# Package: measr (via r-universe)

September 11, 2024

**Title** Bayesian Psychometric Measurement Using 'Stan'

**Version** 1.0.0.9000

**Description** Estimate diagnostic classification models (also called cognitive diagnostic models) with 'Stan'. Diagnostic classification models are confirmatory latent class models, as described by Rupp et al. (2010, ISBN: 978-1-60623-527-0). Automatically generate 'Stan' code for the general loglinear cognitive diagnostic diagnostic model proposed by Henson et al. (2009) <doi:10.1007/s11336-008-9089-5> and other subtypes that introduce additional model constraints. Using the generated 'Stan' code, estimate the model evaluate the model's performance using model fit indices, information criteria, and reliability metrics.

**License** GPL (>= 3)

**URL** https://measr.info, https://github.com/wjakethompson/measr

**BugReports** https://github.com/wjakethompson/measr/issues

**Depends** R (>= 4.1.0)

**Imports** dcm2, dplyr (>= 1.1.1), dtplyr, fs, glue, loo, magrittr, methods, posterior, psych, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rlang (>= 0.4.11), rstan (>= 2.26.0), rstantools (>= 2.3.0), stats, tibble, tidyr (>= 1.3.0)

**LinkingTo** BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.26.0), StanHeaders (>= 2.26.0)

**Suggests** cli, cmdstanr (>= 0.4.0), crayon, knitr, rmarkdown, roxygen2, spelling, testthat (>= 3.0.0)

**Additional_repositories** https://mc-stan.org/r-packages/

**Config/testthat/edition** 3

**Config/Needs/website** wjakethompson/wjake, showtext, ggdist, english

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.1

**Biarch** true

**SystemRequirements** GNU make

**VignetteBuilder** knitr

**Repository** https://wjakethompson.r-universe.dev

**RemoteUrl** https://github.com/wjakethompson/measr

**RemoteRef** HEAD

**RemoteSha** bacc0a799b51ff96af8777d7e1415f006c8baa7b

# Contents

`as_measrfit` *Coerce objects to a* `measrfit`

## Description

Coerce objects to a `measrfit`

## Usage

```
as_measrfit(x, class = character())

## Default S3 method:
as_measrfit(x, class = character())
```

## Arguments

| | |
|---|---|
| `x` | An object to be coerced to a `measrfit`. |
| `class` | Additional classes to be added (e.g., `measrdcm` for a diagnostic classification model). |

## Value

An object of class measrfit.

## See Also

measrfit, `measrfit()`, `is_measrfit()`

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)

new_obj <- as_measrfit(rstn_mdm_lcdm, class = "measrdcm")
```

## c.measrprior *Combine multiple measrprior objects into one measrprior*

### Description

Combine multiple measrprior objects into one measrprior

### Usage

```
## S3 method for class 'measrprior'
c(x, ..., replace = FALSE)
```

### Arguments

| | |
|---|---|
| x | A measrprior object. |
| ... | Additional measrprior objects to be combined. |
| replace | Should only unique priors be kept? If TRUE, the first prior specified is kept. |

### Value

A measrprior object.

## create_profiles *Generate mastery profiles*

### Description

Given the number of attributes, generate all possible patterns of attribute mastery.

### Usage

```
create_profiles(attributes)
```

### Arguments

| | |
|---|---|
| attributes | Positive integer. The number of attributes being measured. |

### Value

A tibble with all possible attribute mastery profiles. Each row is a profile, and each column indicates whether the attribute in that column was mastered (1) or not mastered (0). Thus, the tibble will have 2^attributes rows, and attributes columns.

### Examples

```
create_profiles(3L)
create_profiles(5)
```

## default_dcm_priors *Default priors for diagnostic classification models*

### Description

Default priors for diagnostic classification models

### Usage

```
default_dcm_priors(type = "lcdm", attribute_structure = "unconstrained")
```

### Arguments

| | |
|---|---|
| type | Type of DCM to estimate. Must be one of lcdm, dina, dino, or crum. |
| attribute_structure | Structural model specification. Must be one of unconstrained, or independent. `unconstrained` makes no assumptions about the relationships between attributes, whereas `independent` assumes that proficiency statuses on attributes are independent of each other. |

### Value

A `measrprior` object.

### Examples

