Package 'easyRaschBayes'

April 23, 2026

Title Bayesian Rasch Analysis Using 'brms'

Version 0.2.0.1

Description Reproduces classic Rasch psychometric analysis features using Bayesian item response theory models fitted with 'brms' following Bürkner (2021) <doi:10.18637/jss.v100.i05> and Bürkner (2020) <doi:10.3390/jintelligence8010005>. Supports both dichotomous and polytomous Rasch models. Features include posterior predictive item fit, conditional infit, item-restscore associations, person fit, differential item functioning, local dependence assessment via Q3 residual correlations, dimensionality assessment with residual principal components analysis, person-item targeting plots, item category probability curves, and reliability using relative measurement uncertainty following Bignardi et al. (2025) <doi:10.31234/osf.io/h54k8_v1>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports brms (>= 2.20.0), rlang (>= 1.0.0), dplyr (>= 1.1.0), tidyr (>= 1.3.0), tibble (>= 3.0.0), ggdist, stats, grDevices, ggplot2 (>= 3.4.0), forcats

Suggests ggrepel, patchwork, eRm, testthat (>= 3.0.0), knitr, rmarkdown

URL https://github.com/pgmj/easyRaschBayes, https://pgmj.github.io/easyRaschBayes/

BugReports https://github.com/pgmj/easyRaschBayes/issues

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Magnus Johansson [aut, cre] (ORCID: <https://orcid.org/0000-0003-1669-592X>), Giacomo Bignardi [ctb] (RMU reliability code)

Maintainer Magnus Johansson <pgmj@pm.me>

Repository CRAN

Date/Publication 2026-04-23 07:10:02 UTC

Contents

`dif_statistic` *Differential Item Functioning (DIF) Analysis for Bayesian IRT Models*

Description

Tests for differential item functioning (DIF) in Bayesian Rasch-family models fitted with **brms** by comparing item parameters across subgroups defined by an exogenous variable. The function fits a DIF model that includes group-by-item interactions and summarizes the posterior distribution of the DIF effects.

Usage

```
dif_statistic(
  model,
  group_var,
  item_var = item,
  person_var = id,
```

```
  data = NULL,
  dif_type = c("uniform", "non-uniform"),
  prob = 0.95,
  rope = 0.5,
  refit = TRUE,
  ...
)
```

Arguments

model	A fitted brmsfit object from the baseline (no-DIF) model.
group_var	An unquoted variable name identifying the grouping variable for DIF testing (e.g., gender). Must be a factor or character variable with exactly 2 levels in the current implementation.
item_var	An unquoted variable name identifying the item grouping variable. Default is item.
person_var	An unquoted variable name identifying the person grouping variable. Default is id.
data	An optional data frame containing all variables needed for the DIF model, including the group variable. If NULL (the default), the function attempts to use model$data. Since the baseline model formula typically does not include the group variable, brms will have dropped it from the stored model data. In that case, you must supply the original data frame here.
dif_type	Character. For polytomous ordinal models only. "uniform" (the default) tests for a uniform location shift per item via a group:item fixed-effect interaction. "non-uniform" fits group-specific thresholds per item and computes per-threshold DIF effects as the difference between groups. Ignored for dichotomous models.
prob	Numeric in $(0, 1)$. Width of the credible intervals. Default is 0.95.
rope	Numeric. Half-width of the Region of Practical Equivalence (ROPE) around zero for DIF effects, on the logit scale. Default is 0.5, corresponding to a practically negligible DIF effect. Set to 0 to skip ROPE analysis.
refit	Logical. If TRUE (the default), the DIF model is fitted automatically by updating the baseline model via update, which reuses the compiled Stan code for faster sampling. If FALSE, only the DIF model formula is returned (useful for manual fitting with custom settings).
...	Additional arguments passed to update.brmsfit when refitting the DIF model (e.g., cores, control).

Details

For polytomous models, two types of DIF can be tested:

Uniform DIF (dif_type = "uniform"**, default)** A single location shift per item across groups, modelled as a group:item fixed-effect interaction. This tests whether the average item difficulty differs between groups.

Non-uniform / threshold-level DIF (`dif_type = "non-uniform"`) Each item receives group-specific thresholds via `thres(gr = interaction(item, group))`. DIF effects are computed as the difference in each threshold between groups, revealing whether DIF affects specific response categories.

The function constructs a DIF model by adding a group-by-item interaction to the baseline model:

- **Dichotomous models** (`family = bernoulli()`): The baseline `response ~ 1 + (1 | item) + (1 | id)` becomes `response ~ 1 + group + (1 + group | item) + (1 | id)`, where the group slope varying by item captures item-specific DIF.
- **Polytomous uniform DIF** (`dif_type = "uniform"`): The baseline `response | thres(gr = item) ~ 1 + (1 | id)` becomes `response | thres(gr = item) ~ 1 + group:item + (1 | id)`.
- **Polytomous non-uniform DIF** (`dif_type = "non-uniform"`): The baseline becomes `response | thres(gr = item_group) ~ 1 + (1 | id)`, where `item_group = interaction(item, group)`. Each item × group combination gets its own thresholds. DIF effects are the differences between group-specific thresholds for each item, computed draw-by-draw from the posterior.

DIF effects are summarized using:

Probability of Direction (pd) The proportion of the posterior on the dominant side of zero. Values > 0.975 indicate strong directional evidence.

ROPE The Region of Practical Equivalence (Kruschke, 2018). If > 95\ the DIF effect is practically negligible. If > 95\ outside, the effect is practically significant.

Credible Interval If the CI excludes zero, there is evidence of DIF at the specified credibility level.

Value

A list with the following elements:

summary A `tibble` with one row per item (for uniform DIF) or per item × threshold (for non-uniform DIF) containing: `item`, optionally `threshold`, `dif_estimate` (posterior mean), `dif_lower`, `dif_upper` (credible interval), `dif_sd` (posterior SD), `pd` (probability of direction), `rope_percentage` (proportion inside ROPE), and `flag` (classification).

dif_draws A matrix of posterior draws for the DIF effects (draws × effects), for further analysis.

dif_model The fitted DIF `brmsfit` object (if `refit = TRUE`), or `NULL`.

dif_formula The `brmsformula` used for the DIF model.

baseline_model The original baseline model.

plot A `ggplot` forest plot of DIF effects with credible intervals and ROPE.

References

Kruschke, J. K. (2018). Rejecting or accepting parameter values in Bayesian estimation. *Advances in Methods and Practices in Psychological Science*, *1*(2), 270–280.

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

infit_statistic for item fit, q3_statistic for local dependence, brm, hypothesis.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Dichotomous Rasch with DIF testing ---

set.seed(123)
df <- expand.grid(id = 1:200, item = paste0("I", 1:10)) %>%
  mutate(
    gender = rep(sample(c("M", "F"), 200, TRUE), each = 10),
    theta  = rep(rnorm(200), each = 10),
    delta  = rep(seq(-2, 2, length.out = 10), 200),
    dif    = ifelse(item == "I3" & gender == "F", 1.0,
             ifelse(item == "I7" & gender == "F", -0.8, 0)),
    p      = plogis(theta - delta - dif),
    response = rbinom(n(), 1, p)
  )

fit_base <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df,
  family = bernoulli(),
  chains = 4,
  cores  = 2, # use more cores if you have
  iter   = 1000 # use at least 2000
)

dif_result <- dif_statistic(
  model     = fit_base,
  group_var = gender,
  data      = df
)

dif_result$summary
dif_result$plot

# --- Partial Credit Model: uniform DIF ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  mutate(gender = sample(c("M", "F"), n(), TRUE)) %>%
  pivot_longer(!c(id, gender),
               names_to = "item", values_to = "response")

fit_pcm <- brm(
```

```
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 2, # use more cores if you have
  iter   = 1000 # use at least 2000
)

# Uniform DIF (default): one shift per item
dif_uni <- dif_statistic(fit_pcm, group_var = gender, data = df_pcm)
dif_uni$plot

# Non-uniform DIF: threshold-level effects
dif_nu <- dif_statistic(fit_pcm, group_var = gender, data = df_pcm,
                        dif_type = "non-uniform")
dif_nu$summary
dif_nu$plot
```

`fit_statistic_pcm` *Posterior Predictive Item Fit Statistic for Bayesian IRT Models*

Description

Computes posterior predictive item (or person) fit statistics for Bayesian IRT models fitted with **brms**. For each posterior draw, observed and replicated data are compared via a user-supplied criterion function, grouped by item, person, or any other variable. Posterior predictive p-values can then be derived from the output to assess fit.

Usage

