Description

Extracts posterior predicted values from a list of class bayesnecfit or bayesmanecfit model fits and compares these via bootstrap re sampling.

### Usage

```
compare_endpoints(
  x,
  comparison = "nec",
  ecx_val = 10,
  type = "absolute",
  hormesis_def = "control",
  sig_val = 0.01,
  precision = 100,
  x_range = NA
)
```

## Arguments

| | |
|---|---|
| x | A named list of objects of class bayesnecfit or bayesmanecfit returned by bnec. |
| comparison | The posterior predictions to compare, takes values of "nec", "nsec", "ecx" or "fitted". |
| ecx_val | The desired percentage effect value. This must be a value between 1 and 99 (for type = "relative" and "absolute"), defaults to 10. |
| type | A character vector, taking values of "relative", "absolute" (the default) or "direct". See Details. |
| hormesis_def | A character vector, taking values of "max" or "control". See Details. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values. against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| precision | The number of unique x values over which to find ECx – large values will make the ECx estimate more precise. |
| x_range | A range of x values over which to consider extracting ECx. |

## Value

A named list containing bootstrapped differences in posterior predictions of the bayesnecfit or bayesmanecfit model fits contained in x. See Details.

## See Also

bnec

## Examples

```
## Not run:
library(bayesnec)
data(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
ecx4param <- pull_out(manec_example, model = "ecx4param")
compare_endpoints(list("nec" = ecx4param, "ecx" = nec4param), ecx_val = 50,
comparison="ecx")

## End(Not run)
```

compare_fitted *compare_fitted*

## Description

Extracts posterior predicted values from a list of class bayesnecfit or bayesmanecfit model fits and compares these across a vector of fitted values.

## Usage

```
compare_fitted(x, precision = 50, x_range = NA, make_newdata = TRUE, ...)
```

## Arguments

| | |
|---|---|
| x | A named list of objects of class bayesnecfit or bayesmanecfit returned by bnec. |
| precision | The number of unique x values over which to find ECx – large values will make the ECx estimate more precise. |
| x_range | A range of x values over which to consider extracting ECx. |
| make_newdata | Should the user allow the package to create newdata for predictions? If so, arguments precision and x_range will be used. Defaults to TRUE. See details. |
| ... | Further arguments that control posterior predictions via posterior_epred. |

## Details

The argument make_newdata is relevant to those who want the package to create a data.frame from which to make predictions. This is done via bnec_newdata and uses arguments precision and x_range. If make_newdata = FALSE and no additional newdata argument is provided (via ...), then the predictions are made for the raw data. Else, to generate predictions for a specific user-specific data.frame, set make_newdata = FALSE and provide an additional data.frame via the newdata argument. For guidance on how to structure newdata, see for example posterior_epred.

## Value

A named list containing bootstrapped differences in posterior predictions of the bayesnecfit or bayesmanecfit model fits contained in x. See Details.

## See Also

bnec

## Examples

```
## Not run:
library(bayesnec)
data(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
ecx4param <- pull_out(manec_example, model = "ecx4param")
compare_fitted(list("nec" = ecx4param, "ecx" = nec4param))

## End(Not run)
```

---

compare_posterior *compare_posterior*

---

## Description

Extracts posterior predicted values from a list of class bayesnecfit or bayesmanecfit model fits and compares these via bootstrap re sampling.

## Usage

```
compare_posterior(
  x,
  comparison = "nec",
  ecx_val = 10,
  type = "absolute",
  hormesis_def = "control",
  sig_val = 0.01,
  precision,
  x_range = NA,
  make_newdata = TRUE,
  ...
)
```

## Arguments

| | |
|---|---|
| x | A named list of objects of class bayesnecfit or bayesmanecfit returned by bnec. |
| comparison | The posterior predictions to compare, takes values of "nec", "nsec", "ecx" or "fitted". |
| ecx_val | The desired percentage effect value. This must be a value between 1 and 99 (for type = "relative" and "absolute"), defaults to 10. |
| type | A character vector, taking values of "relative", "absolute" (the default) or "direct". See Details. |
| hormesis_def | A character vector, taking values of "max" or "control". See Details. |

| | |
|---|---|
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values. against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| precision | The number of unique x values over which to find ECx – large values will make the ECx estimate more precise. |
| x_range | A range of x values over which to consider extracting ECx. |
| make_newdata | Only used if comparison = "fitted". Should the user allow the package to create newdata for predictions? If so, arguments precision and x_range will be used. Defaults to TRUE. See details. |
| ... | Further arguments that control posterior predictions via posterior_epred. |

## Details

type "relative" is calculated as the percentage decrease from the maximum predicted value of the response (top) to the minimum predicted value of the response. Type "absolute" (the default) is calculated as the percentage decrease from the maximum value of the response (top) to 0 (or bot for a 4 parameter model fit). Type "direct" provides a direct estimate of the x value for a given y. Note that for the current version, ECx for an "nechorme" (NEC Hormesis) model is estimated at a percent decline from the control.

For hormesis_def, if "max", then ECx or NSEC values – i.e., depending on argument comparison – are calculated as a decline from the maximum estimates (i.e. the peak at NEC); if "control", then ECx or NSEC values are calculated relative to the control, which is assumed to be the lowest observed concentration.

The argument make_newdata is only used if comparison = "fitted". It is relevant to those who want the package to create a data.frame from which to make predictions. This is done via bnec_newdata and uses arguments precision and x_range. If make_newdata = FALSE and no additional newdata argument is provided (via ...), then the predictions are made for the raw data. Else, to generate predictions for a specific user-specific data.frame, set make_newdata = FALSE and provide an additional data.frame via the newdata argument. For guidance on how to structure newdata, see for example posterior_epred.

## Value

A named list containing bootstrapped differences in posterior predictions of the bayesnecfit or bayesnecfit model fits contained in x. See Details.

## See Also

bnec ecx nsec nec bnec_newdata

## Examples

```
## Not run:
library(bayesnec)
data(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
ecx4param <- pull_out(manec_example, model = "ecx4param")
```

```
compare_posterior(list("nec" = ecx4param, "ecx" = nec4param), ecx_val = 50)

## End(Not run)
```

---

`dispersion` *Posterior dispersion*

---

## Description

Calculates a posterior dispersion metric.

## Usage

```
dispersion(model, summary = FALSE, seed = 10)
```

## Arguments

| | |
|---|---|
| `model` | An object of class `bayesnecfit` whose distribution family is either `poisson` or `binomial`. |
| `summary` | Logical. Should summary stats be returned instead of full vector? Defaults to FALSE. |
| `seed` | Change seed for reproducible purposes. |

## Details

This function calculates a dispersion metric which takes the ratio between the observed relative to simulated Pearson residuals sums of squares.

