Package 'hmetad'

March 16, 2026

Title Fit the Meta-D' Model of Confidence Ratings Using 'brms'

Version 0.1.0

Description Implementation of Bayesian regressions over the meta-d' model of psychological data from two alternative forced choice tasks with ordinal confidence ratings. For more information, see Maniscalco & Lau (2012) <doi:10.1016/j.concog.2011.09.021>. The package is a front-end to the 'brms' package, which facilitates a wide range of regression designs, as well as tools for efficiently extracting posterior estimates, plotting, and significance testing.

License GPL (>= 3)

Depends R (>= 4.2.0), brms (>= 2.23.0)

Imports abind (>= 1.4.8), dplyr (>= 1.2.0), glue (>= 1.8.0), posterior (>= 1.6.1), rlang (>= 1.1.7), stats, stringr (>= 1.6.0), tidybayes (>= 3.0.7), tidyr (>= 1.3.2)

Suggests colorspace, knitr, purrr (>= 1.2.1), rmarkdown, testthat (>= 3.0.0), tidyverse (>= 2.0.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

Config/testthat/edition 3

URL https://metacoglab.github.io/hmetad/, https://github.com/metacoglab/hmetad

BugReports https://github.com/metacoglab/hmetad/issues

LazyData true

NeedsCompilation no

Author Kevin O'Neill [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-7401-9802>), Stephen Fleming [aut, cph] (ORCID: <https://orcid.org/0000-0003-0233-4891>)

Maintainer Kevin O'Neill <kevin.o'neill@ucl.ac.uk>

Repository CRAN

Date/Publication 2026-03-16 16:00:26 UTC

Contents

aggregate_metad . . . 2
cor_matrix . . . 4
cov_matrix . . . 5
epred_draws_metad . . . 5
example_data . . . 6
example_model . . . 7
fit_metad . . . 8
joint_response . . . 10
linpred_draws_metad . . . 11
mean_confidence_draws . . . 12
metacognitive_bias_draws . . . 14
metad . . . 16
metad_pmf . . . 17
normal_lcdf . . . 18
predicted_draws_metad . . . 19
response_probabilities . . . 20
rmatrixnorm . . . 21
roc1_draws . . . 21
roc2_draws . . . 23
sim_metad . . . 24
sim_metad_condition . . . 26
sim_metad_participant . . . 27
sim_metad_participant_condition . . . 29
stanvars_metad . . . 31
to_signed . . . 32

Index **34**

`aggregate_metad` *Aggregate* `data` *by* `response`, `confidence`, *and other columns*

Description

Counts number of rows in `data` with unique combinations values in the columns `response`, `confidence`, and any other columns in `...`.

Usage

```
aggregate_metad(
  data,
  ...,
  .stimulus = "stimulus",
  .response = "response",
  .confidence = "confidence",
  .joint_response = "joint_response",
  .name = "N",
  K = NULL
)
```

Arguments

`data`	The data frame to aggregate
`...`	Grouping columns in `data`. These columns will be converted to factors.
`.stimulus`	The name of "stimulus" column
`.response`	The name of "response" column
`.confidence`	The name of "confidence" column
`.joint_response`	The name of "joint_response" column
`.name`	The name of the resulting column containing trial counts
`K`	The number of confidence levels in `data`. If `NULL`, this is estimated from `data`.

Details

The data frame `data` must have one column with the name given by `.stimulus`. Additionally, it must have either:

- Two columns with names given by `.response` and `.confidence`
- One column with the name given by `.joint_response`

Finally, it must also have columns for any additional variables in `...`.

Value

A tibble with one row per combination of the variables in `...`, and another column named by the value of `.response` containing trial counts. For K confidence levels, this will be an $N \times K * 4$ matrix, such that the columns represent (for stimulus S, type 1 response R, and type 2 response C):

$[N_{S=0,R=0,C=K}, \ldots, N_{S=0,R=0,C=1}, N_{S=0,R=1,C=1}, \ldots, N_{S=0,R=1,C=K}, N_{S=1,R=0,C=K}, \ldots, N_{S=1,R=0,C=1}, N_{S=1,R=$

Examples

```
# aggregate a dataset without grouping factors
d <- sim_metad()
aggregate_metad(d)
```

```
# aggregate a dataset with grouping factors
d2 <- sim_metad_condition()
aggregate_metad(d2, condition)

# can also aggregate ignoring grouping factors
aggregate_metad(d2)

# aggregate data with only `joint_response` column
library(dplyr)
d |>
  ungroup() |>
  mutate(joint_response = joint_response(
    response, confidence,
    n_distinct(confidence)
  )) |>
  select(-response, -confidence) |>
  aggregate_metad()
```

`cor_matrix` *Generate a correlation matrix with all off-diagonal values equal to* `r`

Description

Generate a correlation matrix with all off-diagonal values equal to `r`

Usage

```
cor_matrix(r, nrow = 2)
```

Arguments

`r`	The correlation to fill in the matrix off-diagonals
`nrow`	The number of rows (and columns) of the resulting matrix

Value

An `[nrow x nrow]` matrix with values along the diagonal equal to `1` and values off of the diagonal equal to `r`

Examples

```
cor_matrix(0, nrow = 3)

cor_matrix(-.5, nrow = 4)
```

cov_matrix *Generate a covariance matrix.*

Description

Generate a covariance matrix.

Usage

```
cov_matrix(S, OMEGA)
```

Arguments

S	A vector of standard deviations
OMEGA	A correlation matrix

Value

an $N \times N$ covariance matrix, where `N = length(S)`.

Examples

```
sds <- c(1, 2)
corrs <- matrix(c(1, .5, .5, 1), nrow = 2)
cov_matrix(sds, corrs)
```

epred_draws_metad *Obtain posterior draws of joint response probabilities*

Description

Given a data frame and a meta-d' model, adds estimates of joint type 1 and type 2 response probabilities. For `epred_draws_metad` and `add_epred_draws_metad`, estimates are returned in a tidy tibble with one row per posterior draw. For `epred_rvars_metad` and `add_epred_rvars_metad`, parameters are returned as posterior::rvars, with one row per row in `newdata`.