```
fit_statistic_pcm(model, criterion, group, ndraws_use = NULL)
```

Arguments

`model`	A fitted `brmsfit` object.
`criterion`	A function with signature `function(y, p)` that computes a pointwise fit criterion. For ordinal and categorical models, `y` is the observed (or replicated) response category and `p` is the model-predicted probability of that category. For binary models, `y` is the binary response and `p` is the predicted probability of success.
`group`	An unquoted variable name (e.g., `item` or `id`) indicating the grouping variable over which the fit statistic is aggregated. Typically `item` for item fit or `id` for person fit.
`ndraws_use`	Optional positive integer. If specified, a random subset of posterior draws of this size is used, which can speed up computation for large models. If `NULL` (the default), all draws are used.

Details

The function implements the posterior predictive checking approach for item fit described in Bürkner (2020). The procedure works as follows:

1. Draw posterior expected category probabilities via `posterior_epred` and posterior predicted responses via `posterior_predict`.
2. For ordinal or categorical models (3D array output from `posterior_epred`), extract the probability assigned to the observed response category and to the replicated response category for each draw and observation.
3. Apply the user-supplied `criterion` function to compute pointwise fit values for both observed and replicated data.
4. Aggregate (sum) the criterion values within each level of `group` and each posterior draw.

A common choice for ordinal IRT models is the categorical log-likelihood criterion `function(y, p) log(p)`. For binary (e.g., dichotomous Rasch) models, the Bernoulli log-likelihood `function(y, p) y * log(p) + (1 - y) * log(1 - p)` may be used instead.

Value

A `tibble` with the following columns:

`group` The grouping variable (e.g., item name or person id).

draw Integer index of the posterior draw.

crit The observed fit statistic (criterion applied to observed data) summed within each group and draw.

crit_rep The replicated fit statistic (criterion applied to posterior predicted data) summed within each group and draw.

crit_diff The difference `crit_rep - crit`.

The output is grouped by the grouping variable. Posterior predictive p-values can be obtained by computing `mean(crit_rep > crit)` within each group.

References

Bürkner, P.-C. (2020). Analysing Standard Progressive Matrices (SPM-LS) with Bayesian Item Response Models. *Journal of Intelligence*, *8*(1). doi:10.3390/jintelligence8010005

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`fit_statistic_rm` for dichotomous Rasch models, `posterior_epred` for expected predictions, `posterior_predict` for posterior predictive samples, `pp_check` for graphical posterior predictive checks.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Polytomous Rasch (Partial Credit Model) ---

# Prepare data in long format
df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

# Fit a Partial Credit Model using the adjacent category family
fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data    = df_pcm,
  family  = acat,
  chains  = 4,
  cores   = 1, # use more cores if you have
  iter    = 500 # use at least 2000
)

# Categorical log-likelihood criterion (for polytomous models)
ll_categorical <- function(y, p) log(p)

# Compute item fit statistics
item_fit <- fit_statistic_pcm(
  model      = fit_pcm,
  criterion  = ll_categorical,
  group      = item,
  ndraws_use = 100 # use at least 500
)

# Summarise: posterior predictive p-values per item
item_fit %>%
  group_by(item) %>%
  summarise(
    observed   = mean(crit),
    replicated = mean(crit_rep),
    ppp        = mean(crit_rep > crit)
  )

# Use ggplot2 to make a histogram
library(ggplot2)
item_fit %>%
  ggplot(aes(crit_diff)) +
  geom_histogram(aes(fill = ifelse(crit_diff > 0, "above","below"))) +
  facet_wrap("item") +
  theme_bw() +
  theme(legend.position = "none")
```

```
# Compute person fit statistics
person_fit <- fit_statistic_pcm(
  model      = fit_pcm,
  criterion  = ll_categorical,
  group      = id,
  ndraws_use = 100 # use at least 500
)
```

`fit_statistic_rm` *Posterior Predictive Item Fit Statistic for Binary Bayesian IRT Models*

Description

Computes posterior predictive item (or person) fit statistics for dichotomous Bayesian IRT models fitted with **brms**. For each posterior draw, observed and replicated data are compared via a user-supplied criterion function, grouped by item, person, or any other variable. Posterior predictive p-values can then be derived from the output to assess fit.

Usage

```
fit_statistic_rm(model, criterion, group, ndraws_use = NULL)
```

Arguments

`model`	A fitted `brmsfit` object with a binary response (e.g., `family = bernoulli()`).
`criterion`	A function with signature `function(y, p)` that computes a pointwise fit criterion, where `y` is the binary response (0 or 1) and `p` is the predicted probability of success. A common choice is the Bernoulli log-likelihood: `function(y, p) y * log(p) + (1 - y) * log(1 - p)`.
`group`	An unquoted variable name (e.g., `item` or `id`) indicating the grouping variable over which the fit statistic is aggregated. Typically `item` for item fit or `id` for person fit.
`ndraws_use`	Optional positive integer. If specified, a random subset of posterior draws of this size is used, which can speed up computation for large models. If `NULL` (the default), all draws are used.

Details

This function is the binary-response counterpart of `fit_statistic_pcm`, which handles polytomous (ordinal / categorical) models. For dichotomous models, `posterior_epred()` returns a 2D matrix (S x N) of success probabilities, so the criterion function receives the observed binary response and the corresponding probability directly.

The procedure follows the posterior predictive checking approach described in Bürkner (2020):

1. Draw posterior expected success probabilities via `posterior_epred` and posterior predicted binary responses via `posterior_predict`.
2. Apply the user-supplied `criterion` function pointwise to both observed and replicated data paired with the predicted probabilities.
3. Aggregate (sum) the criterion values within each level of `group` and each posterior draw.

The standard criterion for binary models is the Bernoulli log-likelihood:

$$\ell(y, p) = y \log(p) + (1 - y) \log(1 - p).$$

Value

A `tibble` with the following columns:

`group` The grouping variable (e.g., item name or person id).

draw Integer index of the posterior draw.

crit The observed fit statistic (criterion applied to observed data) summed within each group and draw.

crit_rep The replicated fit statistic (criterion applied to posterior predicted data) summed within each group and draw.

crit_diff The difference `crit_rep - crit`.

The output is grouped by the grouping variable. Posterior predictive p-values can be obtained by computing `mean(crit_rep > crit)` within each group.

References

Bürkner, P.-C. (2020). Analysing Standard Progressive Matrices (SPM-LS) with Bayesian Item Response Models. *Journal of Intelligence*, *8*(1). doi:10.3390/jintelligence8010005

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`fit_statistic_pcm` for polytomous (ordinal/categorical) models, `posterior_epred` for expected predictions, `posterior_predict` for posterior predictive samples, `pp_check` for graphical posterior predictive checks.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Dichotomous Rasch Model ---

# Prepare binary response data in long format
df_rm <- eRm::raschdat3 %>%
```

```
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

# Fit a dichotomous Rasch model
fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores   = 1, # use more cores if you have
  iter    = 500 # use at least 2000
)

# Bernoulli log-likelihood criterion
ll_bernoulli <- function(y, p) y * log(p) + (1 - y) * log(1 - p)

# Compute item fit statistics
item_fit <- fit_statistic_rm(
  model      = fit_rm,
  criterion  = ll_bernoulli,
  group      = item,
  ndraws_use = 100 # use at least 500
)

# Summarise: posterior predictive p-values per item
item_fit %>%
  group_by(item) %>%
  summarise(
    observed   = mean(crit),
    replicated = mean(crit_rep),
    ppp        = mean(crit_rep > crit)
  )

# Use ggplot2 to make a histogram
library(ggplot2)
item_fit %>%
  ggplot(aes(crit_diff)) +
  geom_histogram(aes(fill = ifelse(crit_diff > 0, "above","below"))) +
  facet_wrap("item") +
  theme_bw() +
  theme(legend.position = "none")

# Compute person fit statistics
person_fit <- fit_statistic_rm(
  model      = fit_rm,
  criterion  = ll_bernoulli,
  group      = id,
  ndraws_use = 100 # use at least 500
)

person_fit %>%
  group_by(id) %>%
```

```
  summarise(
    observed   = mean(crit),
    replicated = mean(crit_rep),
    ppp        = mean(crit_rep > crit)
  )

# --- 1PL model with item-specific intercepts ---

# Alternative parameterisation with fixed item effects
fit_1pl <- brm(
  response ~ 0 + item + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores   = 1, # use more cores if you have
  iter    = 500 # use at least 2000
)

item_fit_1pl <- fit_statistic_rm(
  model      = fit_1pl,
  criterion  = ll_bernoulli,
  group      = item,
  ndraws_use = 100 # use at least 500
)

item_fit_1pl %>%
  group_by(item) %>%
  summarise(
    observed   = mean(crit),
    replicated = mean(crit_rep),
    ppp        = mean(crit_rep > crit)
  )
```

`infit_post` *Summarize and Plot Posterior Predictive Infit Statistics*

Description

Postprocesses the output of `infit_statistic` to produce summary tables of posterior predictive infit statistics and a combined slab + interval plot comparing observed infit values to the posterior predictive distribution.

Usage

```
infit_post(infit_draws, ci = 0.84)
```

Arguments

infit_draws	A data frame (or tibble) as returned by infit_statistic containing at minimum the columns item, infit (observed infit per draw), and infit_rep (replicated infit per draw).
ci	Numeric in $(0, 1)$. Width of the credible interval used for the posterior predictive HDI and the slab display. Default is 0.84.

Details

Two complementary summary tables are provided:

summary Reports the posterior mean observed and replicated infit values alongside the posterior predictive p-value (ppp). The ppp is the proportion of draws where the replicated infit exceeds the observed infit. Under good fit, the ppp should be near 0.5. A ppp near 0 indicates the observed infit is consistently larger than expected (underfit); a ppp near 1 indicates it is consistently smaller (overfit).

hdi Reports the probability that the observed infit falls above (underfit) or below (overfit) the HDI of the replicated distribution. This provides a more distributional assessment than the ppp alone.

The plot uses two layers from the **ggdist** package:

stat_slab Displays the posterior predictive (replicated) infit distribution as a filled density slab per item, shaded by credible interval level.

stat_slabinterval Displays the observed infit distribution per item as a semi-transparent slab with point and interval summaries.

Under good model fit, the observed infit distribution should overlap substantially with the replicated distribution. Items where the observed distribution sits systematically above the replicated HDI indicate underfit (more variation than expected); items below indicate overfit (less variation than expected).

Value

A list with three elements:

summary A tibble with one row per item containing: item, infit_obs (posterior mean of observed infit), infit_rep (posterior mean of replicated infit), and infit_ppp (posterior predictive p-value: proportion of draws where the replicated infit exceeds the observed infit). Values near 0.5 indicate good fit; values near 0 suggest underfit; values near 1 suggest overfit.

hdi A tibble with one row per item containing: item, underfit (posterior probability that the observed infit exceeds the upper HDI bound of the replicated distribution), and overfit (posterior probability that the observed infit falls below the lower HDI bound).

plot A ggplot object showing the posterior predictive distribution of replicated infit (grey filled slab) overlaid with the observed infit distribution (coloured slab + interval), with a dashed reference line at 1 (perfect fit).

See Also

infit_statistic, item_restscore_post.

Examples

```
## Not run:
library(brms)
library(ggplot2)

# Assuming fit_pcm is a fitted brmsfit object
infit_draws <- infit_statistic(fit_pcm)

result <- infit_post(infit_draws)
result$summary
result$hdi
result$plot

## End(Not run)
```

---

infit_statistic *Posterior Predictive Infit Statistic for Bayesian IRT Models*

Description

Computes a Bayesian analogue of the conditional item infit statistic (as described in Christensen, Kreiner & Mesbah, 2013) for Rasch-family models fitted with **brms**. For each posterior draw, expected values and variances are derived from the category probabilities returned by posterior_epred, and variance-weighted standardised residuals are computed for both observed and replicated data. The result can be summarised into posterior predictive p-values to assess item fit.

Usage