```
default_dcm_priors(type = "lcdm")
```

## ecpe_data *Examination for the Certificate of Proficiency in English (ECPE)*

### Description

This is data from the grammar section of the ECPE, administered annually by the English Language Institute at the University of Michigan. This data contains responses to 28 questions from 2,922 respondents, which ask respondents to complete a sentence with the correct word. This data set has been used by Templin & Hoffman (2013) and Templin & Bradshaw (2014) for demonstrating the log-linear cognitive diagnosis model (LCDM) and the hierarchical diagnostic classification model (HDCM), respectively.

### Usage

```
ecpe_data

ecpe_qmatrix
```

## Format

`ecpe_data` is a tibble containing ECPE response data with 2,922 rows and 29 variables.

- `resp_id`: Respondent identifier
- `E1-E28`: Dichotomous item responses to the 28 ECPE items

`ecpe_qmatrix` is a tibble that identifies which skills are measured by each ECPE item. This section of the ECPE contains 28 items measuring 3 skills. The `ecpe_qmatrix` correspondingly is made up of 28 rows and 4 variables.

- `item_id`: Item identifier, corresponds to `E1-E28` in `ecpe_data`
- `morphosyntactic`, `cohesive`, and `lexical`: Dichotomous indicator for whether or not the skill is measured by each item. A value of `1` indicates the skill is measured by the item and a value of `0` indicates the skill is not measured by the item.

## Details

The skills correspond to knowledge of:

1. Morphosyntactic rules
2. Cohesive rules
3. Lexical rules

For more details, see Buck & Tatsuoka (1998) and Henson & Templin (2007).

## References

Buck, G., & Tatsuoka, K. K. (1998). Application of the rule-space procedure to language testing: Examining attributes of a free response listening test. *Language Testing, 15*(2), 119-157. doi:10.1177/026553229801500201

Henson, R., & Templin, J. (2007, April). *Large-scale language assessment using cognitive diagnosis models.* Paper presented at the Annual meeting of the National Council on Measurement in Education, Chicago, IL.

Templin, J., & Hoffman, L. (2013). Obtaining diagnostic classification model estimates using Mplus. *Educational Measurement: Issues and Practice, 32*(2), 37-50. doi:10.1111/emip.12010

Templin, J., & Bradshaw, L. (2014). Hierarchical diagnostic classification models: A family of models for estimating and testing attribute hierarchies. *Psychometrika, 79*(2), 317-339. doi:10.1007/s1133601393620

`fit_m2.measrdcm` *Estimate the $M_2$ fit statistic for diagnostic classification models*

## Description

For diagnostic classification models, the $M_2$ statistic is calculated as described by Hansen et al. (2016) and Liu et al. (2016).

## Usage

```
## S3 method for class 'measrdcm'
fit_m2(model, ..., ci = 0.9, force = FALSE)
```

## Arguments

| | |
|---|---|
| `model` | An estimated diagnostic classification model. |
| `...` | Unused, for extensibility. |
| `ci` | The confidence interval for the RMSEA. |
| `force` | If the $M_2$ has already been saved to the model object with `add_fit()`, should it be recalculated. Default is `FALSE`. |

## Value

A data frame created by `dcm2::fit_m2()`.

## Methods (by class)

- `fit_m2(measrdcm)`: $M_2$ for diagnostic classification models.

## References

Hansen, M., Cai, L., Monroe, S., & Li, Z. (2016). Limited-information goodness-of-fit testing of diagnostic classification item response models. *British Journal of Mathematical and Statistical Psychology, 69*(3), 225-252. doi:10.1111/bmsp.12074

Liu, Y., Tian, W., & Xin, T. (2016). An application of $M_2$ statistic to evaluate the fit of cognitive diagnostic models. *Journal of Educational and Behavioral Statistics, 41*(1), 3-26. doi:10.3102/1076998615621293

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)

fit_m2(rstn_mdm_lcdm)
```

`fit_ppmc` *Posterior predictive model checks for assessing model fit*

## Description

For models estimated with `method = "mcmc"`, use the posterior distributions to compute expected distributions for fit statistics and compare to values in the observed data.

