Package ‘easyRaschBayes’

March 11, 2026

Title Bayesian Rasch Analysis Using 'brms'

Version 0.1.0

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

Suggests ggplot2 (>= 3.4.0), ggrepel, patchwork, eRm, testthat (>= 3.0.0), knitr, rmarkdown

URL `https://github.com/pgmj/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 2026-03-11 08:10:08 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_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)
```

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.

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. **Outfit** is the unweighted mean of Z_{vi}^2 across persons within each item.
5. **Infit** is the variance-weighted mean:

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

6. The same computations are repeated for replicated data Y^{rep} drawn via `posterior_predict`.

Under perfect fit, both infit and outfit have an expected value of 1. Values substantially above 1 indicate underfit (too much noise), values below 1 indicate overfit (too little variation, e.g., redundancy). Posterior predictive p-values near 0 or 1 indicate misfit.

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 The observed outfit statistic for that item and draw.

outfit_rep 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

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

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`, `pp_check`.

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
)

# Summarise across draws
item_infit %>%
  group_by(item) %>%
  summarise(
    infit_obs = mean(infit),
    infit_rep = mean(infit_rep),
    infit_ppp = mean(infit_rep > infit)
  )

# --- 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
)

item_infit_rm %>%
  group_by(item) %>%
  summarise(
    infit_obs = mean(infit),
    infit_rep = mean(infit_rep),
    infit_ppp = mean(infit_rep > infit)
  )
```

`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 × 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. (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 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
)

# Flag items with too-strong discrimination (ppp > 0.95)
irs %>% filter(ppp > 0.95)

# Flag items with too-weak discrimination (ppp < 0.05)
irs %>% filter(ppp < 0.05)

# --- Dichotomous Rasch Model ---

df_rm <- eRm::rainger %>%
  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
)

irs_rm %>%
  arrange(ppp)
```

`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, similar to the trace plots produced by `itemplot` in the **mirt** package.

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 from packages like **mirt** or **eRm**.

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`, `itemplot`.

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 × 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

```
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
)
```

```
# 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
```

`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::rainger %>%
  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")
```

`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 a posterior predictive p-value for each item pair that is automatically calibrated without requiring knowledge of the sampling distribution.

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 (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 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.
5. Aggregate across draws to obtain posterior means, quantiles, and posterior predictive p-values.

The key advantage over parametric bootstrapping is that the reference distribution is obtained directly from the posterior, automatically accounting for parameter uncertainty and the negative bias inherent in Q3 (which depends on test length and person ability distribution).

Value

A `tibble` with the following columns:

item_1 First item in the pair.

item_2 Second item in the pair.

q3_obs Posterior mean of the observed Q3 residual correlation.

q3_rep Posterior mean of the replicated Q3 residual correlation.

q3_diff Posterior mean of `q3_obs - q3_rep`. Large positive values indicate that the observed residual correlation exceeds what the model expects.

ppp Posterior predictive p-value: `mean(q3_obs > q3_rep)` across draws. Values close to 1 indicate local dependence (observed correlation systematically higher than replicated).

q3_obs_q025, q3_obs_q975 2.5\ (95\ observed Q3.

q3_obs_q005, q3_obs_q995 0.5\ (99\ observed Q3.

q3_diff_q025, q3_diff_q975 2.5\ (95\ Q3 differences.

q3_diff_q005, q3_diff_q995 0.5\ (99\ Q3 differences.

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.

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. (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 posterior predictive fit statistics with user-supplied criterion functions, `fit_statistic_rm` for posterior predictive fit statistics with user-supplied criterion functions, `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_results <- q3_statistic(
  model      = fit_pcm,
  ndraws_use = 100 # use at least 500
)

# Flag item pairs with ppp > 0.95 as locally dependent
q3_results %>%
  filter(ppp > 0.95) %>%
  arrange(desc(q3_diff))

# Inspect 99% credible intervals for Q3 differences
q3_results %>%
  filter(q3_diff_q005 > 0)

# --- 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_rm <- q3_statistic(
  model      = fit_rm,
  ndraws_use = 100 # use at least 500
)

q3_rm %>%
  filter(ppp > 0.95)
```

`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

```
# See https://www.bignardi.co.uk/8_bayes_reliability/tutorial_rmu_sum_score_reliability.html
# for more details on this example

# Simulate data

library(dplyr)
library(tidyr)
library(brms)
set.seed(1)
N                   = 5000 # number of subjects (mice)
J                   = 3    # number of measurements per subject
true_score_variance = 1
error_variance      = 10

df = expand.grid(j = 1:J, mouse = 1:N)

true_scores       = rnorm(N, mean = 10, sd = sqrt(true_score_variance))
measurement_error = rnorm(N*J, mean = 0, sd = sqrt(error_variance))

df$measurement = true_scores[df$mouse] + measurement_error

df_average_lengths = df %>%
  group_by(mouse) %>%
  summarise(average_measurement = mean(measurement))

# Reliability should equal this:

true_score_variance/(true_score_variance+error_variance/J)

# Approximately the same as:

cor(df_average_lengths$average_measurement, true_scores)^2

# Fit model and calculate RMU

brms_model = brm(
```

```
  measurement ~ 1 + (1 | mouse),
  data    = df,
  iter = 500,
  warmup = 150
)

# Extract posterior draws from brms model

posterior_draws = brms_model %>%
  as_draws_df() %>%
  as_tibble %>%
  select(starts_with("r_mouse")) %>%
  t()

# Calculate RMU

RMUreliability(posterior_draws)$hdci
```

Index