```
infit_statistic(
  model,
  item_var = item,
  person_var = id,
  ndraws_use = NULL,
  outfit = FALSE
)
```

Arguments

model	A fitted brmsfit object from an ordinal IRT model (e.g., family = acat for a partial credit model or family = bernoulli() for a dichotomous Rasch model).
item_var	An unquoted variable name identifying the item grouping variable in the model data (e.g., item).

person_var	An unquoted variable name identifying the person grouping variable in the model data (e.g., id).
ndraws_use	Optional positive integer. If specified, a random subset of posterior draws of this size is used. If NULL (the default), all draws are used.
outfit	Logical. If TRUE, outfit statistics are computed alongside infit. Default is FALSE (infit only), since outfit is highly sensitive to outliers and rarely recommended for Rasch diagnostics.

Details

The procedure adapts the conditional infit/outfit statistics (Christensen et al., 2013; Kreiner & Christensen, 2011; Müller, 2020) to the Bayesian framework:

1. For each posterior draw s, category probabilities $P^{(s)}(X_{vi} = c)$ are obtained from posterior_epred.
2. The conditional expected value and variance for each observation are computed as:

$$E_{vi}^{(s)} = \sum_c c \cdot P^{(s)}(X_{vi} = c)$$

$$Var_{vi}^{(s)} = \sum_c (c - E_{vi}^{(s)})^2 \cdot P^{(s)}(X_{vi} = c)$$

3. Standardised squared residuals are:

$$Z_{vi}^{2(s)} = (X_{vi} - E_{vi}^{(s)})^2 / Var_{vi}^{(s)}$$

4. The infit statistic for item i is the variance-weighted mean of Z^2 across persons:

$$Infit_i^{(s)} = \frac{\sum_v Var_{vi}^{(s)} Z_{vi}^{2(s)}}{\sum_v Var_{vi}^{(s)}}$$

5. If requested, the outfit is the unweighted mean of Z^2.

Value

A tibble with the following columns:

item The item identifier.

draw Integer index of the posterior draw.

infit The observed infit statistic for that item and draw.

infit_rep The replicated infit statistic (based on posterior predicted data) for that item and draw.

outfit (Only if outfit = TRUE) The observed outfit statistic for that item and draw.

outfit_rep (Only if outfit = TRUE) The replicated outfit statistic for that item and draw.

The output is grouped by the item variable. Posterior predictive p-values can be obtained by computing, e.g., mean(infit_rep > infit) within each item.

References

Christensen, K. B., Kreiner, S. & Mesbah, M. (Eds.) (2013). *Rasch Models in Health*. Iste and Wiley, pp. 86–90.

Kreiner, S. & Christensen, K. B. (2011). Exact evaluation of Bias in Rasch model residuals. *Advances in Mathematics Research*, 12, 19–40.

Müller, M. (2020). Item fit statistics for Rasch analysis: can we trust them? *Journal of Statistical Distributions and Applications*, *7*(1). doi:10.1186/s40488020001087

See Also

`fit_statistic_pcm` for a general-purpose posterior predictive fit statistic with user-supplied criterion functions, `fit_statistic_rm` for a general-purpose posterior predictive fit statistic with user-supplied criterion functions, `posterior_epred`, `posterior_predict`.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Partial Credit Model (polytomous) ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

# Compute infit per item
item_infit <- infit_statistic(
  model      = fit_pcm,
  ndraws_use = 100 # use at least 500
)

# Post-process draws
infit_results <- infit_post(item_infit)
infit_results$summary
infit_results$hdi
infit_results$plot

# --- Dichotomous Rasch Model ---
```

```
df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

item_infit_rm <- infit_statistic(
  model      = fit_rm,
  ndraws_use = 100 # use at least 500
)

# Post-process draws
infit_results <- infit_post(item_infit_rm)
infit_results$summary
infit_results$hdi
infit_results$plot
```

`item_parameters` *Extract Item Parameters from a Bayesian Rasch Model*

Description

Extracts item difficulty (threshold) parameters from a fitted Bayesian Rasch model. Returns a simple location table in both long and wide formats, a full summary with posterior SEs and HDCIs, item-level information, threshold ordering diagnostics, and optionally the full posterior draws matrix.

Usage

```
item_parameters(
  model,
  item_var = item,
  person_var = id,
  draws = FALSE,
  center = TRUE,
  prob = 0.95
)
```

Arguments

model	A fitted brmsfit object. Supported parameterisations: **Polytomous ordinal (PCM)** e.g., family = acat with thres(gr = item), producing item-specific thresholds. **Dichotomous Rasch (random items)** e.g., response ~ 1 + (1 | item) + (1 | id) with family = bernoulli(). **Dichotomous 1PL (fixed items)** e.g., response ~ 0 + item + (1 | id) with family = bernoulli().
item_var	An unquoted variable name identifying the item grouping variable in the model data. Default is item.
person_var	An unquoted variable name identifying the person grouping variable in the model data. Default is id.
draws	Logical. If TRUE, a draws matrix of full posterior draws is included in the output. Default is FALSE.
center	Logical. If TRUE (the default), item parameters are shifted so that the grand mean of all threshold locations is zero, matching the frequentist CML convention and the centering used in plot_targeting.
prob	Numeric in $(0, 1)$. Width of the highest density continuous interval (HDCI) reported in the summary. Default is 0.95.

Details

Dichotomous models with random item effects (response ~ 1 + (1 | item) + (1 | id)) parameterise item difficulty as $\delta_i = -(b_0 + r_i)$ where b_0 is the global intercept and r_i is the item random effect. Models with fixed item effects (response ~ 0 + item + (1 | id)) parameterise difficulty as $\delta_i = -b_i$.

Polytomous (acat/PCM) models with grouped thresholds (thres(gr = item)) directly estimate item-specific threshold parameters. Each row in the long output represents one threshold within one item; each row in the wide output represents one item.

Item information is computed from the posterior mean item parameters using the standard Rasch/PCM information formulae:

Dichotomous $I_i(\theta) = P_i(\theta)Q_i(\theta)$ where $P_i = \text{logistic}(\theta - \delta_i)$.

Polytomous (PCM) $I_i(\theta) = \sum_c (c - E_i)^2 P_{ic}(\theta)$ where $E_i = \sum_c c \cdot P_{ic}$ is the expected score for item i.

Threshold ordering: In the partial credit model, disordered thresholds ($\tau_{k+1} \leq \tau_k$) indicate that the probability of responding in the intermediate category never exceeds both adjacent categories — the category is empirically "absorbed". This does not necessarily indicate misfit (see Adams et al., 2012), but may suggest response categories should be collapsed. The prob_disordered column reports the posterior probability of at least one disordered pair per item, providing a Bayesian alternative to post-hoc threshold checks.

Value

A list with the following elements:

locations A `tibble` in long format with one row per item (dichotomous) or per item-threshold (polytomous), containing `item`, `threshold` (integer, always 1 for dichotomous), and `location` (posterior mean on the logit scale).

locations_wide A `tibble` in wide format with one row per item, sorted by mean location. For polytomous models, threshold columns are named `t1`, `t2`, etc., and a `location` column gives the mean across thresholds. For dichotomous models, only `item` and `location` columns are present.

summary A `tibble` extending the long `locations` with: `se` (posterior SD), `hdci_lower` and `hdci_upper` (highest density continuous interval bounds at the level specified by `prob`), and `n_eff` (effective sample size for the parameter).

item_information A `tibble` with one row per item containing: `item`, `location` (mean item location, i.e. mean of thresholds for polytomous items), `info_at_location` (Fisher information at the item's own location), and `max_info` (maximum Fisher information across theta). For Rasch dichotomous items, both are 0.25. For polytomous items, information depends on the number and spacing of thresholds.

threshold_order (Polytomous models only) A `tibble` with one row per item containing: `item`, `n_thresholds`, `ordered` (logical: are all thresholds in ascending order?), and `prob_disordered` (posterior probability that at least one pair of adjacent thresholds is disordered, i.e. $\tau_{k+1} \leq \tau_k$). `NULL` for dichotomous models.

person_sd A `tibble` with one row containing: `mean`, `sd`, `hdci_lower`, `hdci_upper` — the posterior summary of the person-level standard deviation parameter σ_θ.

draws_matrix (Only if `draws = TRUE`) A numeric matrix with rows = thresholds (named `"item[threshold]"`) and columns = posterior draws. For dichotomous models, row names are item labels only.

References

Adams, R. J., Wu, M. L., & Wilson, M. (2012). The Rasch rating model and the disordered threshold controversy. *Educational and Psychological Measurement*, *72*(4), 547–573. doi:10.1177/0013164411432166

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`person_parameters`, `plot_targeting`, `plot_ipf`, `posterior_to_prior`.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Partial Credit Model ---
```

```
df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4, cores = 2, iter = 1000 # use more iter and cores
)

ip <- item_parameters(fit_pcm)

# Long format: one row per threshold
ip$locations

# Wide format: one row per item, easy to scan
ip$locations_wide

# Full summary with SE and HDCI
ip$summary

# Item information
ip$item_information

# Threshold ordering diagnostic
ip$threshold_order

# Person SD
ip$person_sd

# With full posterior draws
ip_draws <- item_parameters(fit_pcm, draws = TRUE)
dim(ip_draws$draws_matrix)

# --- Dichotomous Rasch ---

df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4, cores = 2, iter = 1000 # use more iter and cores
)

ip_rm <- item_parameters(fit_rm)
# Wide and long are equivalent for dichotomous:
```

```
ip_rm$locations
ip_rm$locations_wide
ip_rm$summary
```

`item_restscore_post` *Summarize and Plot Posterior Predictive Item-Restscore Associations*

Description

Postprocesses the output of `item_restscore_statistic` to produce a summary table and a slab plot comparing observed item-restscore gamma associations to the posterior predictive distribution.

Usage