## Usage

```
fit_ppmc(
  model,
  ndraws = NULL,
  probs = c(0.025, 0.975),
  return_draws = 0,
  model_fit = c("raw_score"),
  item_fit = c("conditional_prob", "odds_ratio", "pvalue"),
  force = FALSE
)
```

## Arguments

| | |
|---|---|
| `model` | A measrfit object. |
| `ndraws` | The number of posterior draws to base the checks on. Must be less than or equal to the total number of posterior draws retained in the estimated model. If `NULL` (the default) the total number from the estimated model is used. |
| `probs` | The percentiles to be computed by the `[stats::quantile()]` function for summarizing the posterior distributions of the specified fit statistics. |
| `return_draws` | Proportion of posterior draws for each specified fit statistic to be returned. This does not affect the calculation of the posterior predictive checks, but can be useful for visualizing the fit statistics. For example, if `ndraws = 500`, `return_draws = 0.2`, and `model_fit = "raw_score"`, then the raw score chi-square will be computed 500 times (once for each draw) and 100 of those values (0.2 * 500) will be returned. If `0` (the default), only summaries of the posterior are returned (no individual samples). |
| `model_fit` | The posterior predictive model checks to compute for an evaluation of model-level fit. If `NULL`, no model-level checks are computed. See details. |
| `item_fit` | The posterior predictive model checks to compute for an evaluation of item-level fit. If `NULL`, no item-level checks are computed. Multiple checks can be provided in order to calculate more than one check simultaneously (e.g., `item_fit = c("conditional_prob", "odds_ratio")`). See details. |
| `force` | If all requested PPMCs have already been added to the model object using `add_fit()`, should they be recalculated. Default is `FALSE`. |

## Details

Posterior predictive model checks (PPMCs) use the posterior distribution of an estimated model to compute different statistics. This creates an expected distribution of the given statistic, *if our estimated parameters are correct*. We then compute the statistic in our observed data and compare the observed value to the expected distribution. Observed values that fall outside of the expected distributions indicate incompatibility between the estimated model and the observed data.

We currently support PPMCs at the model and item level. At the model level, we calculate the expected raw score distribution (`model_fit = "raw_score"`), as described by Thompson (2019) and Park et al. (2015).

At the item level, we can calculate the conditional probability that a respondent in each class provides a correct response (`item_fit = "conditional_prob"`) as described by Thompson (2019) and Sinharay & Almond (2007) or the overall proportion correct for an item (`item_fit = "pvalue"`), as described by Thompson (2019). We can also calculate the odds ratio for each pair of items (`item_fit = "odds_ratio"`) as described by Park et al. (2015) and Sinharay et al. (2006).

## Value

A list with two elements, "model_fit" and "item_fit". If either `model_fit = NULL` or `item_fit = NULL` in the function call, this will be a one-element list, with the null criteria excluded. Each list element, is itself a list with one element for each specified PPMC containing a tibble. For example if `item_fit = c("conditional_prob", "odds_ratio")`, the "item_fit" element will be a list of length two, where each element is a tibble containing the results of the PPMC. All tibbles follow the same general structure:

- `obs_{ppmc}`: The value of the relevant statistic in the observed data.
- `ppmc_mean`: The mean of the `ndraws` posterior samples calculated for the given statistic.
- Quantile columns: 1 column for each value of `probs`, providing the corresponding quantiles of the `ndraws` posterior samples calculated for the given statistic.
- `samples`: A list column, where each element contains a vector of length `(ndraws * return_draws)`, representing samples from the posterior distribution of the calculated statistic. This column is excluded if `return_draws = 0`.
- `ppp`: The posterior predictive p-value. This is the proportion of posterior samples for calculated statistic that are greater than the observed value. Values very close to 0 or 1 indicate incompatibility between the fitted model and the observed data.

## References

Park, J. Y., Johnson, M. S., Lee, Y-S. (2015). Posterior predictive model checks for cognitive diagnostic models. *International Journal of Quantitative Research in Education, 2*(3-4), 244-264. doi:10.1504/IJQRE.2015.071738

Sinharay, S., & Almond, R. G. (2007). Assessing fit of cognitive diagnostic models. *Educational and Psychological Measurement, 67*(2), 239-257. doi:10.1177/0013164406292025

Sinharay, S., Johnson, M. S., & Stern, H. S. (2006). Posterior predictive assessment of item response theory models. *Applied Psychological Measurement, 30*(4), 298-321. doi:10.1177/0146621605285517

Thompson, W. J. (2019). *Bayesian psychometrics for diagnostic assessments: A proof of concept* (Research Report No. 19-01). University of Kansas; Accessible Teaching, Learning, and Assessment Systems. doi:10.35542/osf.io/jzqs8

## Examples

```
mdm_dina <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "dina",
  method = "mcmc", seed = 63277, backend = "rstan",
  iter = 700, warmup = 500, chains = 2, refresh = 0
)

fit_ppmc(mdm_dina, model_fit = "raw_score", item_fit = NULL)
```

---

# get_parameters *Get a list of possible parameters*

---

## Description

When specifying prior distributions, it is often useful to see which parameters are included in a given model. Using the Q-matrix and type of diagnostic model to estimated, we can create a list of all included parameters for which a prior can be specified.