Usage

```
epred_draws_metad(object, newdata, ...)

add_epred_draws_metad(newdata, object, ...)

epred_rvars_metad(object, newdata, ...)

add_epred_rvars_metad(newdata, object, ...)
```

Arguments

object	The brms model with the metad family
newdata	A data frame from which to generate posterior predictions
...	Additional arguments passed to tidybayes::add_epred_draws or tidybayes::add_epred_rvars

Value

a tibble containing posterior draws of model parameters with the following columns:

- .row: the row of newdata
- .chain, .iteration, .draw: for epred_draws_metad, identifiers for the posterior sample
- stimulus, joint_response, response, confidence: identifiers for the response type
- .epred: probability of the type 1 and type 2 response given the stimulus, $P(R, C \mid S)$

See Also

tidybayes::epred_draws(), tidybayes::epred_rvars()

Examples

```
newdata <- tidyr::tibble(.row = 1)

# obtain model predictions
# equivalent to `add_epred_draws_metad(newdata, example_model)`
epred_draws_metad(example_model, newdata)

# obtain model predictions (`posterior::rvar`)
# equivalent to `add_epred_rvars_metad(newdata, example_model)`
epred_rvars_metad(example_model, newdata)
```

---

example_data *Simulated data for example model fitting*

Description

A simulated data set of 1000 trials from a two-alternative forced choice task with 4 levels of confidence.

Usage

```
example_data
```

Format

A tibble of 1000 observations containing the following columns:

- `trial`: the trial number
- `stimulus`: the stimulus presence (`0` or `1`)
- `response`: the simulated type 1 response
- `confidence`: the simulated type 2 response
- `correct`: the accuracy of the simulated type 1 response
- `dprime`, `c`, `meta_dprime`, `M`, `meta_c2_0`, `meta_c2_1`: the parameters of the model used for simulation
- `theta`, `theta_1`, `theta_2`: the joint, type 1, and type 2 response probabilities of the model used for simulation

Source

Generated using the code `sim_metad(N_trials = 1000)`

See Also

`sim_metad()`

Examples

```
fit_metad(N ~ 1, example_data, chains = 1, iter = 500)
```

`example_model` *Example meta-d' model for model post-processing*

Description

A model fit to the simulated data example_data. This model includes one constant set of parameters, with no multilevel structure.

Usage

```
example_model
```

Format

A `brmsfit` object

Source

Generated using the code `fit_metad(N ~ 1, example_data, iter = 500)`

See Also

fit_metad()

Examples

```
# inspect summary of posterior distribution
summary(example_model)

# obtain posterior expectations
epred_draws_metad(example_model, tidyr::tibble(.row = 1))
```

---

fit_metad *Fit the meta-d' model using* brms *package*

Description

This function is a wrapper around brms::brm() using a custom family for the meta-d' model.

Usage

```
fit_metad(
  formula,
  data,
  ...,
  aggregate = TRUE,
  .stimulus = "stimulus",
  .response = "response",
  .confidence = "confidence",
  .joint_response = "joint_response",
  K = NULL,
  distribution = "normal",
  metac_absolute = TRUE,
  stanvars = NULL,
  categorical = FALSE
)
```

Arguments

formula	A model formula for some or all parameters of the metad brms family. To display all parameter names for a model with K confidence levels, use metad(K).
data	A tibble containing the data to fit the model. • If aggregate==TRUE, data should have one row per observation with columns stimulus, response, confidence, and any other variables in formula

- If aggregate==FALSE, it should be aggregated to have one row per cell of the design matrix, with joint type 1/type 2 response counts in a matrix column (see aggregate_metad()).

...	Additional parameters passed to the brm function.
aggregate	If TRUE, automatically aggregate data by the variables included in formula using aggregate_metad(). Otherwise, data should already be aggregated.
.stimulus	The name of "stimulus" column
.response	The name of "response" column
.confidence	The name of "confidence" column
.joint_response	The name of "joint_response" column
K	The number of confidence levels. By default, this is estimated from the data.
distribution	The noise distribution to use for the signal detection model. By default, uses a normal distribution with a mean parameterized by dprime.
metac_absolute	If TRUE, fix the type 2 criterion to be equal to the type 1 criterion. Otherwise, equate the criteria relatively such that metac/metadprime = c/dprime.
stanvars	Additional stanvars to pass to the model code, for example to define an alternative distribution or a custom model prior (see brms::stanvar()).
categorical	If FALSE (default), use the multinomial likelihood over aggregated data. If TRUE, use the categorical likelihood over individual trials.

Details

fit_metad(formula, data, ...) is approximately the same as brm(formula, data=aggregate_metad(data, ...), family=metad(...), stanvars=stanvars_metad(...), ...). For some models, it may often be easier to use the more explicit version than using fit_metad.

Value

A brmsfit object containing the fitted model

Examples

```
# check which parameters the model has
metad(3)

# fit a basic model on simulated data
# (use `empty=true` to bypass fitting, *do not use in real analysis*)
fit_metad(N ~ 1, sim_metad(), empty = TRUE)

# fit a basic model on simulated data
fit_metad(N ~ 1, sim_metad())

# fit a model with condition-level effects
fit_metad(
  bf(
    N ~ condition,
```

```
    dprime + c + metac2zero1diff + metac2zero2diff +
      metac2one1diff + metac2one1diff ~ condition
  ),
  data = sim_metad_condition()
)
```

`joint_response` *Convert between separate and joint type 1/type 2 responses*

Description

Confidence ratings and decisions are collected in one of two ways.

- For separate ratings, there will be a type 1 response ($R \in \{0, 1\}$) and a type 2 response ($C \in [1, K]$).
- For joint ratings, there is instead a combined type 1/type 2 response ($J \in [1, 2K]$), with values in $[1, K]$ indicating a type 1 response of 0 and values in $[K + 1, 2K]$ indicating a type 1 response of 1, with confident responses at the ends of the scale.

`joint_response` converts separate type 1 and type 2 responses into the joint format

`type1_response` and `type2_response` convert the joint response into separate responses.

Usage

```
joint_response(response, confidence, K)

type1_response(joint_response, K)

type2_response(joint_response, K)
```

Arguments

`response`	A type 1 response (`0` or `1`)
`confidence`	A type 2 response/confidence rating (in `1:K`)
`K`	The number of confidence levels
`joint_response`	A joint type 1/type 2 response

Value

A joint response (for `joint_response`), type 1 response (for `type1_response`), or type 2 response (for `type2_response`)

Examples

```
# convert joint_response to separate responses
joint <- 1:8
K <- 4
type1_response(joint, K)
type2_response(joint, K)

# convert separate responses to a joint response
t1 <- rep(c(0, 1), each = 4)
t2 <- c(4:1, 1:4)
joint_response(t1, t2, K)
```

---

linpred_draws_metad *Obtain posterior draws of meta-d' model parameters*

Description

Given a data frame and a meta-d' model, adds estimates of all model parameters. For `linpred_draws_metad` and `add_linpred_draws_metad`, parameters are returned in a tidy tibble with one row per posterior draw. For `linpred_rvars_metad` and `add_linpred_rvars_metad`, parameters are returned as posterior::rvars, with one row per row in `newdata`.