```
item_restscore_post(item_restscore)
```

Arguments

`item_restscore` A list as returned by `item_restscore_statistic`, containing at minimum:

result A data frame with columns including `item` and summary statistics (first 5 columns are used).

draws A data frame with columns `item`, `gamma` (observed gamma per draw), and `gamma_rep` (replicated gamma per draw).

Details

The item-restscore gamma association measures the strength of the relationship between each item's responses and the rest score (total score excluding that item). Under good fit, the observed gamma should fall within the posterior predictive distribution.

The plot displays:

Grey slab The posterior predictive distribution of replicated gamma values, shaded by 84\ interval levels.

Orange diamonds The observed gamma values per draw, plotted as points on top of the slab.

Items where the observed gamma (orange) falls consistently outside the replicated distribution (grey) indicate poor fit in terms of item discrimination.

Value

A list with two elements:

summary A `tibble` with the first 5 columns of the result table, rounded to 3 decimal places and sorted by item.

plot A `ggplot` object showing the posterior predictive distribution of replicated gamma (grey filled slab) with the observed gamma values overlaid as orange diamond points.

See Also

item_restscore_statistic, infit_post.

Examples

```
## Not run:
library(brms)
library(ggplot2)

# Assuming fit_pcm is a fitted brmsfit object
irs <- item_restscore_statistic(fit_pcm)

result <- item_restscore_post(irs)
result$summary
result$plot

## End(Not run)
```

---

item_restscore_statistic

Posterior Predictive Item-Restscore Association for Bayesian IRT Models

---

Description

Computes a Bayesian analogue of the item-restscore association test (Kreiner, 2011) for Rasch-family models fitted with **brms**. For each posterior draw, the Goodman-Kruskal gamma coefficient between each item's score and the rest-score (total score minus that item) is computed for both observed and replicated data. Posterior predictive p-values indicate whether the observed association is stronger than the model predicts, which signals violations of local independence or unidimensionality.

Usage

```
item_restscore_statistic(
  model,
  item_var = item,
  person_var = id,
  ndraws_use = NULL
)
```

Arguments

model	A fitted brmsfit object from an ordinal IRT model (e.g., family = acat for a partial credit model) or a dichotomous model (family = bernoulli()).
item_var	An unquoted variable name identifying the item grouping variable in the model data (e.g., item).

`person_var` An unquoted variable name identifying the person grouping variable in the model data (e.g., `id`).

`ndraws_use` Optional positive integer. If specified, a random subset of posterior draws of this size is used. If `NULL` (the default), all draws are used.

Details

The item-restscore association is a key diagnostic in Rasch measurement. Under the Rasch model, each item should relate to the latent trait (and hence the rest-score) only through the modelled relationship. Goodman-Kruskal's gamma is a rank-based measure of association for ordinal cross-tabulations that is well-suited for this purpose (Kreiner, 2011).

The procedure for each posterior draw s is:

1. Obtain replicated responses $Y^{rep(s)}$ from `posterior_predict`.
2. For each item i and each person v, compute the rest-score: $R_{vi}^{obs} = \sum_{j \neq i} X_{vj}$ for observed data and $R_{vi}^{rep(s)} = \sum_{j \neq i} Y_{vj}^{rep(s)}$ for replicated data.
3. Cross-tabulate item score $\times$ rest-score and compute the Goodman-Kruskal gamma for both observed and replicated data.
4. Compare the two gammas across draws.

Items with ppp close to 1 have observed item-restscore association that is consistently stronger than the model predicts. This typically indicates that the item discriminates more than assumed under the equal-discrimination Rasch model (i.e., a violation of the Rasch assumption). Items with ppp close to 0 discriminate less than expected.

Value

A `tibble` with the following columns:

item The item identifier.

gamma_obs Posterior mean of the observed Goodman-Kruskal gamma between this item and the rest-score.

gamma_rep Posterior mean of the replicated gamma.

gamma_diff Posterior mean of `gamma_obs - gamma_rep`. Positive values indicate the observed item-restscore association is stronger than the model expects.

ppp Posterior predictive p-value: `mean(gamma_obs > gamma_rep)` across draws. Values close to 1 indicate the item discriminates more than the model predicts (too high discrimination). Values close to 0 indicate the item discriminates less than expected (too low discrimination, e.g., noise or miskeyed item).

gamma_obs_q025, gamma_obs_q975 95\ the observed gamma.

gamma_obs_q005, gamma_obs_q995 99\ the observed gamma.

gamma_diff_q025, gamma_diff_q975 95\ the gamma difference.

gamma_diff_q005, gamma_diff_q995 99\ the gamma difference.

References

Kreiner, S. (2011). A note on item-restscore association in Rasch models. *Applied Psychological Measurement*, *35*(7), 557–561.

Goodman, L. A. & Kruskal, W. H. (1954). Measures of association for cross classifications. *Journal of the American Statistical Association*, *49*(268), 732–764.

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`fit_statistic_pcm` for posterior predictive fit statistics, `fit_statistic_rm` for posterior predictive fit statistics, `infit_statistic` for Bayesian infit/outfit, `q3_statistic` for Bayesian Q3 residual correlations, `posterior_predict`.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

# Item-restscore association
irs <- item_restscore_statistic(
  model      = fit_pcm,
  ndraws_use = 100 # use at least 500
)

# Post-process draws
irs_results <- item_restscore_post(irs)
irs_results$summary
irs_results$plot

# --- Dichotomous Rasch Model ---

df_rm <- eRm::raschdat3 %>%
```

```
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

irs_rm <- item_restscore_statistic(
  model      = fit_rm,
  ndraws_use = 100 # use at least 500
)

# Post-process draws
irs_results <- item_restscore_post(irs_rm)
irs_results$summary
irs_results$plot
```

`person_parameters` *Extract Person Parameters from a Bayesian Rasch Model*

Description

Extracts person ability estimates from a fitted Bayesian Rasch model. Returns both Bayesian EAP (expected a posteriori) estimates with posterior SDs and frequentist WLE (Warm's weighted likelihood) estimates with standard errors, plus a lookup table mapping ordinal sum scores to both scales.

Usage

```
person_parameters(
  model,
  item_var = item,
  person_var = id,
  draws = FALSE,
  center = TRUE,
  theta_range = c(-7, 7)
)
```

Arguments

`model`	A fitted `brmsfit` object. Supported parameterisations:

Polytomous ordinal (PCM) e.g., `family = acat` with `thres(gr = item)`, producing item-specific thresholds.

Dichotomous Rasch (random items) e.g., `response ~ 1 + (1 | item) + (1 | id)` with `family = bernoulli()`.

Dichotomous 1PL (fixed items) e.g., `response ~ 0 + item + (1 | id)` with `family = bernoulli()`.

`item_var` An unquoted variable name identifying the item grouping variable in the model data. Default is `item`.

`person_var` An unquoted variable name identifying the person grouping variable in the model data. Default is `id`.

`draws` Logical. If `TRUE`, a matrix of full posterior draws (persons x draws) is included in the output. Default is `FALSE`.

`center` Logical. If `TRUE` (the default), person parameters and item difficulties are re-centered so that the mean item difficulty is zero, matching the convention in frequentist CML Rasch estimation. If `FALSE`, raw brms parameterisation is used.

`theta_range` A numeric vector of length 2 specifying the range for the Newton-Raphson WLE search. Default is `c(-7, 7)`.

Details

EAP estimates are extracted as the posterior means of the person random effects (`r_id[j, Intercept]`) from `as_draws_df`, with the posterior SD serving as the standard error. These are the standard Bayesian point estimates and reflect shrinkage toward the population mean.

WLE estimates (Warm, 1989) are computed from the posterior mean item parameters using Newton-Raphson iteration with adaptive step damping on the Warm-corrected likelihood. WLE adds a bias correction term $J(\theta)/(2I(\theta))$ to the score equations, where $I(\theta)$ is the test information and $J(\theta) = \sum_i \sum_c (c - E_i)^3 P_{ic}$ is the sum of third central moments. This produces estimates with reduced finite-sample bias compared to MLE, especially at extreme scores (Warm, 1989).

The Newton-Raphson algorithm uses adaptive step damping following the approach in **iarm** (Mueller): the maximum allowed step size shrinks by a factor of 1.05 each iteration, preventing overshoot and ensuring convergence for near-extreme scores.

For extreme scores (sum score = 0 or maximum possible), the WLE is not well-defined. These cases are assigned the boundary values of `theta_range` with `NA` standard errors.

When `center = TRUE` (the default), item difficulty parameters are shifted so their mean is zero, and EAP person parameters are shifted by the same constant. WLE is computed from the centered item parameters. This matches the convention in frequentist CML Rasch estimation.

Row ordering: Both `person_estimates` and `draws_matrix` preserve the original person order from the model data (order of first appearance of each person ID). This allows direct row-binding with the source data without re-matching.

Value

A list with the following elements:

person_estimates A `tibble` with one row per person containing: the person ID, `sum_score`, `eap` (posterior mean of person random effect), `eap_se` (posterior SD), `wle` (Warm's weighted likelihood estimate), and `wle_se` (asymptotic SE of the WLE). Rows are ordered to match the original person order in the model data (i.e., the order of first appearance of each person ID).

score_table A `tibble` mapping each observed ordinal sum score to its mean EAP, mean EAP SE, WLE, and WLE SE. Extreme scores (0 and maximum) receive WLE estimates at the boundary of `theta_range` with `NA` standard errors.

draws_matrix (Only if `draws = TRUE`) A numeric matrix with rows = persons and columns = posterior draws. Row names are person IDs. Rows are ordered to match the original person order in the model data. Can be passed directly to `RMUreliability`.

References

Warm, T. A. (1989). Weighted likelihood estimation of ability in item response theory. *Psychometrika*, *54*(3), 427–450. doi:10.1007/BF02294627

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

Christensen, K. B., Kreiner, S. & Mesbah, M. (Eds.) (2013). *Rasch Models in Health*. Iste and Wiley, pp. 63–70.