## Usage

```
get_parameters(
  qmatrix,
  item_id = NULL,
  rename_att = FALSE,
  rename_item = FALSE,
  type = c("lcdm", "dina", "dino", "crum"),
  attribute_structure = c("unconstrained", "independent")
)
```

## Arguments

| | |
|---|---|
| qmatrix | The Q-matrix. A data frame with 1 row per item and 1 column per attribute. All cells should be either 0 (item does not measure the attribute) or 1 (item does measure the attribute). |
| item_id | Optional. Variable name of a column in `qmatrix` that contains item identifiers. `NULL` (the default) indicates that no identifiers are present in the Q-matrix. |
| rename_att | Should attribute names from the `qmatrix` be replaced with generic, but consistent names (e.g., "att1", "att2", "att3"). |
| rename_item | Should item names from the `qmatrix` be replaced with generic, but consistent names (e.g., 1, 2, 3). |
| type | Type of DCM to estimate. Must be one of lcdm, dina, dino, or crum. |
| attribute_structure | Structural model specification. Must be one of unconstrained, or independent. `unconstrained` makes no assumptions about the relationships between attributes, whereas `independent` assumes that proficiency statuses on attributes are independent of each other. |

## Value

A tibble with one row per parameter.

## Examples

```
get_parameters(ecpe_qmatrix, item_id = "item_id", type = "lcdm")

get_parameters(ecpe_qmatrix, item_id = "item_id", type = "lcdm",
               rename_att = TRUE)
```

---

# is_measrfit *Check if argument is a* measrfit *object*

---

## Description

Check if argument is a measrfit object

## Usage

```
is_measrfit(x)
```

## Arguments

| | |
|---|---|
| x | An object to be checked |

## Value

A logical indicating is x is a measrfit object.

## See Also

measrfit, measrfit(), as_measrfit()

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)

is_measrfit(rstn_mdm_lcdm)
```

## is_measrprior *Checks if argument is a* measrprior *object*

### Description

Checks if argument is a measrprior object

### Usage

```
is_measrprior(x)
```

### Arguments

| | |
|---|---|
| x | An object to be checked |

### Value

A logical indicating if x is a measrprior object.

### Examples

```
prior1 <- prior(lognormal(0, 1), class = maineffect)
is_measrprior(prior1)

prior2 <- 3
is_measrprior(prior2)
```

## loglik_array *Extract the log-likelihood of an estimated model*

### Description

The loglik_array() methods for measrfit objects calculates the log-likelihood for an estimated model via the generated quantities functionality in *Stan* and returns the draws of the log_lik parameter.

### Usage

```
loglik_array(model)

## S3 method for class 'measrdcm'
loglik_array(model)
```

### Arguments

| | |
|---|---|
| model | A measrfit object. |

### Value

A "draws_array" object containing the log-likelihood estimates for the model.

---

`loo.measrfit` *Efficient approximate leave-one-out cross-validation (LOO)*

---

### Description

A `loo::loo()` method that is customized for `measrfit` objects. This is a simple wrapper around `loo::loo.array()`. See the **loo** package vignettes for details.

### Usage

```
## S3 method for class 'measrfit'
loo(x, ..., r_eff = NA, force = FALSE)
```

### Arguments

| | |
|---|---|
| `x` | A measrfit object. |
| `...` | Additional arguments passed to `loo::loo.array()`. |
| `r_eff` | Vector of relative effective sample size estimates for the likelihood (`exp(log_lik)`) of each observation. This is related to the relative efficiency of estimating the normalizing term in self-normalized importance sampling when using posterior draws obtained with MCMC. If MCMC draws are used and `r_eff` is not provided then the reported PSIS effective sample sizes and Monte Carlo error estimates can be over-optimistic. If the posterior draws are (near) independent then `r_eff=1` can be used. `r_eff` has to be a scalar (same value is used for all observations) or a vector with length equal to the number of observations. The default value is 1. See the `relative_eff()` helper functions for help computing `r_eff`. |
| `force` | If the LOO criterion has already been added to the model object with `add_criterion()`, should it be recalculated. Default is `FALSE`. |

### Value

The object returned by `loo::loo.array()`.

## loo_compare.measrfit *Relative model fit comparisons*

### Description

A loo::loo_compare() method that is customized for measrfit objects. See the **loo** package vignettes for details.

### Usage