Usage

```
linpred_draws_metad(object, newdata, ..., pivot_longer = FALSE)

add_linpred_draws_metad(newdata, object, ..., pivot_longer = FALSE)

linpred_rvars_metad(object, newdata, ..., pivot_longer = FALSE)

add_linpred_rvars_metad(newdata, object, ..., pivot_longer = FALSE)
```

Arguments

`object`	The `brms` model with the `metad` family
`newdata`	A data frame from which to generate posterior predictions
`...`	Additional arguments passed to tidybayes::add_linpred_draws or tidybayes::add_linpred_rvars
`pivot_longer`	Return the draws in long format? • if `TRUE`, resulting data frame has one row per posterior draw per model parameter • if `FALSE` (default), resulting data frame has one row per posterior draw

Value

a tibble containing posterior draws of model parameters with the following columns:

- `.row`: the row of `newdata`
- `.chain`, `.iteration`, `.draw`: for `linpred_draws_metad`, identifiers for the posterior sample
- `.variable`, `.value`: if `pivot_longer=TRUE`, `.variable` identifies different meta-d' model parameters and `.value` stores posterior samples
- `M`, `dprime`, `c`, `meta_dprime`, `meta_c`, `meta_c2_0_<k>`, `meta_c2_1_<k>`: if `pivot_longer=FALSE`, posterior samples of all meta-d' model parameters

See Also

`tidybayes::linpred_draws()`, `tidybayes::linpred_rvars()`

Examples

```
newdata <- tidyr::tibble(.row = 1)

# obtain model parameters (wide format)
# equivalent to `add_linpred_draws_metad(newdata, example_model)`
linpred_draws_metad(example_model, newdata)

# obtain model parameters (long format)
# equivalent to `add_linpred_draws_metad(newdata, example_model, pivot_longer = TRUE)`
linpred_draws_metad(example_model, newdata, pivot_longer = TRUE)

# obtain model parameters (wide format, posterior::rvar)
# equivalent to `add_linpred_rvars_metad(newdata, example_model)`
linpred_rvars_metad(example_model, newdata)

# obtain model parameters (long format, posterior::rvar)
# equivalent to `add_linpred_rvars_metad(newdata, example_model, pivot_longer = TRUE)`
linpred_rvars_metad(example_model, newdata, pivot_longer = TRUE)
```

`mean_confidence_draws` *Obtain posterior draws of mean confidence*

Description

Computes posterior mean confidence conditional on stimulus and response ($\mathbb{E}[C \mid S = s, R = r]$), stimulus (averaging over responses, $\mathbb{E}[C \mid S = s]$), response (averaging over stimuli, $\mathbb{E}[C \mid R = r]$), neither (averaging over stimuli and responses, $\mathbb{E}[C]$), or accuracy ($\mathbb{E}[C \mid A = (r = s)]$). For `mean_confidence_draws` and `add_mean_confidence_draws`, estimates are returned in a tidy tibble with one row per posterior draw, stimulus, and response. For `mean_confidence_rvars` and `add_mean_confidence_rvars`, estimates are returned as posterior::rvars, with one row per row in `newdata`.

`add_mean_confidence_draws` is an alias of `mean_confidence_draws` with argument order swapped.

Usage

```
mean_confidence_draws(
  object,
  newdata,
  ...,
  by_stimulus = TRUE,
  by_response = TRUE,
  by_correct = FALSE
)

add_mean_confidence_draws(newdata, object, ...)

mean_confidence_rvars(
  object,
  newdata,
  ...,
  by_stimulus = TRUE,
  by_response = TRUE,
  by_correct = FALSE
)

add_mean_confidence_rvars(newdata, object, ...)
```

Arguments

`object`	The `brms` model with the `metad` family
`newdata`	A data frame from which to generate posterior predictions
`...`	Additional arguments to tidybayes::epred_draws or tidybayes::epred_rvars
`by_stimulus`	If TRUE, predict mean confidence separately by stimulus. Otherwise, predict mean confidence averaging over stimuli. Ignored if `by_correct==TRUE`.
`by_response`	If TRUE, predict mean confidence separately by response Otherwise, predict mean confidence averaging over responses. Ignored if `by_correct==TRUE`.
`by_correct`	If TRUE, predict mean confidence separately for correct and incorrect responses.

Value

a tibble containing posterior draws of mean confidence with the following columns:

- `.row`: the row of `newdata`
- `.chain`, `.iteration`, `.draw`: for `mean_confidence_draws` and `add_mean_confidence_draws`, identifiers for the posterior sample
- `stimulus`: indicator for stimulus presence (if `by_stimulus==TRUE` & `by_correct==FALSE`)
- `response`: indicator for type 1 response (if `by_response==TRUE` & `by_correct==FALSE`)
- `correct`: indicator for the accuracy of the type 1 response (if `by_correct==TRUE`)
- `.epred`: the predicted mean confidence

See Also

tidybayes::epred_draws(), tidybayes::epred_rvars()

Examples

```
newdata <- tidyr::tibble(.row = 1)

# compute mean confidence by stimulus and response
# equivalent to `add_mean_confidence_draws(newdata, example_model)`
mean_confidence_draws(example_model, newdata)

# compute mean confidence by stimulus
# equivalent to `add_mean_confidence_draws(newdata, example_model, by_response = FALSE)`
mean_confidence_draws(example_model, newdata, by_response = FALSE)

# compute mean confidence by response
# equivalent to `add_mean_confidence_draws(newdata, example_model, by_stimulus = FALSE)`
mean_confidence_draws(example_model, newdata, by_stimulus = FALSE)

# compute mean confidence by accuracy
# equivalent to `add_mean_confidence_draws(newdata, example_model, by_correct = TRUE)`
mean_confidence_draws(example_model, newdata, by_correct = TRUE)

# compute mean confidence averaging over stimuli and responses
# equivalent to `add_mean_confidence_draws(newdata, example_model, ...)`
mean_confidence_draws(example_model, newdata, by_stimulus = FALSE, by_response = FALSE)

# use `posterior::rvar` for increased efficiency
# equivalent to `add_mean_confidence_rvars(newdata, example_model)`
mean_confidence_rvars(example_model, newdata)
```

---

metacognitive_bias_draws

Obtain posterior draws of an index of metacognitive bias

Description

Computes meta-Δ, an index of metacognitive bias. meta-Δ is the distance between `meta_c` and the average of the the confidence criteria `meta_c2_0` and `meta_c2_1`. For `metacognitive_bias_draws` and `add_metacognitive_bias_draws`, parameters are returned in a tidy tibble with one row per posterior draw and per response. For `metacognitive_bias_rvars` and `add_metacognitive_bias_rvars`, parameters are returned as posterior::rvars, with one row per row in `newdata` and per response.

Usage

```
metacognitive_bias_draws(object, newdata, ..., by_response = TRUE)

add_metacognitive_bias_draws(newdata, object, ...)
```

```
metacognitive_bias_rvars(object, newdata, ..., by_response = TRUE)

add_metacognitive_bias_rvars(newdata, object, ...)
```

Arguments

object	The brms model with the metad family
newdata	A data frame from which to generate posterior predictions
...	Additional parameters passed to tidybayes::epred_draws or tidybayes::epred_rvars
by_response	If TRUE, compute metacognitive bias separately for the two type 1 responses. If FALSE, compute an un-weighted average of the two measures.