See Also

`RMUreliability`, `plot_targeting`, `ranef`, `as_draws_df`.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Dichotomous Rasch Model (random items) ---

df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4, cores = 2, iter = 1000 # use more iter and cores
)

# Basic usage — rows match original person order
pp <- person_parameters(fit_rm)
pp$person_estimates
pp$score_table
```

```
# With full posterior draws (e.g., for RMUreliability)
pp_draws <- person_parameters(fit_rm, draws = TRUE)
RMUreliability(pp_draws$draws_matrix)

# --- Polytomous PCM (acat) ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4, cores = 2, iter = 1000 # use more iter and cores
)

pp_pcm <- person_parameters(fit_pcm)
pp_pcm$score_table
```

---

plot_bars *Item Response Distribution Bar Chart*

Description

Creates a faceted bar chart showing the response distribution for each item, with counts and percentages displayed on each bar. Each item gets its own panel, with response categories on the x-axis and percentage of responses on the y-axis. This is a descriptive data visualization tool intended for use before model fitting.

Usage

```
plot_bars(
  data,
  item_labels = NULL,
  category_labels = NULL,
  ncol = 1,
  label_wrap = 25,
  text_y = 6,
  viridis_option = "A",
  viridis_end = 0.9,
  font = "sans"
)
```

Arguments

data	A data frame in wide format containing only the item response columns. Each column is one item, each row is one person. All columns must be numeric (integer-valued). Response categories may be coded starting from 0 or 1. Do not include person IDs, grouping variables, or other non-item columns.
item_labels	An optional character vector of descriptive labels for the items (facet strips). Must be the same length as `ncol(data)`. If `NULL` (the default), column names are used. Labels are displayed as `"column_name - label"`.
category_labels	An optional character vector of labels for the response categories (x-axis). Must be the same length as the number of response categories spanning from the minimum to the maximum observed value. If `NULL` (the default), numeric category values are used.
ncol	Integer. Number of columns in the faceted layout. Default is 1.
label_wrap	Integer. Number of characters per line in facet strip labels before wrapping. Default is 25.
text_y	Numeric. Vertical position (in percent units) for the count labels on each bar. Adjust upward if bars are tall. Default is 6.
viridis_option	Character. Viridis palette option for the count text color. One of `"A"` through `"H"`. Default is `"A"`.
viridis_end	Numeric in $[0, 1]$. End point of the viridis color scale for count text. Adjust if text is hard to read against the bar colors. Default is 0.9.
font	Character. Font family for all text. Default is `"sans"`.

Details

Each item is displayed as a separate facet panel with the item label in the strip on the left side. Bars are colored by response category using the viridis palette. Each bar shows the count (`n = X`) as text.

Input requirements:

- All columns must be numeric (integer-valued).
- The data frame must contain at least 2 columns (items) and at least 1 row (person).

Value

A ggplot object.

Examples

```
library(ggplot2)
if (requireNamespace("eRm", quietly = TRUE))

# Basic response distribution plot
plot_bars(eRm::pcmdat2)

# With custom item labels
plot_bars(
```

```
  eRm::pcmdat2,
  item_labels = c("Mood", "Sleep", "Appetite", "Energy")
)

# Two-column layout with wrapped labels
plot_bars(
  eRm::pcmdat2,
  item_labels = c(
    "General mood and emotional wellbeing",
    "Quality of sleep at night",
    "Appetite and eating habits",
    "Overall energy level during the day"
  ),
  ncol = 2, label_wrap = 20
)

# With custom category labels
plot_bars(
  eRm::pcmdat2,
  category_labels = c("Never", "Sometimes", "Often")
)
```

---

plot_icc *Item Characteristic Curves with Class Intervals*

Description

Plots Item Characteristic Curves (ICCs) for Bayesian Rasch-family models fitted with **brms**. Each item panel shows the model-expected item score curve (with a credible interval ribbon) overlaid with observed average item scores computed within class intervals. Optionally, observed scores can be split by a grouping variable to visually assess differential item functioning (DIF).

Usage

```
plot_icc(
  model,
  item_var = item,
  person_var = id,
  items = NULL,
  n_intervals = 5,
  theta_range = c(-4, 4),
  n_points = 200,
  center = TRUE,
  prob = 0.95,
  ncol = NULL,
  line_size = 0.8,
  ribbon_alpha = 0.3,
  point_size = 2.5,
```

```
  dif_var = NULL,
  dif_data = NULL,
  dif_labels = NULL,
  dif_stats = TRUE,
  min_n = 5,
  palette = NULL
)
```

Arguments

model	A fitted brmsfit object from an ordinal IRT model (e.g., family = acat) or a dichotomous model (family = bernoulli()).
item_var	An unquoted variable name identifying the item grouping variable in the model data. Default is item.
person_var	An unquoted variable name identifying the person grouping variable in the model data. Default is id.
items	An optional character vector of item names to plot. If NULL (the default), all items are plotted.
n_intervals	Integer. The number of class intervals into which persons are binned along the sum score. Default is 5.
theta_range	A numeric vector of length 2 specifying the range of theta for the expected curve. Default is c(-4, 4).
n_points	Integer. Number of evenly spaced theta values for computing the expected curve. Default is 200.
center	Logical. If TRUE (the default), the scale is recentered so the grand mean of item thresholds = 0, consistent with plot_targeting.
prob	Numeric in $(0, 1)$. Width of the credible interval ribbon around the expected curve. Default is 0.95.
ncol	Integer. Number of columns in the faceted layout. If NULL, chosen automatically.
line_size	Numeric. Line width for the expected curve. Default is 0.8.
ribbon_alpha	Numeric in $[0, 1]$. Transparency of the credible interval ribbon. Default is 0.3.
point_size	Numeric. Size of observed score points. Default is 2.5.
dif_var	An optional unquoted variable name for a grouping variable to assess DIF visually. If supplied, observed scores are computed separately per group and coloured accordingly.
dif_data	An optional data frame containing the DIF variable. Required when dif_var is specified and the variable was not part of the model formula (since brms drops unused variables from model$data). Must have the same rows and row order as the original model data.
dif_labels	An optional character vector of labels for the DIF groups. If NULL, factor levels are used.
dif_stats	Logical. If TRUE (the default when dif_var is specified), the partial gamma coefficient for each item is annotated in the plot panel. The partial gamma measures the strength of DIF, stratified by total score.

`min_n`	Integer. Minimum number of persons required in a class interval for the observed mean to be plotted. Intervals with fewer persons are dropped to avoid unstable estimates. Default is 5.
`palette`	An optional character vector of colors. If `NULL`, viridis colors are used.

Details

This is the Bayesian analogue of `ICCplot()` from the **iarm** package, using the class interval method.

Expected curve: For each item, the expected item score $E_i(\theta)$ is computed for a dense grid of theta values using the posterior draws of item thresholds. For the adjacent category (PCM/acat) family:

$$E_i(\theta) = \sum_{c=0}^{K} c \cdot P_{ic}(\theta)$$

where P_{ic} are the category probabilities. For dichotomous items, $E_i(\theta) = P_i(\theta)$.

The posterior mean of $E_i(\theta)$ is plotted as a line, with a credible interval ribbon showing the `prob` interval across posterior draws.

Class intervals: Following the approach in `iarm::ICCplot()`, persons are binned into class intervals by their ordinal *sum score* (the sufficient statistic in Rasch models), not by their EAP theta estimate. Within each interval, the mean observed item response is computed and plotted at the theta value corresponding to the mean sum score in that interval, obtained by inverting the posterior-mean expected total score function. Persons with extreme sum scores (0 and maximum) are excluded from the class intervals, as their theta positions cannot be estimated.

Uncertainty around observed points: For each class interval, the standard error of the mean observed response is computed and displayed as error bars showing ± 1.96 SE. These reflect sampling variability of the observed mean within each bin.

DIF overlay: When `dif_var` is specified, observed scores are computed separately for each level of the DIF variable, producing group-specific points connected by lines. Deviations between groups that track alongside each other (but offset from the expected curve) suggest uniform DIF. Crossing group lines suggest non-uniform DIF.

Partial gamma: When `dif_stats = TRUE` (default when DIF is requested), the Goodman-Kruskal partial gamma coefficient is displayed in each item panel. This measures the ordinal association between item response and DIF group, stratified by total score. Values near 0 indicate no DIF; values near ± 1 indicate strong DIF. The computation reuses the concordant/discordant pair counting algorithm from `gk_gamma`.

Value

A `ggplot` object.

See Also

`plot_ipf` for category probability curves, `plot_targeting` for person-item maps, `dif_statistic` for formal Bayesian DIF testing, `gk_gamma` for the underlying gamma algorithm.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)
library(ggplot2)

# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4, cores = 2, iter = 1000 # use more iter and cores
)

# Basic ICC plot with 5 class intervals
plot_icc(fit_pcm)

# Select specific items, more intervals
plot_icc(fit_pcm, items = c("I1", "I2"), n_intervals = 5)

# With DIF overlay and partial gamma annotation

# same dataset, adding a `gender` DIF variable
df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  mutate(gender = sample(c("M", "F"), nrow(.), TRUE)) %>%
  pivot_longer(!c(id,gender), names_to = "item", values_to = "response")

plot_icc(fit_pcm, dif_var = gender, dif_data = df_pcm,
         dif_labels = c("Female", "Male"), n_intervals = 5)
```

---

plot_ipf *Item Category Probability Function Curves for Polytomous IRT Models*

Description

Plots item category probability functions (ICPFs) for polytomous Bayesian IRT models fitted with **brms**. For each item, the probability of endorsing each response category is plotted as a function of the latent variable (theta), with separate colored curves per category. All items are displayed in a combined faceted plot.

Usage

```
plot_ipf(
  model,
  item_var = item,
  person_var = id,
  items = NULL,
  theta_range = c(-4, 4),
  n_points = 100,
  ncol = NULL,
  line_size = 0.8,
  ribbon_alpha = 0.15,
  prob = 0.95,
  category_labels = NULL,
  palette = NULL
)
```

Arguments

model	A fitted brmsfit object from a polytomous IRT model (e.g., family = acat for a partial credit model or family = cumulative for a graded response model).
item_var	An unquoted variable name identifying the item grouping variable in the model data (e.g., item).
person_var	An unquoted variable name identifying the person grouping variable in the model data (e.g., id).
items	An optional character vector of item names to plot. If NULL (the default), all items in the model are plotted.
theta_range	A numeric vector of length 2 specifying the range of the latent variable (theta) for the x-axis. Default is c(-4, 4).
n_points	Integer. Number of evenly spaced theta values at which to evaluate the category probabilities. Default is 100.
ncol	Integer. Number of columns in the faceted plot layout. If NULL (the default), an appropriate number is chosen automatically.
line_size	Numeric. Line width for the probability curves. Default is 0.8.
ribbon_alpha	Numeric in $[0, 1]$. Transparency of the credible interval ribbons. Default is 0.15. Set to 0 to hide ribbons.
prob	Numeric in $(0, 1)$. Width of the credible interval for the ribbons. Default is 0.95.
category_labels	An optional character vector of labels for the response categories. If NULL (the default), categories are labelled as integers starting from 1.
palette	An optional character vector of colors, one per response category. If NULL (the default), the viridis discrete scale from **ggplot2** is used.