```
## S3 method for class 'measrfit'
loo_compare(x, ..., criterion = c("loo", "waic"), model_names = NULL)
```

### Arguments

| | |
|---|---|
| x | A measrfit object. |
| ... | Additional objects of class measrfit. |
| criterion | The name of the criterion to be extracted from the measrfit object for comparison. |
| model_names | Names given to each provided model in the comparison output. If NULL (the default), the names will be parsed from the names of the objects passed for comparison. |

### Value

The object returned by loo::loo_compare().

## mdm_data *MacReady & Dayton (1977) Multiplication Data*

### Description

This is a small data set of multiplication item responses. This data contains responses to 4 items from 142 respondents, which ask respondents to complete an integer multiplication problem.

### Usage

```
mdm_data

mdm_qmatrix
```

### Format

mdm_data is a tibble containing responses to multiplication items, as described in MacReady & Dayton (1977). There are 142 rows and 5 variables.

- respondent: Respondent identifier
- mdm1-mdm4: Dichotomous item responses to the 4 multiplication items

mdm_qmatrix is a tibble that identifies which skills are measured by each MDM item. This MDM data contains 4 items, all of which measure the skill of multiplication. The mdm_qmatrix correspondingly is made up of 4 rows and 2 variables.

- item: Item identifier, corresponds to mdm1-mdm4 in mdm_data
- multiplication: Dichotomous indicator for whether or not the multiplication skill is measured by each item. A value of 1 indicates the skill is measured by the item and a value of 0 indicates the skill is not measured by the item.

### References

MacReady, G. B., & Dayton, C. M. (1977). The use of probabilistic models in the assessment of mastery. *Journal of Educational Statistics, 2*(2), 99-120. doi:10.2307/1164802

---

## measrfit *Create a* measrfit *object*

---

### Description

Models fitted with **measr** are represented as a measrfit object. If a model is estimated with *Stan*, but not **measr**, a measrfit object can be created in order to access other functionality in **measr** (e.g., model fit, reliability).

### Usage

```
measrfit(
  data = list(),
  type = character(),
  prior = default_dcm_priors(type = type),
  stancode = character(),
  method = character(),
  algorithm = character(),
  backend = character(),
  model = NULL,
  respondent_estimates = list(),
  fit = list(),
  criteria = list(),
  reliability = list(),
  file = NULL,
  version = list(),
  class = character()
)
```

## Arguments

| | |
|---|---|
| `data` | The data and Q-matrix used to estimate the model. |
| `type` | The type of DCM that was estimated. |
| `prior` | A measrprior object containing information on the priors used in the model. |
| `stancode` | The model code in **Stan** language. |
| `method` | The method used to fit the model. |
| `algorithm` | The name of the algorithm used to fit the model. |
| `backend` | The name of the backend used to fit the model. |
| `model` | The fitted Stan model. This will object of class rstan::stanfit if `backend = "rstan"` and `CmdStanMCMC` if `backend = "cmdstanr"` was specified when fitting the model. |
| `respondent_estimates` | An empty list for adding estimated person parameters after fitting the model. |
| `fit` | An empty list for adding model fit information after fitting the model. |
| `criteria` | An empty list for adding information criteria after fitting the model. |
| `reliability` | An empty list for adding reliability information after fitting the model. |
| `file` | Optional name of a file which the model objects was saved to or loaded from. |
| `version` | The versions of **measr**, **Stan**, **rstan** and/or **cmdstanr** that were used to fit the model. |
| `class` | Additional classes to be added (e.g., `measrdcm` for a diagnostic classification model). |

## Value

A measrfit object.

## See Also

measrfit, `as_measrfit()`, `is_measrfit()`

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)

new_obj <- measrfit(
  data = rstn_mdm_lcdm$data,
  type = rstn_mdm_lcdm$type,
  prior = rstn_mdm_lcdm$prior,
  stancode = rstn_mdm_lcdm$stancode,
  method = rstn_mdm_lcdm$method,
  algorithm = rstn_mdm_lcdm$algorithm,
  backend = rstn_mdm_lcdm$backend,
  model = rstn_mdm_lcdm$model,
```

```
  respondent_estimates = rstn_mdm_lcdm$respondent_estimates,
  fit = rstn_mdm_lcdm$fit,
  criteria = rstn_mdm_lcdm$criteria,
  reliability = rstn_mdm_lcdm$reliability,
  file = rstn_mdm_lcdm$file,
  version = rstn_mdm_lcdm$version,
  class = "measrdcm"
)
```

---

`measrfit-class` *Class* `measrfit` *of models fitted with the* **measr** *package*

---

## Description

Models fitted with the **measr** package are represented as a `measrfit` object, which contains the posterior draws, Stan code, priors, and other relevant information.