## Value

A `numeric` vector. If `summary` is FALSE, an n-long vector containing the dispersion metric, where n is the number of post warm-up posterior draws from the `brmsfit` object. If TRUE, then a `data.frame` containing the summary stats (mean, median, 95% highest density intervals) of the dispersion metric.

## References

Zuur, A. F., Hilbe, J. M., & Ieno, E. N. (2013). A Beginner's Guide to GLM and GLMM with R: A Frequentist and Bayesian Perspective for Ecologists. Highland Statistics Limited.

## Examples

```
## Not run:
library(bayesnec)
data(nec_data)
nec_data$y <- as.integer(round(nec_data$y * 100))
nec4param <- bnec(y ~ crf(x, "nec4param"), data = nec_data, chains = 2)
dispersion(nec4param, summary = TRUE)

## End(Not run)
```

---

ecx *Extracts the predicted ECx value*

---

## Description

Extracts the predicted ECx value as desired from an object of class bayesnecfit or bayesnecfit.

## Usage

```
ecx(
  object,
  ecx_val = 10,
  precision = 1000,
  posterior = FALSE,
  type = "absolute",
  hormesis_def = "control",
  x_range = NA,
  xform = identity,
  prob_vals = c(0.5, 0.025, 0.975)
)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit returned by bnec. |
| ecx_val | The desired percentage effect value. This must be a value between 1 and 99 (for type = "relative" and "absolute"), defaults to 10. |
| precision | The number of unique x values over which to find ECx – large values will make the ECx estimate more precise. |
| posterior | A logical value indicating if the full posterior sample of calculated ECx values should be returned instead of just the median and 95 credible intervals. |
| type | A character vector, taking values of "relative", "absolute" (the default) or "direct". See Details. |
| hormesis_def | A character vector, taking values of "max" or "control". See Details. |
| x_range | A range of x values over which to consider extracting ECx. |
| xform | A function to apply to the returned estimated concentration values. |

prob_vals — A vector indicating the probability values over which to return the estimated ECx value. Defaults to 0.5 (median) and 0.025 and 0.975 (95 percent credible intervals).

## Details

type "relative" is calculated as the percentage decrease from the maximum predicted value of the response (top) to the minimum predicted value of the response. Type "absolute" (the default) is calculated as the percentage decrease from the maximum value of the response (top) to 0 (or 'bot' for a four-parameter model fit). Type "direct" provides a direct estimate of the x value for a given y. Note that for the current version, ECx for an "nechorme" (NEC Hormesis) model is estimated at a percent decline from the control.

For hormesis_def, if "max", then ECx values are calculated as a decline from the maximum estimates (i.e. the peak at NEC); if "control", then ECx values are calculated relative to the control, which is assumed to be the lowest observed concentration.

Calls to functions ecx and nsec and compare_fitted do not require the same level of flexibility in the context of allowing argument newdata (from a posterior_predict perspective) to be supplied manually, as this is and should be handled within the function itself. The argument precision controls how precisely the ecx or nsec value is estimated, with argument x_range allowing estimation beyond the existing range of the observed data (otherwise the default range) which can be useful in a small number of cases. There is also no reasonable case where estimating these from the raw data would be of value, because both functions would simply return one of the treatment concentrations, making NOEC a better metric in that case.

## Value

A vector containing the estimated ECx value, including upper and lower 95% credible interval bounds.

## See Also

bnec

## Examples

```
library(brms)
library(bayesnec)
data(manec_example)
ecx(manec_example, ecx_val = 50)
ecx(manec_example)
```

expand_manec *Extracts a range of statistics from a list of* prebayesnecfit *objects.*

## Description

Extracts a range of statistics from a list of prebayesnecfit objects.

## Usage

```
expand_manec(
  object,
  formula,
  x_range = NA,
  precision = 1000,
  sig_val = 0.01,
  loo_controls
)
```

## Arguments

| | |
|---|---|
| object | A list of objects of class prebayesnecfit. |
| formula | Either a character string defining an R formula or an actual formula object. See bayesnecformula and check_formula. It could also be a list of formulas if multiple objects are passed to object. |
| x_range | A range of predictor values over which to consider extracting ECx. |
| precision | The length of the predictor vector used for posterior predictions, and over which to extract ECx values. Large values will be slower but more precise. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| loo_controls | A named list of two elements ("fitting" and/or "weights"), each being a named list containing the desired arguments to be passed on to loo (via "fitting") or to loo_model_weights (via "weights"). If "fitting" is provided with argument pointwise = TRUE (due to memory issues) and family = "beta_binomial2", the bnec will fail because that is a custom family. If "weights" is not provided by the user, bnec will set the default method argument in loo_model_weights to "pseudobma". See ?loo_model_weights for further info. |

## Value

A list of model statistical output derived from the input model list.

expand_nec *Extracts a range of statistics from a* prebayesnecfit *object.*

## Description

Extracts a range of statistics from a prebayesnecfit object.

## Usage

```
expand_nec(
  object,
  formula,
  x_range = NA,
  precision = 1000,
  sig_val = 0.01,
  loo_controls,
  ...
)
```

## Arguments

| | |
|---|---|
| object | An object of class prebayesnecfit. |
| formula | Either a character string defining an R formula or an actual formula object. See bayesnecformula and check_formula. |
| x_range | A range of predictor values over which to consider extracting ECx. |
| precision | The length of the predictor vector used for posterior predictions, and over which to extract ECx values. Large values will be slower but more precise. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| loo_controls | A named list of two elements ("fitting" and/or "weights"), each being a named list containing the desired arguments to be passed on to loo (via "fitting") or to loo_model_weights (via "weights"). If "fitting" is provided with argument pointwise = TRUE (due to memory issues) and family = "beta_binomial2", the bnec will fail because that is a custom family. If "weights" is not provided by the user, bnec will set the default method argument in loo_model_weights to "pseudobma". See ?loo_model_weights for further info. |
| ... | Further arguments to internal function. |

## Value

A list of model statistical output derived from the input model object.

fitted *Generates mean posterior linear predictions for objects fitted by* bnec

## Description

Generates mean posterior linear predictions for objects fitted by bnec. object should be of class bayesnecfit or bayesmanecfit.

## Usage

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

## S3 method for class 'bayesmanecfit'
fitted(object, summary = TRUE, robust = FALSE, probs = c(0.025, 0.975), ...)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit. |
| ... | Additional arguments to fitted.brmsfit if object is of class bayesnecfit, or to posterior_epred.brmsfit if object is of class bayesmanecfit. |
| summary | Should summary statistics be returned instead of the raw values? Default is TRUE. |
| robust | If FALSE (the default) the mean is used as the measure of central tendency and the standard deviation as the measure of variability. If TRUE, the median and the median absolute deviation (MAD) are applied instead. Only used if summary is TRUE. |
| probs | The percentiles to be computed by the quantile function. Only used if summary is TRUE. |

## Value

See ?brms:fitted.brmsfit.

## Examples