Details

The function computes category probabilities directly from the posterior draws of the item threshold parameters. For the brms `acat` (adjacent category / partial credit) family with logit link, the density is:

$$P(Y = y|\eta) = \frac{\exp\left(\sum_{k=1}^{y}(\eta - \tau_k)\right)}{\sum_{k=0}^{K}\exp\left(\sum_{j=1}^{k}(\eta - \tau_j)\right)}$$

where η is the linear predictor (i.e., theta for a Rasch model with no additional fixed effects) and τ_k are the item thresholds. Analogous formulas are used for the `cumulative`, `sratio`, and `cratio` families.

Posterior uncertainty in the thresholds propagates into credible interval ribbons around the category probability curves — a Bayesian advantage over point-estimate-based plots.

Value

A `ggplot` object. The plot can be further customised using standard **ggplot2** functions.

References

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`posterior_epred`, `conditional_effects`,

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)
library(ggplot2)

# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

# Plot all items
```

```
plot_ipf(fit_pcm, item_var = item, person_var = id)

# Plot a subset of items
plot_ipf(fit_pcm, item_var = item, person_var = id,
         items = c("I1", "I2", "I3"))

# Customise appearance
plot_ipf(fit_pcm, item_var = item, person_var = id,
         theta_range = c(-6, 6), ncol = 3, prob = 0.90) +
  theme_minimal() +
  labs(title = "Item Category Probability Functions")
```

`plot_residual_pca` *Residual PCA Contrast Plot for Bayesian IRT Models*

Description

Assesses dimensionality of Bayesian IRT models by performing a principal component analysis (PCA) of standardized residuals for each posterior draw. The item loadings on the first residual contrast are plotted against item locations, with posterior uncertainty displayed as 2D density contours, crosshairs, or both. A posterior predictive p-value for the first-contrast eigenvalue tests whether the observed residual structure exceeds what the model predicts under unidimensionality.

Usage

```
plot_residual_pca(
  model,
  item_var = item,
  person_var = id,
  center = TRUE,
  prob = 0.95,
  ndraws_use = NULL,
  style = c("both", "density", "crosshair"),
  density_alpha = 0.3,
  density_bins = 6,
  density_palette = NULL,
  label_items = TRUE,
  point_size = 2.5,
  point_color = "#0072B2"
)
```

Arguments

`model`	A fitted `brmsfit` object from an ordinal IRT model (e.g., `family = acat`) or a dichotomous model (`family = bernoulli()`).

item_var	An unquoted variable name identifying the item grouping variable in the model data. Default is item.
person_var	An unquoted variable name identifying the person grouping variable in the model data. Default is id.
center	Logical. If TRUE (the default), item locations are mean-centered to zero, matching the convention used in plot_targeting.
prob	Numeric in $(0, 1)$. Width of the credible intervals for both loading and location whiskers. Default is 0.95.
ndraws_use	Optional positive integer. Number of posterior draws to use. If NULL (the default), up to 500 draws are used.
style	Character. Visual style for displaying uncertainty. "density" (the default) overlays filled 2D density contours per item computed from the draw-level location and loading values, showing the full joint posterior uncertainty. "crosshair" shows point estimates with horizontal and vertical credible interval bars. "both" displays density contours with crosshairs on top.
density_alpha	Numeric in $[0, 1]$. Maximum opacity of the density contours when style is "density" or "both". Default is 0.3.
density_bins	Integer. Number of contour bins for geom_density_2d_filled. Default is 6.
density_palette	An optional character vector of colors for the density contour fills (from low to high density). If NULL (the default), a blue sequential ramp is used. The length should match density_bins; the lowest (background) level is always transparent.
label_items	Logical. If TRUE (the default), item names are displayed next to points.
point_size	Numeric. Size of the item points. Default is 2.5.
point_color	Color for the item points and error bars. Default is "#0072B2".

Details

The procedure for each posterior draw s is:

1. Obtain category probabilities from posterior_epred. Compute expected values $E_{vi}^{(s)}$ and variances $Var_{vi}^{(s)}$.
2. Compute standardized residuals for observed data:

$$z_{vi}^{obs(s)} = \frac{X_{vi} - E_{vi}^{(s)}}{\sqrt{Var_{vi}^{(s)}}}$$

3. Generate replicated data $Y^{rep(s)}$ from posterior_predict and compute standardized residuals:

$$z_{vi}^{rep(s)} = \frac{Y_{vi}^{rep(s)} - E_{vi}^{(s)}}{\sqrt{Var_{vi}^{(s)}}}$$

4. Reshape both sets of residuals into person $\times$ item matrices and perform SVD on each.

5. Extract the first-contrast eigenvalue and item loadings from both observed and replicated SVDs.
6. Compare eigenvalues across draws: the posterior predictive p-value `ppp = mean(eigenvalue_obs > eigenvalue_rep)` tests whether the observed residual structure is stronger than what the model produces under its own assumptions.

When `style = "density"` or `style = "both"`, the draw-level (location, loading) pairs for each item are used to construct filled 2D kernel density contours via `geom_density_2d_filled`. The lowest contour level (outside all contours) is set to transparent so the white panel background shows through. Higher density regions use progressively darker fills.

Item loadings are aligned across draws using majority-sign alignment to resolve the sign indeterminacy of eigenvectors.

Value

A list with three elements:

plot A `ggplot` object showing item loadings on the first residual contrast (y-axis) versus item locations (x-axis).

data A `tibble` with columns: `item`, `location` (posterior mean item location), `location_lower`, `location_upper` (location CI), `loading` (posterior mean loading on first contrast), `loading_lower`, `loading_upper` (loading CI).

eigenvalue A `tibble` with columns: `eigenvalue_obs` (posterior mean observed eigenvalue), `eigenvalue_rep` (posterior mean replicated eigenvalue), `eigenvalue_diff` (posterior mean difference), `ppp` (posterior predictive p-value), `var_explained_obs`, `var_explained_rep` (posterior mean proportions of residual variance explained).

References

Smith, E. V. (2002). Detecting and evaluating the impact of multidimensionality using item fit statistics and principal component analysis of residuals. *Journal of Applied Measurement*, *3*, 205–231.

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`plot_targeting` for person-item maps, `plot_ipf` for item category probability curves, `q3_statistic` for Q3 residual correlations (another local dependence / dimensionality diagnostic).

Examples

```
## Not run:
library(brms)
library(dplyr)
library(tidyr)
library(tibble)
library(ggplot2)
```

```
# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 4,
  iter   = 2000
)

# 2D density contours (default)
result <- plot_residual_pca(fit_pcm)
result$plot

# Crosshair style
result_c <- plot_residual_pca(fit_pcm, style = "crosshair")
result_c$plot

# Both combined
result_b <- plot_residual_pca(fit_pcm, style = "both")
result_b$plot

# Custom warm palette
result_w <- plot_residual_pca(
  fit_pcm,
  density_palette = c("#FEE8C8", "#FDBB84", "#E34A33",
                      "#B30000", "#7F0000", "#4A0000"),
  point_color = "#B30000"
)
result_w$plot

## End(Not run)
```

---

plot_stackedbars *Stacked Bar Chart of Item Response Distributions*

Description

Creates a horizontal stacked bar chart showing the response distribution for all items. Each bar represents one item, with segments colored by response category. Counts are displayed as text labels within each segment. This is a descriptive data visualization tool intended for use before model fitting.

Usage

```
plot_stackedbars(
  data,
  item_labels = NULL,
  category_labels = NULL,
  show_n = TRUE,
  show_percent = FALSE,
  text_color = "sienna1",
  text_size = 3,
  min_label_n = 0,
  viridis_option = "D",
  viridis_end = 0.99,
  title = "Item responses"
)
```

Arguments

data	A data frame in wide format containing only the item response columns. Each column is one item, each row is one person. All columns must be numeric (integer-valued). Response categories may be coded starting from 0 or 1. Do not include person IDs, grouping variables, or other non-item columns.
item_labels	An optional character vector of descriptive labels for the items (y-axis). Must be the same length as `ncol(data)`. If `NULL` (the default), column names are used.
category_labels	An optional character vector of labels for the response categories (legend). Must be the same length as the number of response categories spanning from the minimum to the maximum observed value, ordered from lowest to highest category. If `NULL` (the default), numeric category values are used.
show_n	Logical. If `TRUE` (the default), the count of responses is displayed as a text label inside each bar segment.
show_percent	Logical. If `TRUE`, the percentage of responses is displayed instead of (or in addition to) counts. Default is `FALSE`.
text_color	Character. Color for the count/percentage labels. Default is `"sienna1"`.
text_size	Numeric. Size of the count/percentage labels. Default is 3.
min_label_n	Integer. Minimum count required for a label to be displayed within a bar segment. Segments with fewer responses are left unlabelled to avoid clutter. Default is 0 (all segments labelled).
viridis_option	Character. Viridis palette option. One of `"A"` through `"H"`. Default is `"D"`.
viridis_end	Numeric in $[0, 1]$. End point of the viridis color scale. Default is 0.99.
title	Character. Plot title. Default is `"Item responses"`.

Details

Items are displayed on the y-axis in the same order as the columns in `data` (first column at the top). Each bar is divided into segments representing response categories, with the lowest category on the

left and the highest on the right. The total bar length equals the number of non-missing responses for that item.

Categories with zero responses still appear in the legend but produce no visible bar segment, which helps identify gaps in the response distribution.

Input requirements:

- All columns must be numeric (integer-valued).
- The data frame must contain at least 2 columns (items) and at least 1 row (person).

Value

A ggplot object.

Examples

```
library(ggplot2)
if (requireNamespace("eRm", quietly = TRUE))

# Basic stacked bar chart
plot_stackedbars(eRm::pcmdat2)

# With custom item and category labels
plot_stackedbars(
  eRm::pcmdat2,
  item_labels = c("Mood", "Sleep", "Appetite", "Energy"),
  category_labels = c("Never", "Sometimes", "Often")
)