## Slots

`data` The data and Q-matrix used to estimate the model.

`type` The type of DCM that was estimated.

`prior` A measrprior object containing information on the priors used in the model.

`stancode` The model code in **Stan** language.

`method` The method used to fit the model.

`algorithm` The name of the algorithm used to fit the model.

`backend` The name of the backend used to fit the model.

`model` The fitted Stan model. This will object of class rstan::stanfit if `backend = "rstan"` and CmdStanMCMC if `backend = "cmdstanr"` was specified when fitting the model.

`respondent_estimates` An empty list for adding estimated person parameters after fitting the model.

`fit` An empty list for adding model fit information after fitting the model.

`criteria` An empty list for adding information criteria after fitting the model.

`reliability` An empty list for adding reliability information after fitting the model.

`file` Optional name of a file which the model objects was saved to or loaded from.

`version` The versions of **measr**, **Stan**, **rstan** and/or **cmdstanr** that were used to fit the model.

## See Also

`measrfit()`, `as_measrfit()`, `is_measrfit()`

`measrprior` *Prior definitions for* **measr** *models*

## Description

Create prior definitions for classes of parameters, or specific parameters.

## Usage

```
measrprior(
  prior,
 class = c("structural", "intercept", "maineffect", "interaction", "slip", "guess"),
  coef = NA,
  lb = NA,
  ub = NA
)

prior(prior, ...)

prior_(prior, ...)

prior_string(prior, ...)
```

## Arguments

| | |
|---|---|
| `prior` | A character string defining a distribution in **Stan** language. A list of all distributions supported by **Stan** can be found in *Stan Language Functions Reference* at https://mc-stan.org/users/documentation/. |
| `class` | The parameter class. Defaults to `"intercept"`. Must be one of `"intercept"`, `"maineffect"`, `"interaction"` for the LCDM, or one of `"slip"` or `"guess"` for DINA or DINO models. |
| `coef` | Name of a specific parameter within the defined class. If not defined, the prior is applied to all parameters within the class. |
| `lb` | Lower bound for parameter restriction. Defaults to no restriction. |
| `ub` | Upper bound for parameter restriction. Defaults to no restriction. |
| `...` | Additional arguments passed to `measrprior()`. |

## Value

A tibble of class `measrprior`.

## Functions

- `prior()`: Alias of `measrprior()` which allows arguments to be specified as expressions without quotation marks.

- `prior_()`: Alias of `measrprior()` which allows arguments to be specified as one-sided formulas or wrapped in `base::quote()`.
- `prior_string()`: Alias of `measrprior()` which allows arguments to be specified as character strings.

## Examples

```
# Use alias functions to define priors without quotes, as formulas,
# or as character strings.
(prior1 <- prior(lognormal(0, 1), class = maineffect))

(prior2 <- prior_(~lognormal(0, 1), class = ~maineffect))

(prior3 <- prior_string("lognormal(0, 1)", class = "maineffect"))

identical(prior1, prior2)
identical(prior1, prior3)
identical(prior2, prior3)

# Define a prior for an entire class of parameters
prior(beta(5, 25), class = "slip")

# Or for a specific item (e.g., just the slipping parameter for item 7)
prior(beta(5, 25), class = "slip", coef = "slip[7]")
```

---

# measr_dcm *Fit Bayesian diagnostic classification models*

---

## Description

Estimate diagnostic classification models (DCMs; also known as cognitive diagnostic models) using 'Stan'. Models can be estimated using Stan's optimizer, or full Markov chain Monte Carlo (MCMC).

## Usage

```
measr_dcm(
  data,
  missing = NA,
  qmatrix,
  resp_id = NULL,
  item_id = NULL,
  type = c("lcdm", "dina", "dino", "crum"),
  max_interaction = Inf,
  attribute_structure = c("unconstrained", "independent"),
  method = c("mcmc", "optim"),
  prior = NULL,
  backend = getOption("measr.backend", "rstan"),
```

```
  file = NULL,
  file_refit = getOption("measr.file_refit", "never"),
  ...
)
```