```
## Not run:
library(bayesnec)
# Uses default `precision` and `x_range` to generate `newdata` internally
fitted(manec_example)
# Provide user-specified `newdata`
nd_ <- data.frame(x = seq(0, 3, length.out = 200))
fits <- fitted(manec_example, ecx_val = 50, newdata = nd_,
               make_newdata = FALSE)
nrow(fits) == 200
# Predictions for raw input data
nec4param <- pull_out(manec_example, model = "nec4param")
```

```
fits <- fitted(nec4param, make_newdata = FALSE)
x <- pull_brmsfit(nec4param)$data$x
plot(x, fits[, 1])

## End(Not run)
```

---

`formula` *Retrieve formulas from models fitted by* `bnec`

---

## Description

Retrieve formulas from models fitted by `bnec`. `x` should be of class `bayesnecfit` or `bayesmanecfit`.

## Usage

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

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

## Arguments

| | |
|---|---|
| `x` | An object of class `bayesnecfit` or `bayesmanecfit`. |
| `...` | Unused. |
| `model` | A valid model string. |

## Value

An object of class `bayesnecformula`.

## Examples

```
library(bayesnec)
formula(manec_example, model = "nec4param")
nec4param <- pull_out(manec_example, "nec4param")
formula(nec4param)
```

ggbnec_data *Creates the data.frame for plotting with* autoplot.

## Description

Creates the data.frame for plotting with autoplot.

## Usage

```
ggbnec_data(x, add_nec = TRUE, add_ecx = FALSE, xform = identity, ...)
```

## Arguments

| | |
|---|---|
| x | An object of class bayesnecfit or bayesmanecfit, as returned by function bnec. |
| add_nec | Should NEC values be added to the plot? Defaults to TRUE. |
| add_ecx | Should ECx values be added to the plot? Defaults to FALSE. |
| xform | A function to apply to the returned estimated concentration values. |
| ... | Additional arguments to be passed to ecx. By default, function ecx returns EC10. |

## Value

A data.frame.

## Examples

```
library(bayesnec)
options(mc.cores = 2)
data(manec_example)

ggbnec_data(manec_example)
ggbnec_data(manec_example, add_ecx = TRUE, ecx_val = 50)
```

herbicide *Herbicide phytotoxicity data*

## Description

Herbicide phytotoxicity dataset from Jones & Kerswell (2003).

## Format

An object of class `data.frame` with 580 rows and 3 columns.

## Details

The response data (Fv/Fm) Chlorophyll fluorescence measurements of symbiotic dinoflagellates still in the host tissue of the coral (in hospite or in vivo) were measured using a DIVING-PAM chlorophyll fluorometer (Walz) on vertical planes of tissue 2 to 3 cm above the base of the corals, using either a 6 mm (Acropora formosa) or 2 mm (Seriatopora hystrix) fibre-optic probe. Parameters measured were the maximum potential quantum yield (Fv/Fm).

Additional information on each of the herbicides included is available from the original publication Jones & Kerswell (2003).

The columns are as follows:

**herbicide** The herbicide (chr).

**concentration** The treatment concentration in µg / L (dbl).

**fvfm** Maximum effective quantum yield (dbl).

## References

Jones RJ, Kerswell AP (2003) Phytotoxicity of Photosystem II (PSII) herbicides to coral. Marine Ecology Progress Series, 261: 149-159. doi: 10.3354/meps261149.

## Examples

```
head(herbicide)
```

## is_manecsummary *Checks if argument is a* manecsummary *object*

### Description

Checks if argument is a manecsummary object

### Usage

```
is_manecsummary(x)
```

### Arguments

| | |
|---|---|
| x | An R object |

### Value

A logical

## is_necsummary *Checks if argument is a* necsummary *object*

### Description

Checks if argument is a necsummary object

### Usage

```
is_necsummary(x)
```

### Arguments

| | |
|---|---|
| x | An R object |

## log_lik_beta_binomial2 *log_lik_beta_binomial2*

### Description

Beta-binomial wrapper LL

### Usage

```
log_lik_beta_binomial2(i, prep)
```

### Arguments

| | |
|---|---|
| i | observation i. |
| prep | data with posterior. |

### Value

Log likelihood of the beta binomial distribution

## make_brmsformula *Expose the final* brmsformula

### Description

Checks the input formula according to bayesnec requirements and expose the final brmsformula which is to be fitted via package **brms**.

### Usage

```
make_brmsformula(formula, data)
```

### Arguments

| | |
|---|---|
| formula | Either a character string defining an R formula or an actual formula object. See details. |
| data | A data.frame containing the variables specified in formula. |

### Value

A named list, with each element containing the final brmsformula to be passed to brm.

### See Also

bayesnecformula, check_formula

## Examples

```
library(bayesnec)
nec3param <- function(beta, nec, top, x) {
  top * exp(-exp(beta) * (x - nec) *
    ifelse(x - nec < 0, 0, 1))
}

data <- data.frame(x = seq(1, 20, length.out = 10), tr = 100, wght = c(1, 2),
                   group_1 = sample(c("a", "b"), 10, replace = TRUE),
                   group_2 = sample(c("c", "d"), 10, replace = TRUE))
data$y <- nec3param(beta = -0.2, nec = 4, top = 100, data$x)