# Show percentages, suppress small segments
plot_stackedbars(
  eRm::pcmdat2,
  show_percent = TRUE,
  show_n       = FALSE,
  min_label_n  = 5
)
```

---

plot_targeting *Person-Item Map (Targeting Plot) for Bayesian IRT Models*

Description

Plots a person-item map (also known as a Wright map or targeting plot) for Bayesian IRT models fitted with **brms**. The plot consists of three vertically stacked panels sharing the same latent variable (theta / logit) x-axis:

Usage

```
plot_targeting(
  model,
  item_var = item,
  person_var = id,
  robust = FALSE,
  center = TRUE,
  sort_items = c("data", "location"),
  bins = 30,
  prob = 0.95,
  palette = NULL,
  person_fill = "#0072B2",
  threshold_fill = "#D55E00",
  height_ratios = c(3, 2, 5)
)
```

Arguments

`model`	A fitted `brmsfit` object from an ordinal IRT model (e.g., `family = acat`) or a dichotomous model (`family = bernoulli()`).
`item_var`	An unquoted variable name identifying the item grouping variable in the model data. Default is `item`.
`person_var`	An unquoted variable name identifying the person grouping variable in the model data. Default is `id`.
`robust`	Logical. If `FALSE` (the default), the histogram annotations use mean ± SD. If `TRUE`, median ± MAD is used instead.
`center`	Logical. If `TRUE` (the default), the scale is recentered so that the grand mean of all item threshold locations is zero, following the convention in frequentist Rasch analysis. Person estimates are shifted by the same constant. If `FALSE`, the raw brms parameterisation is used.
`sort_items`	Character. How to order items on the y-axis of the bottom panel. `"data"` (the default) preserves the order in which items first appear in the model data, with the first item at the top. `"location"` sorts items by their mean threshold location (easiest at top, hardest at bottom).
`bins`	Integer. Number of bins for both histograms. Default is 30.
`prob`	Numeric in $(0, 1)$. Width of the credible intervals for the item threshold whiskers. Default is 0.95.
`palette`	An optional character vector of colors for the response categories. If `NULL` (the default), the `viridis` discrete scale is used.
`person_fill`	Fill color for the person histogram. Default is `"#0072B2"` (blue).
`threshold_fill`	Fill color for the threshold histogram. Default is `"#D55E00"` (vermillion).
`height_ratios`	Numeric vector of length 3 specifying the relative heights of the top (person), middle (threshold), and bottom (dot-whisker) panels. Default is `c(3, 2, 5)`.

Details

1. **Top**: A histogram of person ability estimates, with a reference line for the mean (or median) and shading for ±1 SD (or ±1 MAD).
2. **Middle**: An inverted histogram of item threshold locations, with a reference line for the mean (or median) and shading for ±1 SD (or ±1 MAD), mirroring the top panel to visualise the overlap between person abilities and item difficulties.
3. **Bottom**: A dot-and-whisker plot of item thresholds by item, with credible intervals and color-coded response categories.

Together, the top and middle panels form a half-moon (or back-to-back histogram) display that makes it easy to assess whether the test is well-targeted to the sample.

Person estimates are obtained as the posterior means of the person random effects from the fitted model via `ranef`.

Item thresholds are extracted from the posterior draws. For models with grouped thresholds (`thres(gr = item)`), each item has its own set of threshold parameters. For models with a single set of thresholds (e.g., dichotomous Rasch with `(1 | item)`), the item random effects are subtracted from the global thresholds to obtain item-specific locations.

When `center = TRUE` (the default), the grand mean of all item threshold posterior means is computed and subtracted from every threshold estimate, its credible interval bounds, and every person estimate. This is a uniform translation of the entire scale that preserves all relative distances and matches the zero-centered item difficulty convention used in frequentist CML estimation.

Value

A `patchwork` object (combined `ggplot`).

References

Wright, B. D. & Stone, M. H. (1979). *Best Test Design*. MESA Press.

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`plot_ipf` for item category probability curves, `ranef`, `as_draws_df`.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)
library(ggplot2)
library(patchwork)

# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
```

```
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

# Default: centered, mean ± SD, items in data order
plot_targeting(fit_pcm)

# Uncentered (raw brms parameterisation)
plot_targeting(fit_pcm, center = FALSE)

# Robust: median ± MAD, items sorted by location
plot_targeting(fit_pcm, robust = TRUE, sort_items = "location")

# --- Dichotomous Rasch Model ---

df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

plot_targeting(fit_rm, sort_items = "location")
```

`plot_tile` *Tile Plot of Item Response Distributions*

Description

Creates a tile (heat map) plot showing the distribution of responses across all items and response categories. Each cell displays the count (or percentage) of responses, with optional conditional highlighting for cells with low counts. This is a descriptive data visualization tool intended for use before model fitting.

Usage

```
plot_tile(
  data,
  cutoff = 10,
  highlight = TRUE,
  percent = FALSE,
  text_color = "orange",
  item_labels = NULL,
  category_labels = NULL
)
```

Arguments

data	A data frame in wide format containing only the item response columns. Each column is one item, each row is one person. All columns must be numeric (integer-valued). Response categories may be coded starting from 0 or 1. Do not include person IDs, grouping variables, or other non-item columns.
cutoff	Integer. Cells with counts below this value are highlighted (when highlight = TRUE). Default is 10.
highlight	Logical. If TRUE (the default), cell labels with counts below cutoff are displayed in red. This includes cells with zero responses (empty categories), which is useful for identifying gaps in the response distribution.
percent	Logical. If TRUE, cell labels show percentages instead of raw counts. Default is FALSE.
text_color	Character. Color for cell label text (when not highlighted). Default is "orange".
item_labels	An optional character vector of descriptive labels for the items (y-axis). Must be the same length as ncol(data). If NULL (the default), column names are used.
category_labels	An optional character vector of labels for the response categories (x-axis). Must be the same length as the number of response categories spanning from the minimum to the maximum observed value. If NULL (the default), numeric category values are used.

Details

The plot displays items on the y-axis (in the same order as the columns in data, from top to bottom) and response categories on the x-axis. Cell shading represents the count of responses (darker = more responses). Cell labels show either raw counts or percentages.

Categories with zero responses are explicitly shown (as cells with n = 0), which helps identify gaps in the response distribution — one of the primary purposes of this plot.

Input requirements:

- All columns must be numeric (integer-valued).
- The data frame must contain at least 2 columns (items) and at least 1 row (person).

Value

A ggplot object.

Examples

```
library(ggplot2)
if (requireNamespace("eRm", quietly = TRUE))

# Basic tile plot
plot_tile(eRm::pcmdat2)

# With custom item labels
plot_tile(
  eRm::pcmdat2,
  item_labels = c("Mood", "Sleep", "Appetite", "Energy")
)

# With custom category labels and percentages
plot_tile(
  eRm::pcmdat2,
  category_labels = c("Never", "Sometimes", "Often"),
  percent = TRUE
)

# Adjust cutoff for highlighting
plot_tile(eRm::pcmdat2, cutoff = 20, highlight = TRUE)
```

---

posterior_to_prior *Extract Informative Priors from a Fitted Bayesian IRT Model*

Description

Takes a fitted brmsfit object and constructs a brmsprior object in which each item parameter receives a `normal(mean, sd)` prior derived from its posterior distribution. The person-level random effect SD prior is also updated. The returned prior can be passed directly to update (or brm) to refit the model with empirical Bayes / informative priors — useful for anchoring scales, warm-starting a model on new data, or regularising estimation with small samples.

Usage

```
posterior_to_prior(
  model,
  item_var = item,
  person_var = id,
  mult = 1,
  target_link = c("source", "logit", "probit")
)
```

Arguments

model	A fitted brmsfit object. Supported parameterisations: **Polytomous ordinal** e.g., family = acat with thres(gr = item), producing item-specific thresholds. **Dichotomous Rasch (random items)** e.g., response ~ 1 + (1 | item) + (1 | id) with family = bernoulli(). **Dichotomous 1PL (fixed items)** e.g., response ~ 0 + item + (1 | id) with family = bernoulli().
item_var	An unquoted variable name identifying the item grouping variable in the model data. Default is item.
person_var	An unquoted variable name identifying the person grouping variable in the model data. Default is id.
mult	Numeric multiplier applied to each posterior SD before it is used as the prior SD. Values > 1 widen the priors (less informative); values < 1 tighten them. Default is 1 (use posterior SD directly).
target_link	Character string specifying the link function of the model the priors will be used with. One of "logit", "probit", or "source" (the default). When "source", the link function of the fitted model is used and no transformation is applied. When different from the source model's link, all location and scale parameters are rescaled using the approximation $\beta_{\text{probit}} \approx \beta_{\text{logit}}/1.7$. This is useful when transferring priors from a logit-fitted model to a probit model or vice versa.

Details

The function extracts all posterior draws via as_draws_df, computes the mean and SD of each parameter's marginal posterior, and constructs normal(mean, sd * mult) priors.

Polytomous ordinal models with grouped thresholds (thres(gr = item)): each threshold receives its own prior via brms::set_prior("normal(...)", class = "Intercept",group = item, coef = threshold_index).

Dichotomous Rasch models parameterised as response ~ 1 + (1 | item) + (1 | id): priors are set on the global intercept (class = "Intercept"), the item-level SD (class = "sd", group = item_var), and the person-level SD.

Dichotomous 1PL models parameterised as response ~ 0 + item + (1 | id): each item-specific fixed effect (e.g., b_itemI1) receives its own normal(mean, sd) prior via brms::set_prior(..., class = "b", coef = "itemI1").

In all cases the person-level SD receives a normal(mean, sd * mult) prior (brms applies the lower bound of zero automatically for SD parameters).

Link function transformation: When target_link differs from the source model's link function, all parameters (means and SDs) are rescaled by a factor of approximately 1.7. This uses the well-known approximation that $\Phi(x) \approx \text{logistic}(1.7\,x)$, so logit-scale parameters can be converted to probit-scale by dividing by 1.7, and vice versa. The approximation is excellent for parameters in the range $|\beta| < 3$ and adequate beyond that range.