## Arguments

| | |
|---|---|
| data | Response data. A data frame with 1 row per respondent and 1 column per item. |
| missing | An R expression specifying how missing data in data is coded (e.g., NA, ".", -99, etc.). The default is NA. |
| qmatrix | The Q-matrix. A data frame with 1 row per item and 1 column per attribute. All cells should be either 0 (item does not measure the attribute) or 1 (item does measure the attribute). |
| resp_id | Optional. Variable name of a column in data that contains respondent identifiers. NULL (the default) indicates that no identifiers are present in the data, and row numbers will be used as identifiers. |
| item_id | Optional. Variable name of a column in qmatrix that contains item identifiers. NULL (the default) indicates that no identifiers are present in the Q-matrix. In this case, the column names of data (excluding any column specified in resp_id) will be used as the item identifiers. NULL also assumes that the order of the rows in the Q-matrix is the same as the order of the columns in data (i.e., the item in row 1 of qmatrix is the item in column 1 of data, excluding resp_id). |
| type | Type of DCM to estimate. Must be one of lcdm, dina, dino, or crum. |
| max_interaction | If type = "lcdm", the highest level of interaction to estimate. The default is to estimate all possible interactions. For example, an item that measures 4 attributes would have 4 main effects, 6 two-way interactions, 4 three-way interactions, and 1 four-way interaction. Setting max_interaction = 2 would result in only estimating the main effects and two-way interactions, excluding the three- and four- way interactions. |
| attribute_structure | Structural model specification. Must be one of unconstrained, or independent. unconstrained makes no assumptions about the relationships between attributes, whereas independent assumes that proficiency statuses on attributes are independent of each other. |
| method | Estimation method. Options are "mcmc", which uses Stan's sampling method, or "optim", which uses Stan's optimizer. |
| prior | A measrprior object. If NULL, default priors are used, as specified by default_dcm_priors(). |
| backend | Character string naming the package to use as the backend for fitting the Stan model. Options are "rstan" (the default) or "cmdstanr". Can be set globally for the current R session via the "measr.backend" option (see options()). Details on the **rstan** and **cmdstanr** packages are available at https://mc-stan.org/rstan/ and https://mc-stan.org/cmdstanr/, respectively. |
| file | Either NULL (the default) or a character string. If a character string, the fitted model object is saved as an .rds object using saveRDS() using the supplied character string. The .rds extension is automatically added. If the specified file |

already exists, **measr** will load the previously saved model. Unless `file_refit` is specified, the model will not be refit.

`file_refit` Controls when a saved model is refit. Options are `"never"`, `"always"`, and `"on_change"`. Can be set globally for the current R session via the "measr.file_refit" option (see `options()`).

- For `"never"` (the default), the fitted model is always loaded if the `file` exists, and model fitting is skipped.
- For `"always"`, the model is always refitted, regardless of whether or not `file` exists.
- For `"on_change"`, the model will be refit if the `data`, `prior`, or `method` specified are different from that in the saved `file`.

`...` Additional arguments passed to Stan.

- For `backend = "rstan"`, arguments are passed to `rstan::sampling()` or `rstan::optimizing()`.
- For `backend = "cmdstanr"`, arguments are passed to the sample or optimize methods of the CmdStanModel class.

## Value

A measrfit object.

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)
```

---

# measr_examples *Determine if code is executed interactively or in pkgdown*

---

## Description

Used for determining examples that shouldn't be run on CRAN, but can be run for the pkgdown website.

## Usage

```
measr_examples()
```

## Value

A logical value indicating whether or not the examples should be run.

## Examples

```
measr_examples()
```

---

`measr_extract` *Extract components of a* `measrfit` *object.*

---

## Description

Extract components of a `measrfit` object.

Extract components of an estimated diagnostic classification model

## Usage

```
measr_extract(model, ...)