# make one single model
f_1 <- "y | trials(tr) ~ crf(sqrt(x), \"nec3param\")"
make_brmsformula(f_1, data)
# make an entire class of models
f_2 <- y ~ crf(x, "ecx") + ogl(group_1) + pgl(group_2)
make_brmsformula(f_2, data)
```

---

# manecsummary-class *Class* manecsummary *of models fitted with the* **brms** *package*

---

## Description

Multiple models fitted with the bayesnec package are summarised as a manecsummary object, which contains the name of the non-linear models fitted, the family distribution used to fit all the models, the total post-warm-up sample size, a table containing the model weights, the method to calculate the weights, whether this model is an ECx-type model (see details below), and the ECx summary values should the user decide to calculate them.

## Details

See methods(class = "manecsummary") for an overview of available methods.

## Slots

models A character string indicating the name of the fitted non-linear models.

family A list indicating the family distribution and link function used to fit all the models.

sample_size The total post-warm-up sample size.

mod_weights A table containing the model weights.

mod_weights_method The method to calculate the weights.

ecx_mods A logical indicating which models are ECx-type models.

nec_vals The model-averaged NEC values. Note that if model stack contains ECx-type models, these will be via NSEC proxies.

ecs A list containing the ECx values should the user decide to calculate them (see the non-exported bayesnec:::summary.bayesnecfit help file for details). Different from the single-model case of class bayesnecfit, these ECx estimates will be based on the model weights.

rhat_issues A list detailing whether each fitted model exhibited convergence issues based on the Rhat evaluation.

## See Also

bayesnec, bnec, bayesnecfit, bayesmanecfit, necsummary

---

# manec_example *Example bayesmanecfit object*

---

## Description

Example bayesmanecfit object

## Format

An object of class bayesmanecfit. This was created to reduce run time in examples and tests, and to give the user an example to toy with. This was fitted to bayesnec built-in mock dataset (see ?nec_data), using models "nec4param" and "ecx4param". The number of chains were set to 2 and number of iterations were 50 only to make sure that package size was below 5 Mb. See help files for function bnec and class bayesmanecfit for details.

## Source

Code used to generate these models can be downloaded from https://github.com/open-AIMS/bayesnec/blob/master/data-raw/manec_example.R

---

# model.frame *Model.frame methods in bayesnec.*

---

## Description

Retrieve data.frame used to fit models via bnec, or directly from a bayesnecformula. formula should be of class bayesnecfit, bayesmanecfit or bayesnecformula.

## Usage

```
## S3 method for class 'bayesnecfit'
model.frame(formula, ...)

## S3 method for class 'bayesmanecfit'
model.frame(formula, model, ...)

## S3 method for class 'bayesnecformula'
model.frame(formula, data, ...)
```

## Arguments

| | |
|---|---|
| formula | An model object of class bayesnecfit, bayesmanecfit, or a formula of class bayesnecformula. |
| ... | Unused if formula is a bayesnecfit or a bayesmanecfit. Else, if formula is a bayesnecformula, additional arguments to be passed to check_formula. |
| model | A valid model string. |
| data | A data.frame containing the variables specified in formula. |

## Details

If formula is a bayesnecformula and it contains transformations to variables x and y, these are evaluated and returned as part of the data.frame.

## Value

If formula is a bayesnecfit or a bayesmanecfit, a data.frame containing the data used to fit the model.

If, instead, formula is a bayesnecformula, a data.frame with additional attributes detailing the population-level variables (attribute "bnec_pop") (response y, predictor x, and, if binomial a formula, trials) and, if applicable, the group-level variables (attribute "bnec_group").

## Examples

```
library(bayesnec)
# if input is of class `bayesnecfit` or `bayesmanecfit`
model.frame(manec_example, model = "nec4param")
nec4param <- pull_out(manec_example, "nec4param")
model.frame(nec4param)
# if input is of class `bayesnecformula`
nec3param <- function(beta, nec, top, x) {
  top * exp(-exp(beta) * (x - nec) *
    ifelse(x - nec < 0, 0, 1))
}

data <- data.frame(x = seq(1, 20, length.out = 10), tr = 100, wght = c(1, 2),
                   group_1 = sample(c("a", "b"), 10, replace = TRUE),
                   group_2 = sample(c("c", "d"), 10, replace = TRUE))
data$y <- nec3param(beta = -0.2, nec = 4, top = 100, data$x)

f_1 <- y ~ crf(x, "nec3param")
f_2 <- "y | trials(tr) ~ crf(sqrt(x), \"nec3param\")"
f_3 <- y | trials(tr) ~ crf(x, "nec3param") + ogl(group_1) + pgl(group_2)
f_4 <- y | trials(tr) ~ crf(x, "nec3param") + (nec + top | group_1)

m_1 <- model.frame(bnf(f_1), data)
attr(m_1, "bnec_pop")
model.frame(bnf(f_2), data)
m_3 <- model.frame(bnf(f_3), data)
attr(m_3, "bnec_group")
```

```
model.frame(bnf(f_4), data)
```

---

`models` *models*

---

## Description

Lists the fitted or available models.

## Usage

```
models(object)
```

## Arguments

| | |
|---|---|
| `object` | An object of class `bayesnecfit` or `bayesmanecfit` as returned by `bnec`, a `character` vector indicating the type of model set for which to list the available models, or a `numeric` vector indicating the natural range of values which the models should be able to handle (see Details). If missing, all available models and their groups are listed. |

## Details

The available models are "nec3param", "nec4param", "nechorme", "nechorme4", "necsigm", "neclin", "neclinhorme", "nechormepwr", "nechorme4pwr", "nechormepwr01", "ecxlin", "ecxexp", "ecxsigm", "ecx4param", "ecxwb1", "ecxwb2", "ecxwb1p3", "ecxwb2p3", "ecxll5", "ecxll4", "ecxll3", "ecxhormebc4", and "ecxhormebc5".

To see the model formula and parameters for a specific model use the function `show_params`.

To see all the models in an available set (e.g. "all", "nec" or ecx") use the function `models` specifying the group name.

To see the model names, model formula and parameters fitted in an existing `bayesnecfit` or `bayesmanecfit` model object use the function `models` specifying the fitted object.

To see what models are available for a given type of data use the function `models` passing a `numeric` vector indicating the range of possible data types. Models that have an exponential decay (most models with parameter "beta") with no "bot" parameter are zero-bounded and are not suitable for the Gaussian family, or any family modelled using a logit or log link function. Models with a linear decay (containing the string "lin" in their name) are not suitable for modelling families that are zero bounded (Gamma, Poisson, Negative binomial) using an identity link. Models with a linear decay or hormesis linear increase (all models with parameter "slope") are not suitable for modelling families that are 0, 1 bounded (binomial, beta, betabinomial2). These restrictions do not need to be controlled by the user and a call to `bnec` with `models = "all"` will simply exclude inappropriate models.

## Value

A `list` of the available or fitted models.

## Examples

```
library(bayesnec)
# default to all models and model groups
models()
# single model
show_params("nec3param")
# group of models
models("all")
# models that are suitable for 0,1 bounded data
models(c(0,1))
```

---

nec *Extracts the predicted NEC value as desired from an object of class* bayesnecfit *or* bayesmanecfit.

---

## Description

Extracts the predicted NEC value as desired from an object of class bayesnecfit or bayesmanecfit.

## Usage

```
nec(
  object,
  posterior = FALSE,
  xform = identity,
  prob_vals = c(0.5, 0.025, 0.975)
)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit returned by bnec. |
| posterior | A logical value indicating if the full posterior sample of calculated NEC values should be returned instead of just the median and 95% credible intervals. |
| xform | A function to apply to the returned estimated concentration values. |
| prob_vals | A vector indicating the probability values over which to return the estimated NEC value. Defaults to 0.5 (median) and 0.025 and 0.975 (95 percent credible intervals). |

## Value

A vector containing the estimated NEC value, including upper and lower 95% credible interval bounds (or other interval as specified by prob_vals).