Value

a tibble containing posterior draws of meta-Δ with the following columns:

- .row: the row of newdata
- .chain, .iteration, .draw: for metacognitive_bias_draws and add_metacognitive_bias_draws, identifiers for the posterior sample
- response: the type 1 response for perceived stimulus presence
- metacognitive_bias: the distance between meta_c and the average of the confidence criteria meta_c2_{response}.

See Also

tidybayes::linpred_draws(), tidybayes::linpred_rvars()

Examples

```
newdata <- tidyr::tibble(.row = 1)

# compute metacognitive bias
# equivalent to `add_metacognitive_bias_draws(newdata, example_model)`
metacognitive_bias_draws(example_model, newdata)

# use `posterior::rvar` for increased efficiency
# equivalent to `add_metacognitive_bias_rvars(newdata, example_model)`
metacognitive_bias_rvars(example_model, newdata)

# average over the two type 1 responses
metacognitive_bias_rvars(example_model, newdata, by_response = FALSE)
```

`metad` `brms` *family for the metad' model*

Description

`brms` family for the metad' model

Usage

```
metad(K, distribution = "normal", metac_absolute = TRUE, categorical = FALSE)
```

Arguments

`K`	The number of confidence levels
`distribution`	The noise distribution to use for the signal detection model
`metac_absolute`	If `TRUE`, fix the type 2 criterion to be equal to the type 1 criterion. Otherwise, equate the criteria relatively such that $$\frac{\text{meta-}c}{\text{meta-}d'} = \frac{c}{d'}$$
`categorical`	If `FALSE` (default), use the multinomial likelihood over aggregated data. If `TRUE`, use the categorical likelihood over individual trials.

Value

A `brms` family for the metad' model with K confidence levels

Examples

```
# create a family using the normal distribution and 3 levels of confidence
metad(3)

# create a family with meta_c = M * c
metad(3, metac_absolute = FALSE)

# create a family with an alternative distribution
# note: cumulative distribution functions must be defined
# in R and in Stan using [brms::stanvar()]
metad(4, distribution = "gumbel_min")
```

`metad_pmf` *Generate (log) probability simplex over the joint type 1/type 2 responses*

Description

Generate (log) probability simplex over the joint type 1/type 2 responses

Usage

```
metad_pmf(
  stimulus,
  dprime,
  c,
  meta_dprime,
  meta_c,
  meta_c2_0,
  meta_c2_1,
  lcdf = normal_lcdf,
  lccdf = normal_lccdf,
  log = FALSE
)
```

Arguments

`stimulus`	the stimulus (0 or 1)
`dprime`	the type 1 sensitivity
`c`	the type 1 response criterion
`meta_dprime`	the type 2 sensitivity
`meta_c`	the type 1 criteriom for generating confidence ratings
`meta_c2_0`	the type 2 response criteria for `"0"` responses, indexed by increasing confidence levels
`meta_c2_1`	the type 2 response criteria for `"1"` responses, indexed by increasing confidence levels
`lcdf`	The log cumulative distribution function for the underlying distribution in the metad' model. By default, uses the normal distribution with a standard deviation of 1.
`lccdf`	The log complement cumulative distribution function for the underlying distribution in the metad' model. By default, uses the normal distribution with a standard deviation of 1.
`log`	if TRUE, return log probabilities instead of probabilities

Value

A probability simplex

$$\left[P(R=0,C=K|S=0),\ldots,P(R=0,C=1|S=0),P(R=0,C=1|S=1),\ldots,P(R=1,C=1|S=1)\right]$$

for response R and confidence C given stimulus S, as defined by the meta-d' model.

Examples

```
metad_pmf(
  stimulus = 0, dprime = 2, c = .5, meta_dprime = 1, meta_c = .5,
  meta_c2_0 = c(0, -.5), meta_c2_1 = c(1, 1.5)
)
```

`normal_lcdf` *Normal cumulative distribution functions*

Description

Normal cumulative distribution functions

Usage

```
normal_lcdf(x, mu)

normal_lccdf(x, mu)
```

Arguments

`x`	The quantile to evaluate the l(c)cdf at
`mu`	The mean of the normal distribution

Value

$log(P(X < x))$ (for `normal_lcdf`) or $log(P(X > x))$ (for `normal_lccdf`) where X is sampled from a normal distribution with mean `mu` and standard deviation of 1

Examples

```
normal_lcdf(0, mu = 1)
normal_lccdf(0, mu = 1)
```

predicted_draws_metad *Obtain posterior predictions of joint responses*

Description

Given a data frame and a meta-d' model, adds predictions of joint type 1 and type 2 responses For `predicted_draws_metad` and `add_predicted_draws_metad`, predictions are returned in a tidy tibble with one row per posterior draw. For `predicted_rvars_metad` and `add_predicted_rvars_metad`, parameters are returned as posterior::rvars, with one row per row in `newdata`.

Usage

```
predicted_draws_metad(object, newdata, ...)

add_predicted_draws_metad(newdata, object, ...)

predicted_rvars_metad(object, newdata, ...)

add_predicted_rvars_metad(newdata, object, ...)
```

Arguments

`object`	The `brms` model with the `metad` family
`newdata`	A data frame from which to generate posterior predictions
`...`	Additional arguments passed to tidybayes::add_predicted_draws or tidybayes::add_predicted_rvars

Value

a tibble containing posterior draws of model parameters with the following columns:

- `.row`: the row of `newdata`
- `.chain`, `.iteration`, `.draw`: for `predicted_draws_metad`, identifiers for the posterior sample
- `stimulus`, `joint_response`, `response`, `confidence`: identifiers for the response type
- `.prediction`: predicted type 1 and type 2 responses given the stimulus

See Also

`tidybayes::predicted_draws()`, `tidybayes::predicted_rvars()`

Examples

```
newdata <- aggregate_metad(example_data)

# obtain model predictions
# equivalent to `add_predicted_draws_metad(newdata, example_model)`
predicted_draws_metad(example_model, newdata)
```

```
# obtain model predictions (posterior::rvar)
# equivalent to `add_predicted_rvars_metad(newdata, example_model)`
predicted_rvars_metad(example_model, newdata)
```

`response_probabilities` *Compute joint response probabilities from aggregated counts*

Description

Compute joint response probabilities from aggregated counts

Usage

```
response_probabilities(counts)
```

Arguments

`counts`	A vector (or matrix) of counts of joint type 1/type 2 responses as provided by aggregate_metad