Value

A brmsprior object that can be supplied to the prior argument of brm or update.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4, cores = 2, iter = 1000 # use more iter (and cores if you have)
)

# Extract posterior-informed priors (same link)
new_priors <- posterior_to_prior(fit_pcm)
new_priors

# Narrow the prior's sd by a factor of 0.5
wide_priors <- posterior_to_prior(fit_pcm, mult = 0.5)

# Extract priors for use with a probit model
probit_priors <- posterior_to_prior(fit_pcm, target_link = "probit")

# --- Dichotomous 1PL (fixed item effects) ---

df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_1pl <- brm(
  response ~ 0 + item + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4, cores = 2, iter = 1000 # use more iter (and cores if you have)
)

priors_1pl <- posterior_to_prior(fit_1pl)
priors_1pl

# Transfer logit priors to a probit refit
priors_probit <- posterior_to_prior(fit_1pl, target_link = "probit")
```

q3_post *Summarize and Plot Posterior Predictive Q3 Residual Correlations*

Description

Postprocesses the output of q3_statistic to produce summary tables of posterior predictive Q3 statistics and a combined slab + interval plot comparing observed Q3 residual correlations to the posterior predictive distribution for each item pair.

Usage

```
q3_post(
  q3_draws,
  ci = 0.84,
  n_pairs = NULL,
  sort_by = c("q3_diff", "q3_obs", "ppp")
)
```

Arguments

q3_draws	A data frame (or tibble) as returned by q3_statistic containing at minimum the columns item_pair, q3 (observed Q3 per draw), and q3_rep (replicated Q3 per draw).
ci	Numeric in $(0, 1)$. Width of the credible interval used for the posterior predictive HDI and the slab display. Default is 0.84.
n_pairs	Integer. Maximum number of item pairs to display in the plot, selected by largest absolute q3_diff. If NULL (the default), all pairs are shown. Useful when the number of item pairs is large.
sort_by	Character. How to order item pairs on the y-axis. "q3_diff" (the default) sorts by the posterior mean difference between observed and replicated Q3 (largest at top). "q3_obs" sorts by the posterior mean observed Q3. "ppp" sorts by the posterior predictive p-value.

Details

Two complementary summary tables are provided:

summary Reports posterior mean observed and replicated Q3 values alongside the posterior predictive p-value (ppp). The ppp is the proportion of draws where the observed Q3 exceeds the replicated Q3. Under good fit (no local dependence), the ppp should be near 0.5. A ppp close to 1 indicates the observed correlation is systematically higher than expected (local dependence); a ppp close to 0 indicates it is systematically lower (local repulsion, e.g., speed-accuracy tradeoffs).

hdi Reports the probability that the observed Q3 falls above (local dependence) or below (local repulsion) the HDI of the replicated distribution. This provides a more distributional assessment than the ppp alone.

The plot uses two layers from the **ggdist** package:

`stat_slab` Displays the posterior predictive (replicated) Q3 distribution as a filled density slab per item pair, shaded by credible interval level.

`stat_slabinterval` Displays the observed Q3 distribution per item pair as a semi-transparent slab with point and interval summaries.

Value

A list with three elements:

summary A `tibble` with one row per item pair containing: `item_pair`, `item_1`, `item_2`, `q3_obs` (posterior mean observed Q3), `q3_rep` (posterior mean replicated Q3), `q3_diff` (posterior mean difference), and `q3_ppp` (posterior predictive p-value: proportion of draws where the observed Q3 exceeds the replicated Q3).

hdi A `tibble` with one row per item pair containing: `item_pair`, `item_1`, `item_2`, `ld` (local dependence probability: proportion of draws where observed Q3 exceeds upper HDI bound of replicated distribution), and `lr` (local repulsion probability: proportion of draws where observed Q3 falls below lower HDI bound).

plot A `ggplot` object showing the posterior predictive distribution of replicated Q3 (grey filled slab) overlaid with the observed Q3 distribution (coloured slab + interval), with a dashed reference line at 0.

See Also

`q3_statistic`, `infit_post`, `item_restscore_post`.

Examples

```
## Not run:
library(brms)
library(ggplot2)

# Assuming fit_pcm is a fitted brmsfit object
q3_draws <- q3_statistic(fit_pcm, ndraws_use = 500)

result <- q3_post(q3_draws)
result$summary
result$hdi
result$plot

# Show only top 10 pairs by Q3 difference
result_top <- q3_post(q3_draws, n_pairs = 10)
result_top$plot

## End(Not run)
```

`q3_statistic` *Posterior Predictive Q3 Residual Correlations for Bayesian IRT Models*

Description

Computes a Bayesian analogue of Yen's Q3 statistic (Yen, 1984) for detecting local dependence between item pairs in Rasch-family models fitted with **brms**. For each posterior draw, residual correlations are computed for both observed and replicated data, yielding draw-level Q3 values that can be summarized and visualized via `q3_post`.

Usage

```
q3_statistic(model, item_var = item, person_var = id, ndraws_use = NULL)
```

Arguments

`model`	A fitted `brmsfit` object from an ordinal IRT model (e.g., `family = acat` for a partial credit model) or a dichotomous model (`family = bernoulli()`).
`item_var`	An unquoted variable name identifying the item grouping variable in the model data. Default is `item`.
`person_var`	An unquoted variable name identifying the person grouping variable in the model data. Default is `id`.
`ndraws_use`	Optional positive integer. If specified, a random subset of posterior draws of this size is used. If `NULL` (the default), all draws are used.

Details

The procedure works as follows for each posterior draw s:

1. Compute expected values $E_{vi}^{(s)}$ from the category probabilities returned by `posterior_epred`. For ordinal models: $E_{vi}^{(s)} = \sum_c c \cdot P^{(s)}(X_{vi} = c)$. For binary models: $E_{vi}^{(s)} = P^{(s)}(X_{vi} = 1)$.
2. Compute observed residuals: $d_{vi}^{(s)} = X_{vi} - E_{vi}^{(s)}$.
3. Compute replicated residuals: $d_{vi}^{rep(s)} = Y_{vi}^{rep(s)} - E_{vi}^{(s)}$, where Y^{rep} is drawn via `posterior_predict`.
4. For each item pair (i, j), compute Q3 as the Pearson correlation of residuals across all persons who responded to both items.

Value

A `tibble` in long format with one row per draw per item pair, containing:

draw Integer draw index.

item_pair Character label of the item pair (`"item1 : item2"`).

item_1 First item in the pair.

item_2 Second item in the pair.

q3 Observed Q3 residual correlation for this draw.

q3_rep Replicated Q3 residual correlation for this draw.

This long-format output parallels the structure of `infit_statistic` and can be passed directly to `q3_post` for summary tables and plots.

References

Yen, W. M. (1984). Effects of local item dependence on the fit and equating performance of the three-parameter logistic model. *Applied Psychological Measurement*, *8*(2), 125–145. doi:10.1177/014662168400800201

Christensen, K. B., Makransky, G. & Horton, M. (2017). Critical values for Yen's Q3: Identification of local dependence in the Rasch model using residual correlations. *Applied Psychological Measurement*, *41*(3), 178–194. doi:10.1177/0146621616677520

Bürkner, P.-C. (2021). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*, *100*, 1–54. doi:10.18637/jss.v100.i05

See Also

`q3_post` for postprocessing summaries and plots, `infit_statistic` for Bayesian infit/outfit, `posterior_epred`, `posterior_predict`.

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Partial Credit Model ---

df_pcm <- eRm::pcmdat2 %>%
  mutate(across(everything(), ~ .x + 1)) %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_pcm <- brm(
  response | thres(gr = item) ~ 1 + (1 | id),
  data   = df_pcm,
  family = acat,
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

# Q3 residual correlations
q3_draws <- q3_statistic(fit_pcm, ndraws_use = 500)

# Postprocess
result <- q3_post(q3_draws)
result$summary
```

```
result$hdi
result$plot

# --- Dichotomous Rasch Model ---

df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

q3_draws <- q3_statistic(fit_rm, ndraws_use = 500)

# Postprocess
result <- q3_post(q3_draws)
result$summary
result$hdi
result$plot
```

`RMUreliability` *Estimate reliability (Relative Measurement Uncertainty) from Bayesian measurement models*

Description

This function measures reliability using posterior draws from a fitted Bayesian model.

Usage

```
RMUreliability(input_draws, verbose = FALSE, level = 0.95)
```

Arguments

`input_draws`	A matrix or data frame of posterior draws. Rows represent subjects and columns represent draws.
`verbose`	Logical. Print detailed information about the input data. Default is TRUE.
`level`	Numeric. Credibility level for the highest density continuous interval. Default is 0.95.

Details

To use this function, you will need to provide a matrix (input_draws) that contains the posterior draws for the parameter you wish to calculate reliability. The function assumes that rows of input_draws represent subjects and columns represent posterior draws.

For an example of how to apply this function to calculate mean score reliability using brms, see this tutorial.

For an example of how to apply this function to go/go-no task data using brms, see this tutorial.

Value

A list containing:

- hdci: A data frame with a point-estimate (posterior mean) and highest density continuous interval for reliability, calculated using the ggdist::mean_hdci function
- reliability_posterior_draws: A numeric vector of posterior draws for reliability, of length K/2 (K = number of columns/draws in your input_draws matrix)

References

Bignardi, G., Kievit, R., & Bürkner, P. C. (2025). A general method for estimating reliability using Bayesian Measurement Uncertainty. PsyArXiv. doi:10.31234/osf.io/h54k8

Examples

```
library(brms)
library(dplyr)
library(tidyr)
library(tibble)

# --- Dichotomous Rasch Model ---

df_rm <- eRm::raschdat3 %>%
  as.data.frame() %>%
  rownames_to_column("id") %>%
  pivot_longer(!id, names_to = "item", values_to = "response")

fit_rm <- brm(
  response ~ 1 + (1 | item) + (1 | id),
  data   = df_rm,
  family = bernoulli(),
  chains = 4,
  cores  = 1, # use more cores if you have
  iter   = 500 # use at least 2000
)

# Extract posterior draws from brms model

posterior_draws = fit_rm %>%
  as_draws_df() %>%
  as_tibble() %>%
```

```
  select(starts_with("r_id")) %>%
  t()

# Calculate RMU

RMUreliability(posterior_draws)
```

Index