## See Also

bnec

### Examples

```
library(bayesnec)
data(manec_example)
nec(manec_example)
```

---

necsummary-class *Class* necsummary *of models fitted with the* **brms** *package*

---

### Description

Single models fitted with the bayesnec package are summarised as a necsummary object, which contains the original brmsfit object summary, the name of the non-linear model fitted, whether this model is an ECx-type model (see details below), and the ECx summary values should the user decide to calculate them.

### Details

See methods(class = "necsummary") for an overview of available methods.

### Slots

brmssummary The standard summary for the fitted Bayesian model of class brmsfit.

model A character string indicating the name of the fitted non-linear model.

is_ecx A logical indicating whether model is an ECx-type model.

ecs A list containing the ECx values should the user decide to calculate them (see the non-exported bayesnec:::summary.bayesnecfit help file for details).

### See Also

bayesnec, bnec, bayesnecfit, bayesmanecfit, manecsummary

---

nec_data *Example data of non-linear decay*

---

### Description

A simulated dataset containing a series of response measurements as a function of a concentration axis. Data simulated by Diego Barneche.

### Format

A data frame with 100 rows and 2 variables:

- x: Concentration (predictor) axis.
- y: Response.

nsec *Extracts the predicted NSEC value as desired from an object of class* bayesnecfit *or* bayesmanecfit.

## Description

Extracts the predicted NSEC value as desired from an object of class bayesnecfit or bayesmanecfit.

## Usage

```
nsec(
  object,
  sig_val = 0.01,
  precision = 1000,
  posterior = FALSE,
  x_range = NA,
  hormesis_def = "control",
  xform = identity,
  prob_vals = c(0.5, 0.025, 0.975)
)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit returned by bnec. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values. against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| precision | The number of unique x values over which to find NSEC - large values will make the NSEC estimate more precise. |
| posterior | A logical value indicating if the full posterior sample of calculated NSEC values should be returned instead of just the median and 95 credible intervals. |
| x_range | A range of x values over which to consider extracting NSEC. |
| hormesis_def | A character vector, taking values of "max" or "control". See Details. |
| xform | A function to apply to the returned estimated concentration values. |
| prob_vals | A vector indicating the probability values over which to return the estimated NSEC value. Defaults to 0.5 (median) and 0.025 and 0.975 (95 percent credible intervals). |

## Details

For hormesis_def, if "max", then NSEC values are calculated as a decline from the maximum estimates (i.e. the peak at NEC); if "control", then ECx values are calculated relative to the control, which is assumed to be the lowest observed concentration.

Calls to functions ecx and nsec and compare_fitted do not require the same level of flexibility in the context of allowing argument newdata (from a posterior_predict perspective) to be supplied manually, as this is and should be handled within the function itself. The argument precision controls how precisely the ecx or nsec value is estimated, with argument x_range allowing estimation beyond the existing range of the observed data (otherwise the default range) which can be useful in a small number of cases. There is also no reasonable case where estimating these from the raw data would be of value, because both functions would simply return one of the treatment concentrations, making NOEC a better metric in that case.

## Value

A vector containing the estimated NSEC value, including upper and lower 95% credible interval bounds.

## See Also

bnec

## Examples

```
library(bayesnec)

data(manec_example)
nsec(manec_example)
```

---

plot *Generates a plot for objects fitted by* bnec

---

## Description

Generates a plot for objects fitted by bnec. x should be of class bayesnecfit or bayesmanecfit.

## Usage

```
## S3 method for class 'bayesnecfit'
plot(
  x,
  ...,
  CI = TRUE,
  add_nec = TRUE,
  position_legend = "topright",
  add_ec10 = FALSE,
  xform = identity,
  lxform = identity,
  jitter_x = FALSE,
```

```
  jitter_y = FALSE,
  ylab = "Response",
  xlab = "Predictor",
  xticks = NA
)

## S3 method for class 'bayesmanecfit'
plot(
  x,
  ...,
  CI = TRUE,
  add_nec = TRUE,
  position_legend = "topright",
  add_ec10 = FALSE,
  xform = identity,
  lxform = identity,
  jitter_x = FALSE,
  jitter_y = FALSE,
  ylab = "Response",
  xlab = "Predictor",
  xticks = NA,
  all_models = FALSE
)
```

## Arguments

| | |
|---|---|
| x | An object of class bayesnecfit or bayesmanecfit. |
| ... | Additional arguments to plot. |
| CI | A logical value indicating if credibility intervals on the model fit should be plotted, calculated as the upper and lower bounds of the individual predicted values from all posterior samples. |
| add_nec | A logical value indicating if the estimated NEC value and 95% credible intervals should be added to the plot. |
| position_legend | A numeric vector indicating the location of the NEC or EC10 legend, as per a call to legend. |
| add_ec10 | A logical value indicating if an estimated EC10 value and 95% credible intervals should be added to the plot. |
| xform | A function to be applied as a transformation of the x data. |
| lxform | A function to be applied as a transformation only to axis labels and the annotated NEC / EC10 values. |
| jitter_x | A logical value indicating if the x data points on the plot should be jittered. |
| jitter_y | A logical value indicating if the y data points on the plot should be jittered. |
| ylab | A character vector to use for the y-axis label. |
| xlab | A character vector to use for the x-axis label. |

| | |
|---|---|
| xticks | A numeric vector indicate where to place the tick marks of the x-axis. |
| all_models | A logical value indicating if all models in the model set should be plotted simultaneously, or if a model average plot should be returned. |

## Value

A plot of the fitted model.

## Examples

```
library(bayesnec)
nec4param <- pull_out(manec_example, "nec4param")
# plot single models (bayesnecfit)
plot(nec4param)
plot(nec4param, add_nec = FALSE)
plot(nec4param, add_ec10 = TRUE)

# plot model averaged predictions (bayesmanecfit)
plot(manec_example)
# plot all panels together
plot(manec_example, add_ec10 = TRUE, all_models = TRUE)
```

---

# posterior_epred *Generates posterior linear predictions for objects fitted by* bnec

---

## Description

Generates posterior linear predictions for objects fitted by bnec. object should be of class bayesnecfit or bayesmanecfit.

## Usage

```
## S3 method for class 'bayesnecfit'
posterior_epred(object, ...)