Details

For response R, confidence C, stimulus S, and number of confidence levels K, `counts` should be a vector (or matrix with rows) of the form:

$$[N_{S=0,R=0,C=K}, \ldots, N_{S=0,R=0,C=1}, N_{S=0,R=1,C=1}, \ldots, N_{S=0,R=1,C=K}, N_{S=1,R=0,C=K}, \ldots, N_{S=1,R=0,C=1}, N_{S=1,R=}$$

Returns a vector (or matrix with rows) of the form:

$$[P(R=0,C=K \mid S=0), ..., P(R=0,C=1 \mid S=0), P(R=1,C=1 \mid S=0), ..., P(R=1,C=K \mid S=0), P(R$$

Value

A vector (or matrix) of response probabilities $P(R, C \mid S)$

Examples

```
# Aggregate responses from simulated data
d <- sim_metad() |> aggregate_metad()

# Compute conditional response probabilities
response_probabilities(d$N)

# Also works on matrices
matrix(rep(1, 16), nrow = 2) |> response_probabilities()
```

`rmatrixnorm` *Sample from a matrix-normal distribution*

Description

Sample from a matrix-normal distribution

Usage

```
rmatrixnorm(mu, L_sigma_rows, L_sigma_cols)
```

Arguments

`mu`	a matrix of means
`L_sigma_rows`	the Cholesky-decomposed covariance matrix for the rows
`L_sigma_cols`	the Cholesky-decomposed covariance matrix for the columns

Value

A single sample from a matrix-normal distribution with mean `mu` (a matrix), row-wise covariances `sigma_rows`, and column-wise covariances `sigma_cols`, where `L_sigma_rows` and `L_sigma_cols` are the Cholesky-decomposed covariance matrices

Examples

```
mu <- matrix(rep(0, 8), nrow = 4)
sd_rows <- rep(1, 4)
sd_cols <- rep(1, 2)
r_rows <- cor_matrix(.25, 4)
r_cols <- cor_matrix(.75, 2)
L_sigma_rows <- chol(cov_matrix(sd_rows, r_rows))
L_sigma_cols <- chol(cov_matrix(sd_cols, r_cols))
rmatrixnorm(mu, L_sigma_rows, L_sigma_cols)
```

`roc1_draws` *Obtain posterior draws of the pseudo type 1 receiver operating characteristic (ROC) curve.*

Description

Given a data frame and a meta-d' model, adds estimates of the cumulative probability over joint_responses. For `roc1_draws` and `add_roc1_draws`, estimates are returned in a tidy tibble with one row per posterior draw and per joint response. For `roc1_rvars` and `add_roc1_rvars`, parameters are returned as posterior::rvars, with one row per row in `newdata` and per joint response.

Usage

```
roc1_draws(object, newdata, ..., bounds = FALSE)

add_roc1_draws(newdata, object, ...)

roc1_rvars(object, newdata, ..., bounds = FALSE)

add_roc1_rvars(newdata, object, ...)
```

Arguments

object	The brms model with the metad family
newdata	A data frame from which to generate posterior predictions
...	Additional parameters passed to tidybayes::epred_draws or tidybayes::epred_rvars
bounds	If TRUE, include the endpoints of the ROC at $(0, 0)$ and $(1, 1)$. Otherwise, the endpoints are excluded.

Value

a tibble containing posterior draws of the pseudo type 1 ROC with the following columns:

- .row: the row of newdata
- .chain, .iteration, .draw: for roc1_draws and add_roc1_draws, identifiers for the posterior sample
- joint_response: the combined type 1 / type 2 response ($J \in [1, 2K]$) for K confidence levels)
- response: the type 1 response for perceived stimulus presence ($R \in \{0, 1\}$)
- confidence: the type 2 confidence response ($C \in [1, K]$)
- p_fa: the cumulative probability of a 'present'/'old' response for stimulus==0 ($P(J \geq j \mid S = 0)$)
- p_hit: the cumulative probability of a 'present'/'old' response for stimulus==1 ($P(J \geq j \mid S = 1)$)

See Also

tidybayes::epred_draws(), tidybayes::epred_rvars()

Examples

```
newdata <- tidyr::tibble(.row = 1)

# compute pseudo-type 1 ROC curve
# equivalent to ``
roc1_draws(example_model, newdata)
add_roc1_draws(newdata, example_model)

# use posterior::rvar for additional efficiency
```

```
# equivalent to `add_roc1_draws(newdata, example_model)`
roc1_rvars(example_model, newdata)

# include the ROC bounds
# equivalent to `add_roc1_draws(newdata, example_model, bounds = TRUE)`
roc1_draws(example_model, newdata, bounds = TRUE)
```

`roc2_draws` *Obtain posterior draws of the response-specific type 2 receiver operating characteristic (ROC) curves.*

Description

Given a data frame and a meta-d' model, adds estimates of the cumulative probability over confidence for each type 1 response. For `roc2_draws` and `add_roc2_draws`, estimates are returned in a tidy tibble with one row per posterior draw and per joint response. For `roc2_rvars` and `add_roc2_rvars`, parameters are returned as `posterior::rvars`, with one row per row in `newdata` and per joint response.

Usage

```
roc2_draws(object, newdata, ..., bounds = FALSE)

add_roc2_draws(newdata, object, ...)

roc2_rvars(object, newdata, ..., bounds = FALSE)

add_roc2_rvars(newdata, object, ...)
```

Arguments

`object`	The `brms` model with the `metad` family
`newdata`	A data frame from which to generate posterior predictions
`...`	Additional parameters passed to tidybayes::epred_draws
`bounds`	If `TRUE`, include the endpoints of the ROC at $(0, 0)$ and $(1, 1)$. Otherwise, the endpoints are excluded.

Value

a tibble containing posterior draws of the pseudo type 1 ROC with the following columns:

- `.row`: the row of `newdata`
- `.chain`, `.iteration`, `.draw`: for `roc2_draws` and `add_roc2_draws`, identifiers for the posterior sample
- `response`: the type 1 response for perceived stimulus presence ($R \in \{0, 1\}$)

- `confidence`: the type 2 confidence response ($C \in [1, K]$)
- `p_fa2`: the cumulative probability of an incorrect response ($P(C \geq c \mid R \neq S)$)
- `p_hit2`: the cumulative probability of a correct response ($P(C \geq c \mid R = S)$)

See Also

`tidybayes::epred_draws()`, `tidybayes::epred_rvars()`

Examples

```
newdata <- tidyr::tibble(.row = 1)

# compute type 2 ROC curve
# equivalent to `add_roc2_draws(newdata, example_model)`
roc2_draws(example_model, newdata)

# use posterior::rvar for additional efficiency
# equivalent to `add_roc2_rvars(newdata, example_model)`
roc2_rvars(example_model, newdata)

# include the ROC bounds
# equivalent to `roc2_draws(newdata, example_model, bounds = TRUE)`
roc2_draws(example_model, newdata, bounds = TRUE)
```

`sim_metad` *Simulate from the meta-d' model*

Description

Generate a simulated dataset from the meta-d' model with sensitivity `dprime`, response bias `c`, metacognitive efficiency `log_M`, and distances between confidence thresholds `c2_0_diff` and `c2_1_diff` (for the two responses).

Usage

```
sim_metad(
  N_trials = 100,
  dprime = 1,
  c = 0,
  log_M = 0,
  c2_0_diff = rep(0.5, 3),
  c2_1_diff = rep(0.5, 3),
  metac_absolute = TRUE,
  summarize = FALSE,
  lcdf = normal_lcdf,
  lccdf = normal_lccdf
)
```

Arguments

N_trials	Total number of trials to simulate. Half of these trials will have stimulus=0 and half will have stimulus=1.
dprime	The sensitivity of the signal detection agent to simulate
c	The response bias of the signal detection agent to simulate
log_M	The metacognitive efficiency of the agent on the logarithmic scale, where 0 indicates optimal metacognitive sensitivity, negative numbers indicate metacognitive inefficiency, and positive numbers indicate metacognitive hyper-efficiency.
c2_0_diff, c2_1_diff	Distances between confidence thresholds for "0" and "1" responses, such that meta_c2_0 = meta_c - cumsum(c2_0_diff) and meta_c2_1 = meta_c + cumsum(c2_1_diff).
metac_absolute	Determines how to fix the type 1 threshold for modeling confidence ratings. If metac_absolute=TRUE, meta_c = c. Otherwise, meta_c = M * c.
summarize	Aggregate the data? • If FALSE, returns a dataset with one row per observation. • If summarize=TRUE, returns an aggregated dataset where n is the number of observations per response, accuracy, and confidence level.
lcdf, lccdf	The log (complement) cumulative distribution function of the underlying signal distribution. By default, uses a normal(+/-dprime/2, 1) distribution.

Value

A simulated dataset of type 1 responses and confidence ratings, with columns:

- trial: the simulated trial number
- stimulus: the value of the stimulus on each trial (either 0 or 1)
- response: the simulated type 1 response (either 0 or 1)
- correct: whether stimulus==response (either 0 or 1)
- confidence: the simulated type 2 response (from 1 to length(c2_0_diff)+1)
- dprime:theta_2: the simulated agent's parameter values

If summarize=TRUE, the trial column is replaced with an n column indicating the number of simulated type 1/type 2 responses for each possible value.

Examples

```
sim_metad(N_trials = 10)
sim_metad(N_trials = 10000, summarize = TRUE)
sim_metad(N_trials = 10, c2_0_diff = 1, c2_1_diff = 1)
```

`sim_metad_condition` *Simulate from the meta-d' model across separate conditions*

Description

Generate a simulated dataset across separate conditions from the meta-d' model with sensitivity `dprime`, response bias `c`, metacognitive efficiency `log_M`, and distances between confidence thresholds `c2_0_diff` and `c2_1_diff` (for the two responses).

Usage

```
sim_metad_condition(
  N_trials = 100,
  dprime = rep(1, 2),
  c = rep(0, 2),
  log_M = rep(0, 2),
  c2_0_diff = list(rep(0.5, 3), rep(0.5, 3)),
  c2_1_diff = list(rep(0.5, 3), rep(0.5, 3)),
  metac_absolute = TRUE,
  summarize = FALSE,
  lcdf = normal_lcdf,
  lccdf = normal_lccdf
)
```

Arguments

`N_trials`	Total number of trials to simulate. Half of these trials will have `stimulus=0` and half will have `stimulus=1`.
`dprime`	The sensitivity of the signal detection agent to simulate
`c`	The response bias of the signal detection agent to simulate
`log_M`	The metacognitive efficiency of the agent on the logarithmic scale, where `0` indicates optimal metacognitive sensitivity, negative numbers indicate metacognitive inefficiency, and positive numbers indicate metacognitive hyper-efficiency.
`c2_0_diff`, `c2_1_diff`	Distances between confidence thresholds for `"0"` and `"1"` responses, such that `meta_c2_0 = meta_c - cumsum(c2_0_diff)` and `meta_c2_1 = meta_c + cumsum(c2_1_diff)`.
`metac_absolute`	Determines how to fix the type 1 threshold for modeling confidence ratings. If metac_absolute=TRUE, `meta_c = c`. Otherwise, `meta_c = M * c`.
`summarize`	Aggregate the data? If `summarize=FALSE`, returns a dataset with one row per observation. If `summarize=TRUE`, returns an aggregated dataset where `n` is the number of observations per response, accuracy, and confidence level.
`lcdf`, `lccdf`	The log (complement) cumulative distribution function of the underlying signal distribution. By default, uses a `normal(+/-dprime/2, 1)` distribution.

Value

A simulated dataset of type 1 responses and confidence ratings, with columns:

- `trial`: the simulated trial number
- `condition`: the simulated condition number
- `stimulus`: the value of the stimulus on each trial (either `0` or `1`)
- `response`: the simulated type 1 response (either `0` or `1`)
- `correct`: whether `stimulus==response` (either `0` or `1`)
- `confidence`: the simulated type 2 response (from `1` to `length(c2_0_diff)+1`)
- `dprime:theta_2`: the simulated agent's parameter values

If `summarize=TRUE`, the `trial` column is replaced with an `n` column indicating the number of simulated type 1/type 2 responses for each possible value.

Examples

```
sim_metad_condition(N_trials = 10)
sim_metad_condition(N_trials = 10000, summarize = TRUE)
sim_metad_condition(N_trials = 10, c2_0_diff = list(1, .5), c2_1_diff = list(1, .5))
```

---

sim_metad_participant *Simulate from the hierarchical meta-d' model*

Description

Generate a simulated dataset across participants from the meta-d' model with sensitivity `dprime`, response bias `c`, metacognitive efficiency `log_M`, and distances between confidence thresholds `c2_0_diff` and `c2_1_diff` (for the two responses).