## S3 method for class 'bayesmanecfit'
posterior_epred(object, ...)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit. |
| ... | Additional arguments to posterior_epred. |

## Value

See ?brms:posterior_epred.

## Examples

```
## Not run:
library(bayesnec)
# Uses default `precision` and `x_range` to generate `newdata` internally
posterior_epred(manec_example)
# Provide user-specified `newdata`
nd_ <- data.frame(x = seq(0, 3, length.out = 200))
ppreds <- posterior_epred(manec_example, ecx_val = 50, newdata = nd_,
                          make_newdata = FALSE)
ncol(ppreds) == 200 # cols are x, rows are iterations
# Predictions for raw input data
nec4param <- pull_out(manec_example, model = "nec4param")
preds <- posterior_epred(nec4param, make_newdata = FALSE)
x <- pull_brmsfit(nec4param)$data$x
plot(sort(x), preds[1, order(x)], type = "l", col = alpha("black", 0.1),
     ylim = c(-6, 3))
for (i in seq_len(nrow(preds))[-1]) {
  lines(sort(x), preds[i, order(x)], type = "l", col = alpha("black", 0.1))
}

## End(Not run)
```

---

`posterior_epred_beta_binomial2` *posterior_epred_beta_binomial2*

---

## Description

Beta-binomial wrapper posterior_epred method

## Usage

```
posterior_epred_beta_binomial2(prep)
```

## Arguments

| | |
|---|---|
| `prep` | data with posterior. |

## Value

A `numeric` value or vector containing predicted random values of the beta binomial distribution

posterior_predict *Generates posterior predictions for objects fitted by* bnec

## Description

Generates posterior predictions for objects fitted by bnec. `object` should be of class bayesnecfit or bayesmanecfit.

## Usage

```
## S3 method for class 'bayesnecfit'
posterior_predict(object, ...)

## S3 method for class 'bayesmanecfit'
posterior_predict(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | An object of class bayesnecfit or bayesmanecfit. |
| `...` | Additional arguments to posterior_predict. |

## Value

See ?brms::posterior_predict.

## Examples

```
## Not run:
library(bayesnec)
# Uses default `precision` and `x_range` to generate `newdata` internally
posterior_predict(manec_example)
# Provide user-specified `newdata`
nd_ <- data.frame(x = seq(0, 3, length.out = 200))
ppreds <- posterior_predict(manec_example, ecx_val = 50, newdata = nd_,
                            make_newdata = FALSE)
ncol(ppreds) == 200 # cols are x, rows are iterations
# Posterior predictions for raw input data
nec4param <- pull_out(manec_example, model = "nec4param")
preds <- posterior_predict(nec4param, make_newdata = FALSE)
x <- pull_brmsfit(nec4param)$data$x
plot(sort(x), preds[1, order(x)], type = "l", col = alpha("black", 0.1),
     ylim = c(-8, 5))
for (i in seq_len(nrow(preds))[-1]) {
  lines(sort(x), preds[i, order(x)], type = "l", col = alpha("black", 0.1))
}

## End(Not run)
```

## posterior_predict_beta_binomial2 *posterior_predict_beta_binomial2*

### Description

Beta-binomial wrapper posterior_predict method

### Usage

```
posterior_predict_beta_binomial2(i, prep, ...)
```

### Arguments

| | |
|---|---|
| i | observation i. |
| prep | data with posterior. |
| ... | unused. |

### Value

A numeric value or vector containing predicted probability values of the beta binomial distribution

## prebayesnecfit-class *Class* prebayesnecfit *of models fitted with the* **brms** *package*

### Description

This is an intermediate class that was created to make both bayesnecfit and bayesmanecfit objects lighter to handle. It contains the original brmsfit fitted object, name of non-linear model that was fitted, the list of initialisation values applied, and the validated bayesnecformula.

### Details

See methods(class = "prebayesnecfit") for an overview of available methods.

### Slots

fit The fitted Bayesian model of class brmsfit.

model A character string indicating the name of the fitted model.

init A list containing the initialisation values for to fit the model.

bayesnecformula An object of class bayesnecformula and formula.

### See Also

bayesnec, bnec, bayesnecfit, bayesmanecfit, bayesnecformula

predict *Generates mean posterior predictions for objects fitted by* bnec

## Description

Generates mean posterior predictions for objects fitted by bnec. object should be of class bayesnecfit or bayesmanecfit.

## Usage

```
## S3 method for class 'bayesnecfit'
predict(object, ...)

## S3 method for class 'bayesmanecfit'
predict(object, summary = TRUE, robust = FALSE, probs = c(0.025, 0.975), ...)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit. |
| ... | Additional arguments to predict.brmsfit if object is of class bayesnecfit, or to posterior_predict.brmsfit if object is of class bayesmanecfit. |
| summary | Should summary statistics be returned instead of the raw values? Default is TRUE. |
| robust | If FALSE (the default) the mean is used as the measure of central tendency and the standard deviation as the measure of variability. If TRUE, the median and the median absolute deviation (MAD) are applied instead. Only used if summary is TRUE. |
| probs | The percentiles to be computed by the quantile function. Only used if summary is TRUE. |

## Value

See ?brms::predict.brmsfit.

## Examples

```
## Not run:
library(bayesnec)
# Uses default `precision` and `x_range` to generate `newdata` internally
predict(manec_example)
# Provide user-specified `newdata`
nd_ <- data.frame(x = seq(0, 3, length.out = 200))
predict(manec_example, ecx_val = 50, newdata = nd_, make_newdata = FALSE)
# Predictions for raw input data
nec4param <- pull_out(manec_example, model = "nec4param")
preds <- predict(nec4param, make_newdata = FALSE)
x <- pull_brmsfit(nec4param)$data$x
```

```
plot(x, preds[, 1])

## End(Not run)
```

---

print *Prints a summary for objects fitted by* bnec

---

## Description

Prints a summary for objects fitted by bnec. x should be of class bayesnecfit or bayesmanecfit.

## Usage

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

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

## Arguments

| | |
|---|---|
| x | An object of class bayesnecfit or bayesmanecfit. |
| ... | Unused. |

## Value

A summary print of the fitted model as returned for a brmsfit object.

## Examples

```
library(bayesnec)
print(manec_example)
nec4param <- pull_out(manec_example, "nec4param")
print(nec4param)
```

pull_brmsfit.bayesnecfit *Extract and object of class* brmsfit *from* bayesnecfit *or* bayesmanecfit.