Usage

```
sim_metad_participant(
  N_participants = 100,
  N_trials = 100,
  mu_dprime = 1,
  sd_dprime = 0.5,
  mu_c = 0,
  sd_c = 0.5,
  mu_log_M = 0,
  sd_log_M = 0.5,
  mu_z_c2_0 = rep(-1, 3),
  sd_z_c2_0 = rep(0.1, 3),
  r_z_c2_0 = diag(3),
  mu_z_c2_1 = rep(-1, 3),
  sd_z_c2_1 = rep(0.1, 3),
  r_z_c2_1 = diag(3),
```

```
  metac_absolute = TRUE,
  summarize = FALSE,
  lcdf = normal_lcdf,
  lccdf = normal_lccdf
)
```

Arguments

N_trials, N_participants	Total number of participants and trials to simulate per participant. Half of these trials will have stimulus=0 and half will have stimulus=1.
mu_dprime, sd_dprime	The mean and standard deviation of sensitivities of the signal detection agents to simulate
mu_c, sd_c	The mean and standard deviation of response bias of the signal detection agents to simulate
mu_log_M, sd_log_M	The mean and standard deviation of metacognitive efficiency of the agents on the logarithmic scale, where 0 indicates optimal metacognitive sensitivity, negative numbers indicate metacognitive inefficiency, and positive numbers indicate metacognitive hyper-efficiency.
mu_z_c2_0, mu_z_c2_1	Mean distance between confidence thresholds for "0" and "1" responses on the log_scale, such that meta_c2_0 = meta_c - cumulative_sum(exp(z_c2_0)) and meta_c2_1 = meta_c + cumulative_sum(exp(z_c2_1)).
sd_z_c2_0, sd_z_c2_1	SD of log distances between confidence thresholds for "0" and "1" responses on the log_scale.
r_z_c2_0, r_z_c2_1	Correlation of log distances between confidence thresholds for "0" and "1" responses on the log_scale.
metac_absolute	Determines how to fix the type 1 threshold for modeling confidence ratings. If metac_absolute=TRUE, meta_c = c. Otherwise, meta_c = M * c.
summarize	Aggregate the data? If summarize=FALSE, returns a dataset with one row per observation. If summarize=TRUE, returns an aggregated dataset where n is the number of observations per response, accuracy, and confidence level.
lcdf	The log cumulative distribution function of the underlying signal distribution. By default, uses a normal(+/-dprime/2, 1) distribution.
lccdf	The log complement cumulative distribution function of the underlying signal distribution. By default, uses a normal(+/-dprime/2, 1) distribution.

Value

A simulated dataset of type 1 responses and confidence ratings, with columns:

- trial: the simulated trial number
- participant: the simulated participant number

- stimulus: the value of the stimulus on each trial (either 0 or 1)
- response: the simulated type 1 response (either 0 or 1)
- correct: whether stimulus==response (either 0 or 1)
- confidence: the simulated type 2 response (from 1 to length(c2_0_diff)+1)
- dprime:theta_2: the simulated agent's parameter values

If summarize=TRUE, the trial column is replaced with an n column indicating the number of simulated type 1/type 2 responses for each possible value.

Examples

```
sim_metad_participant(N_participants = 10, N_trials = 10)
sim_metad_participant(N_participants = 25, mu_dprime = 2, mu_log_M = -1)
```

---

sim_metad_participant_condition

Simulate from the hierarchical meta-d' model across within-participant conditions

---

Description

Generate a simulated dataset across participants and conditions from the meta-d' model with sensitivity dprime, response bias c, metacognitive efficiency log_M, and distances between confidence thresholds c2_0_diff and c2_1_diff (for the two responses).

Usage

```
sim_metad_participant_condition(
  N_participants = 100,
  N_trials = 100,
  mu_dprime = rep(1, 2),
  sd_dprime = rep(0.5, 2),
  r_dprime = diag(2),
  mu_c = rep(0, 2),
  sd_c = rep(0.5, 2),
  r_c = diag(2),
  mu_log_M = rep(0, 2),
  sd_log_M = rep(0.5, 2),
  r_log_M = diag(2),
  mu_z_c2_0 = matrix(rep(-1, 6), nrow = 3, ncol = 2),
  sd_z_c2_0_condition = rep(0.1, 2),
  r_z_c2_0_condition = diag(2),
  sd_z_c2_0_confidence = rep(0.1, 3),
  r_z_c2_0_confidence = diag(3),
  mu_z_c2_1 = matrix(rep(-1, 6), nrow = 3, ncol = 2),
  sd_z_c2_1_condition = rep(0.1, 2),
```

```
  r_z_c2_1_condition = diag(2),
  sd_z_c2_1_confidence = rep(0.1, 3),
  r_z_c2_1_confidence = diag(3),
  metac_absolute = TRUE,
  summarize = FALSE,
  lcdf = normal_lcdf,
  lccdf = normal_lccdf
)
```

Arguments

`N_trials`, `N_participants`
: Total number of participants and trials to simulate per participant. Half of these trials will have `stimulus=0` and half will have `stimulus=1`.

`mu_dprime`, `sd_dprime`, `r_dprime`
: The mean, standard deviation, and within-participant correlations of sensitivities of the signal detection agents to simulate

`mu_c`, `sd_c`, `r_c`
: The mean, standard deviation, and within-participant correlations of response bias of the signal detection agents to simulate

`mu_log_M`, `sd_log_M`, `r_log_M`
: The mean, standard deviation, and within-participant correlations of metacognitive efficiency of the agents on the logarithmic scale, where `0` indicates optimal metacognitive sensitivity, negative numbers indicate metacognitive inefficiency, and positive numbers indicate metacognitive hyper-efficiency.

`mu_z_c2_0`, `mu_z_c2_1`
: Mean distance between confidence thresholds for `"0"` and `"1"` responses on the log_scale, such that `meta_c2_0 = meta_c - cumulative_sum(exp(z_c2_0))` and `meta_c2_1 = meta_c + cumulative_sum(exp(z_c2_1))`.

`sd_z_c2_0_condition`, `sd_z_c2_1_condition`
: SD of log distances across conditions between confidence thresholds for `"0"` and `"1"` responses on the log_scale.

`r_z_c2_0_condition`, `r_z_c2_1_condition`
: Correlation across conditions of log distances between confidence thresholds for `"0"` and `"1"` responses on the log_scale.

`sd_z_c2_0_confidence`, `sd_z_c2_1_confidence`
: SD of log distances across confidence levels between confidence thresholds for `"0"` and `"1"` responses on the log_scale.

`r_z_c2_0_confidence`, `r_z_c2_1_confidence`
: Correlation across confidence levels of log distances between confidence thresholds for `"0"` and `"1"` responses on the log_scale.

`metac_absolute`
: Determines how to fix the type 1 threshold for modeling confidence ratings. If metac_absolute=TRUE, `meta_c = c`. Otherwise, `meta_c = M * c`.

`summarize`
: Aggregate the data? If summarize=FALSE, returns a dataset with one row per observation. If summarize=TRUE, returns an aggregated dataset where `n` is the number of observations per response, accuracy, and confidence level.

`lcdf`
: The log cumulative distribution function of the underlying signal distribution. By default, uses a `normal(+/- dprime/2, 1)` distribution.

`lccdf` The log complement cumulative distribution function of the underlying signal distribution. By default, uses a `normal(+/- dprime/2, 1)` distribution.