## Description

Extract and object of class brmsfit from bayesnecfit or bayesmanecfit.

## Usage

```
## S3 method for class 'bayesnecfit'
pull_brmsfit(object, ...)

## S3 method for class 'bayesmanecfit'
pull_brmsfit(object, model = NA, ...)

pull_brmsfit(object, ...)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit returned by bnec. |
| ... | Arguments passed to other methods. |
| model | An optional character vector specifying the model to extract. |

## Value

A plot of the prior and posterior parameter probability densities.

## See Also

bnec

## Examples

```
library(bayesnec)
data(manec_example)
brms_fit <- pull_brmsfit(manec_example, model = "nec4param")
```

pull_out *pull_out*

## Description

Subsets model(s) from an existing object of class bayesmanecfit

## Usage

```
pull_out(manec, model, loo_controls, ...)
```

## Arguments

| | |
|---|---|
| manec | An object of class bayesmanecfit as returned by bnec. |
| model | A character string indicating which model or suite of models to pull out. |
| loo_controls | A named list of two elements ("fitting" and/or "weights"), each being a named list containing the desired arguments to be passed on to loo (via "fitting") or to loo_model_weights (via "weights"). If "fitting" is provided with argument pointwise = TRUE (due to memory issues) and family = "beta_binomial2", the bnec will fail because that is a custom family. If "weights" is not provided by the user, bnec will set the default method argument in loo_model_weights to "pseudobma". See ?loo_model_weights for further info. |
| ... | Additional arguments to expand_nec or expand_manec. |

## Value

If model is a string representing a single model, an object of class bayesnecfit; If model is instead a string depicting a suite of models, and object of class bayesmanecfit.

## See Also

bnec, models.

## Examples

```
## Not run:
library(bayesnec)
data(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
# use "ecx" to get all ECx-containing models
# (only one ["ecx4param"] in this minimal example)
ecx_models <- pull_out(manec_example, model = "ecx")

## End(Not run)
```

## pull_prior *pull_prior*

### Description

Extracts the priors from an object of class bayesnecfit or bayesmanecfit.

### Usage

```
pull_prior(object)
```

### Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit returned by bnec. |

### Value

A list containing the priors.

### Examples

```
library(bayesnec)
data(manec_example)
pull_prior(manec_example)
```

## rhat *Extract Diagnostic Quantities of 'brms' Models*

### Description

Extract Rhat statistic that can be used to diagnose sampling behaviour of the algorithms applied by 'Stan' at the back-end of 'brms'. x should be of class bayesnecfit or bayesmanecfit.

### Usage

```
## S3 method for class 'bayesnecfit'
rhat(x, rhat_cutoff = 1.05, ...)

## S3 method for class 'bayesmanecfit'
rhat(x, rhat_cutoff = 1.05, ...)
```

### Arguments

| | |
|---|---|
| x | An object of class bayesnecfit or bayesmanecfit. |
| rhat_cutoff | A numeric vector indicating the Rhat cut-off used to test for model convergence. |
| ... | Unused. |

## Value

A list containing a vector or Rhat values returned for each parameter for a brmsfit object, for each of the fitted models.

## Examples

```
## Not run:
library(bayesnec)
rhat(manec_example)
nec4param <- pull_out(manec_example, model = "nec4param")
rhat(nec4param)

## End(Not run)
```

---

sample_priors *sample_priors*

---

## Description

Creates list or generates a plot of prior samples

## Usage

```
sample_priors(priors, n_samples = 10000, plot = "ggplot")
```

## Arguments

| | |
|---|---|
| priors | An object of class brmsprior from package **brms**. |
| n_samples | The number of prior samples to return. |
| plot | NA returns a list of numeric vectors of sampled priors, "ggplot" (default) returns a ggplot and "base" returns a histogram in base R. |

## Value

A list containing the initialisation values.

## See Also

bnec

## Examples

```
library(bayesnec)
data(manec_example)
exmp <- pull_brmsfit(manec_example, model = "nec4param")
sample_priors(exmp$prior)
```

show_params *show_params*

## Description

Displays non-linear equation and parameter names

## Usage

```
show_params(model = "all")
```

## Arguments

| | |
|---|---|
| model | Removed in version 2.0. Use formula instead. Used to be a character vector indicating the model(s) to fit. See Details for more information. |

## Value

An list of brmsformula.

## Examples

```
library(bayesnec)
# default to all models (i.e. model = "all")
show_params()
# single model
show_params(model = "nec3param")
# group of models
show_params(model = c("nec3param", "ecx"))
```

summary *Generates a summary for objects fitted by* bnec

## Description

Generates a summary for objects fitted by bnec. object should be of class bayesnecfit or bayesmanecfit.

## Usage

```
## S3 method for class 'bayesnecfit'
summary(object, ..., ecx = FALSE, ecx_vals = c(10, 50, 90))

## S3 method for class 'bayesmanecfit'
summary(object, ..., ecx = FALSE, ecx_vals = c(10, 50, 90))
```

### Arguments

| | |
|---|---|
| object | An object of class bayesnecfit or bayesmanecfit. |
| ... | Unused. |
| ecx | Should summary ECx values be calculated? Defaults to FALSE. |
| ecx_vals | ECx targets (between 1 and 99). Only relevant if ecx = TRUE. If no value is specified by the user, returns calculations for EC10, EC50, and EC90. |

### Value

A summary of the fitted model. In the case of a bayesnecfit object, the summary contains most of the original contents of a brmsfit object with the addition of an R2. In the case of a bayesmanecfit object, summary displays the family distribution information, model weights and averaging method, the estimated model-averaged NEC, and R2 estimates for each individual model. Warning messages are also printed to screen in case model fits are not satisfactory with regards to their Rhats.

### Examples

```
library(bayesnec)
summary(manec_example)
nec4param <- pull_out(manec_example, "nec4param")
summary(nec4param)
```

---

## update.bnecfit *Update an object of class* bnecfit *as fitted by function* bnec.

---

### Description

Update an object of class bnecfit as fitted by function bnec.

### Usage