Value

A simulated dataset of type 1 responses and confidence ratings, with columns:

- `trial`: the simulated trial number
- `stimulus`: the value of the stimulus on each trial (either `0` or `1`)
- `response`: the simulated type 1 response (either `0` or `1`)
- `correct`: whether `stimulus==response` (either `0` or `1`)
- `confidence`: the simulated type 2 response (from `1` to `length(c2_0_diff)+1`)
- `dprime:theta_2`: the simulated agent's parameter values If `summarize=TRUE`, the `trial` column is replaced with an `n` column indicating the number of simulated type 1/type 2 responses for each possible value.

Examples

```
sim_metad_participant_condition(10, 10)
```

---

stanvars_metad *Generate Stan code for the meta-d' model*

Description

Generate Stan code for the meta-d' model

Usage

```
stanvars_metad(
  K,
  distribution = "normal",
  metac_absolute = TRUE,
  categorical = FALSE
)
```

Arguments

`K` The number of confidence levels

`distribution` The noise distribution to use. Should be a parameter-free distribution, i.e., one that is mean-centered without additional variance/shape parameters. If the distribution is not already available in stan, you must additionally provide two functions to Stan (one for `<distribution>_lcdf` and one for `<distribution>_lccdf`).

`metac_absolute` Should the type 2 criterion (metac) be fixed to the absolute type 1 criterion (c)? If `TRUE`, the model will set `metac = c`. Otherwise, it will set `metac = M * c`, such that the type 2 criterion is *relatively* equal to the type 1 criterion (i.e., `meta_c/meta_dprime = c/dprime`)

`categorical` If `FALSE` (default), use the multinomial likelihood over aggregated data. If `TRUE`, use the categorical likelihood over individual trials.

Value

A brms::stanvar object containing Stan code defining the likelihood for the metad' model with `K` confidence levels, signal distributed according to the distribution `distribution`, and where `metac = c` if `metac_absolute==TRUE`, and `metac = M*c` otherwise.

Examples

```
# create stancode for the meta-d' model
# using the normal distribution and 3 levels of confidence
stanvars_metad(3)

# create stancode for the meta-d' model with meta_c = M * c
stanvars_metad(3, metac_absolute = FALSE)

# create stancode for the meta-d' model with
# an alternative distribution
# note: cumulative distribution functions must be defined
# in R and in Stan using [brms::stanvar()]
stanvars_metad(4, distribution = "gumbel_min")
```

---

to_signed *Convert binary variable x between $\{0, 1\}$ and $\{-1, 1\}$*

Description

- `to_signed(x)` converts $x \in \{0, 1\}$ to $x' \in \{-1, 1\}$
- `to_unsigned(x)` converts $x \in \{-1, 1\}$ to $x' \in \{0, 1\}$

Usage

```
to_signed(x)

to_unsigned(x)
```

Arguments

`x` A binary variable

Value

A signed (for `to_signed`) or unsigned (for `to_unsigned`) version of `x`

Examples

```
# should return `1`
to_signed(0)

# should return `1`
to_signed(1)

# should return `0`
to_unsigned(-1)

# should return `1`
to_unsigned(1)

# `to_signed` also works with objects `R` interprets as `0` or `1`
to_signed(10)

# `to_unsigned` also works with any signed integer
to_unsigned(-10)

# neither function works with factors
try(to_signed(factor(1)))
tryCatch(to_unsigned(factor(1)), warning=function(w) w)
```

Index

∗ **datasets**
    example_data, 6
    example_model, 7

add_epred_draws_metad
        *(*epred_draws_metad*)*, 5
add_epred_rvars_metad
        *(*epred_draws_metad*)*, 5
add_linpred_draws_metad
        *(*linpred_draws_metad*)*, 11
add_linpred_rvars_metad
        *(*linpred_draws_metad*)*, 11
add_mean_confidence_draws
        *(*mean_confidence_draws*)*, 12
add_mean_confidence_rvars
        *(*mean_confidence_draws*)*, 12
add_metacognitive_bias_draws
        *(*metacognitive_bias_draws*)*, 14
add_metacognitive_bias_rvars
        *(*metacognitive_bias_draws*)*, 14
add_predicted_draws_metad
        *(*predicted_draws_metad*)*, 19
add_predicted_rvars_metad
        *(*predicted_draws_metad*)*, 19
add_roc1_draws *(*roc1_draws*)*, 21
add_roc1_rvars *(*roc1_draws*)*, 21
add_roc2_draws *(*roc2_draws*)*, 23
add_roc2_rvars *(*roc2_draws*)*, 23
aggregate_metad, 2, *20*
aggregate_metad(), *9*

brms::brm(), *8*
brms::stanvar, *32*
brms::stanvar(), *9*

cor_matrix, 4
cov_matrix, 5

epred_draws_metad, 5
epred_rvars_metad *(*epred_draws_metad*)*, 5
example_data, 6, *7*
example_model, 7

fit_metad, 8
fit_metad(), *8*

joint_response, 10

linpred_draws_metad, 11
linpred_rvars_metad
        *(*linpred_draws_metad*)*, 11

mean_confidence_draws, 12
mean_confidence_rvars
        *(*mean_confidence_draws*)*, 12
metacognitive_bias_draws, 14
metacognitive_bias_rvars
        *(*metacognitive_bias_draws*)*, 14
metad, 16
metad_pmf, 17

normal_lccdf *(*normal_lcdf*)*, 18
normal_lcdf, 18

posterior::rvar, *5*, *11*, *12*, *14*, *19*, *21*
predicted_draws_metad, 19
predicted_rvars_metad
        *(*predicted_draws_metad*)*, 19

response_probabilities, 20
rmatrixnorm, 21
roc1_draws, 21
roc1_rvars *(*roc1_draws*)*, 21
roc2_draws, 23
roc2_rvars *(*roc2_draws*)*, 23

sim_metad, 24
sim_metad(), *7*
sim_metad_condition, 26
sim_metad_participant, 27
sim_metad_participant_condition, 29

stanvars_metad, 31

tidybayes::add_epred_draws, *6*
tidybayes::add_epred_rvars, *6*
tidybayes::add_linpred_draws, *11*
tidybayes::add_linpred_rvars, *11*
tidybayes::add_predicted_draws, *19*
tidybayes::add_predicted_rvars, *19*
tidybayes::epred_draws, *13*, *15*, *22, 23*
tidybayes::epred_draws(), *6*, *14*, *22*, *24*
tidybayes::epred_rvars, *13*, *15*, *22*
tidybayes::epred_rvars(), *6*, *14*, *22*, *24*
tidybayes::linpred_draws(), *12*, *15*
tidybayes::linpred_rvars(), *12*, *15*
tidybayes::predicted_draws(), *19*
tidybayes::predicted_rvars(), *19*
to_signed, 32
to_unsigned *(*to_signed*)*, 32
type1_response *(*joint_response*)*, 10
type2_response *(*joint_response*)*, 10