```
## S3 method for class 'bnecfit'
update(
  object,
  newdata = NULL,
  recompile = NULL,
  x_range = NA,
  precision = 1000,
  sig_val = 0.01,
  loo_controls,
  force_fit = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| object | An object of class bnecfit as fitted by function bnec. |
| newdata | Optional data.frame to update the model with new data. Data-dependent default priors will not be updated automatically. |
| recompile | A logical, indicating whether the Stan model should be recompiled. If NULL (the default), update tries to figure out internally, if recompilation is necessary. Setting it to FALSE will cause all Stan code changing arguments to be ignored. |
| x_range | A range of predictor values over which to consider extracting ECx. |
| precision | The length of the predictor vector used for posterior predictions, and over which to extract ECx values. Large values will be slower but more precise. |
| sig_val | Probability value to use as the lower quantile to test significance of the predicted posterior values against the lowest observed concentration (assumed to be the control), to estimate NEC as an interpolated NOEC value from smooth ECx curves. |
| loo_controls | A named list of two elements ("fitting" and/or "weights"), each being a named list containing the desired arguments to be passed on to loo (via "fitting") or to loo_model_weights (via "weights"). If "fitting" is provided with argument pointwise = TRUE (due to memory issues) and family = "beta_binomial2", the bnec will fail because that is a custom family. If "weights" is not provided by the user, bnec will set the default method argument in loo_model_weights to "pseudobma". See ?loo_model_weights for further info. |
| force_fit | Should model truly be updated in case either newdata of a new family is provided? |
| ... | Further arguments to brm. |

## Value

An object of class bnecfit. If one single model is returned, then also an object of class bayesnecfit; otherwise, if multiple models are returned, also an object of class bayesmanecfit.

## Examples

```
## Not run:
library(bayesnec)
data(manec_example)
# due to package size issues, `manec_example` does not contain original
# stanfit DSO, so need to recompile here
smaller_manec <- update(manec_example, chains = 2, iter = 50,
                        recompile = TRUE)
# original `manec_example` is fit with a Gaussian
# change to Beta distribution by adding newdata with original `nec_data$y`
# function will throw informative message.
beta_manec <- update(manec_example, newdata = nec_data, recompile = TRUE,
                     chains = 2, iter = 50,
                     family = Beta(link = "identity"), force_fit = TRUE)

## End(Not run)
```

# Index


∗ **datasets**
herbicide, 35
+.bnecfit, 3

amend, 4, *5*
autoplot, 5, *15*, *34*
average_endpoints, 7

bayesmanecfit, *3–8*, *10*, *15–19*, *21–25*, *28*, *32–34*, *39–47*, *49–55*, *57–59*
bayesmanecfit *(*bayesmanecfit-class*)*, 9
bayesmanecfit-class, 9
bayesnec, *5*, *9–12*, *17*, *20*, *37–39*, *43*, *50*
bayesnec *(*bayesnec-package*)*, 3
bayesnec-package, 3
bayesnecfit, *6–9*, *15–19*, *21–28*, *32–34*, *39–47*, *49–55*, *57–59*
bayesnecfit *(*bayesnecfit-class*)*, 9
bayesnecfit-class, 9
bayesnecformula, *9, 10*, 10, *11, 12*, *14, 15*, *17*, *20, 21*, *30, 31*, *33*, *37*, *39, 40*, *50*
beta_binomial2, 12
beta_binomial2_lpmf, 13
beta_binomial2_rng, 13
binomial, *27*
bnec, *4, 5*, *7–10*, *14*, 14, *15–26*, *28–34*, *39*, *41–45*, *47*, *49–59*
bnec_newdata, 17, *24*, *26*
bnecfit, *3*, *18*, *58, 59*
bnecfit *(*bnecfit-class*)*, 17
bnecfit-class, 17
bnf *(*bayesnecformula*)*, 10
brm, *11, 12*, *15, 16*, *37*, *59*
brmsfit, *9, 10*, *27*, *43*, *50*, *52, 53*, *56*, *58*
brmsformula, *11*, *37*, *57*
brmsprior, *5*, *56*

c.bnecfit, 18
character, *5*, *8–11*, *14, 15*, *19*, *22, 23*, *25*, *28*, *30, 31*, *37, 38*, *41*, *43, 44*, *46*, *50*, *53, 54*, *57*
check_chains, 19
check_formula, *12*, *14*, *17*, 20, *30, 31*, *37*, *40*
check_priors, 21
compare_endpoints, 22
compare_fitted, 24, *29*, *45*
compare_posterior, 25
customfamily, *12*

data.frame, *9, 10*, *14*, *18*, *20*, *27*, *34*, *37*, *40*, *59*
dispersion, 27

ecx, *26*, 28, *29*, *34*, *45*
exp, *8*
expand_manec, 30, *54*
expand_nec, 31, *54*

fitted, 32
fitted.brmsfit, *32*
formula, *10–12*, *14, 15*, *20*, *30, 31*, 33, *37*, *50*

ggbnec_data, *6*, 34
ggplot, *6*, *56*

herbicide, 35
hypothesis, *16*

is_manecsummary, 36
is_necsummary, 36

list, *5*, *7*, *9, 10*, *14, 15*, *23–26*, *30, 31*, *37–39*, *41*, *43*, *50*, *54–57*, *59*
log_lik_beta_binomial2, 37
logical, *7*, *20*, *28*, *36*, *38*, *42–44*, *46, 47*, *59*
loo, *5*, *14*, *16*, *30, 31*, *54*, *59*
loo_model_weights, *5*, *14*, *16*, *30, 31*, *54*, *59*

make_brmsformula, *12*, 37
manec_example, 39
manecsummary, *43*
manecsummary *(*manecsummary-class*)*, 38

manecsummary-class, 38
model.frame, *12*, *15*, *20*, 39
models, *11, 12*, *15–17*, *41*, 41, *54*

nec, *26*, 42
nec_data, *39*, 43
necsummary, *39*
necsummary *(*necsummary-class*)*, 43
necsummary-class, 43
nsec, *26*, *29*, 44, *45*
numeric, *13*, *18*, *27*, *41*, *46*, *48*, *50*, *55*

plot, *15*, 45, *46, 47*
poisson, *27*
posterior_epred, *24*, *26*, *47*, 47
posterior_epred.brmsfit, *32*
posterior_epred_beta_binomial2, 48
posterior_predict, *29*, *45*, *49*, 49
posterior_predict.brmsfit, *51*
posterior_predict_beta_binomial2, 50
prebayesnecfit, *30, 31*
prebayesnecfit *(*prebayesnecfit-class*)*, 50
prebayesnecfit-class, 50
predict, 51
predict.brmsfit, *51*
print, *15*, 52
pull_brmsfit *(*pull_brmsfit.bayesnecfit*)*, 53
pull_brmsfit.bayesnecfit, 53
pull_out, *16*, 54
pull_prior, 55

rhat, 55

sample_priors, 56
show_params, *12*, *16, 17*, *41*, 57
summary, *15*, 57

update.bnecfit, 58