## S3 method for class 'measrdcm'
measr_extract(model, what, ...)
```

## Arguments

| | |
|---|---|
| `model` | The estimated to extract information from. |
| `...` | Additional arguments passed to each extract method.<br>• `ppmc_interval`:<br>For `what = "odds_ratio_flags"` and `what = "conditional_prob_flags"`, the compatibility interval used for determining model fit flags to return. For example, a `ppmc_interval` of 0.95 (the default) will return any PPMCs where the posterior predictive $p$-value (ppp) is less than 0.025 or greater than 0.975.<br>• `agreement`:<br>For `what = "classification_reliability"`, additional measures of agreement to include. By default, the classification accuracy and consistency metrics defined Johnson & Sinharay (2018) are returned. Additional metrics that can be specified to `agreement` are Goodman & Kruskal's lambda (`lambda`), Cohen's kappa (`kappa`), Youden's statistic (`youden`), the tetrachoric correlation (`tetra`), true positive rate (`tp`), and the true negative rate (`tn`).<br>For `what = "probability_reliability"`, additional measures of agreement to include. By default, the informational reliability index defined by Johnson & Sinharay (2020) is returned. Additional metrics that can be specified to `agreement` are the point biserial reliability index (`bs`), parallel forms reliability index (`pf`), and the tetrachoric reliability index (`tb`), which was originally defined by Templin & Bradshaw (2013). |
| `what` | Character string. The information to be extracted. See details for available options. |

## Details

For diagnostic classification models, we can extract the following information:

- `item_param`: The estimated item parameters. This shows the name of the parameter, the class of the parameter, and the estimated value.
- `strc_param`: The estimated structural parameters. This is the base rate of membership in each class. This shows the class pattern and the estimated proportion of respondents in each class.
- `prior`: The priors used when estimating the model.
- `classes`: The possible classes or profile patterns. This will show the class label (i.e., the pattern of proficiency) and the attributes included in each class.
- `class_prob`: The probability that each respondent belongs to class (i.e., has the given pattern of proficiency).
- `attribute_prob`: The proficiency probability for each respondent and attribute.
- `m2`: The $M_2$ fit statistic. See `fit_m2()` for details. Model fit information must first be added to the model using `add_fit()`.
- `rmsea`: The root mean square error of approximation (RMSEA) fit statistic and associated confidence interval. See `fit_m2()` for details. Model fit information must first be added to the model using `add_fit()`.
- `srmsr`: The standardized root mean square residual (SRMSR) fit statistic. See `fit_m2()` for details. Model fit information must first be added to the model using `add_fit()`.
- `ppmc_raw_score`: The observed and posterior predicted chi-square statistic for the raw score distribution. See `fit_ppmc()` for details. Model fit information must first be added to the model using `add_fit()`.
- `ppmc_conditional_prob`: The observed and posterior predicted conditional probabilities of each class providing a correct response to each item. See `fit_ppmc()` for details. Model fit information must first be added to the model using `add_fit()`.
- `ppmc_conditional_prob_flags`: A subset of the PPMC conditional probabilities where the *ppp* is outside the specified `ppmc_interval`.
- `ppmc_odds_ratio`: The observed and posterior predicted odds ratios of each item pair. See `fit_ppmc()` for details. Model fit information must first be added to the model using `add_fit()`.
- `ppmc_odds_ratio_flags`: A subset of the PPMC odds ratios where the *ppp* is outside the specified `ppmc_interval`.
- `ppmc_pvalue`: The observed and posterior predicted proportion of correct responses to each item. See `fit_ppmc()` for details.
- `ppmc_pvalue_flags`: A subset of the PPMC proportion correct values where the *ppp* is outside the specified `ppmc_interval`.
- `loo`: The leave-one-out cross validation results. See `loo::loo()` for details. The information criterion must first be added to the model using `add_criterion()`.
- `waic`: The widely applicable information criterion results. See `loo::waic()` for details. The information criterion must first be added to the model using `add_criterion()`.
- `pattern_reliability`: The accuracy and consistency of the overall attribute profile classification, as described by Cui et al. (2012). Reliability information must first be added to the model using `add_reliability()`.

- `classification_reliability`: The classification accuracy and consistency for each attribute, using the metrics described by Johnson & Sinharay (2018). Reliability information must first be added to the model using `add_reliability()`.
- `probability_reliability`: Reliability estimates for the probability of proficiency on each attribute, as described by Johnson & Sinharay (2020). Reliability information must first be added to the model using `add_reliability()`.

## Value

The extracted information. The specific structure will vary depending on what is being extracted, but usually the returned object is a tibble with the requested information.

## Methods (by class)

- `measr_extract(measrdcm)`: Extract components of an estimated diagnostic classification model.

## References

Cui, Y., Gierl, M. J., & Chang, H.-H. (2012). Estimating classification consistency and accuracy for cognitive diagnostic assessment. *Journal of Educational Measurement, 49*(1), 19-38. doi:10.1111/j.17453984.2011.00158.x

Johnson, M. S., & Sinharay, S. (2018). Measures of agreement to assess attribute-level classification accuracy and consistency for cognitive diagnostic assessments. *Journal of Educational Measurement, 55*(4), 635-664. doi:10.1111/jedm.12196

Johnson, M. S., & Sinharay, S. (2020). The reliability of the posterior probability of skill attainment in diagnostic classification models. *Journal of Educational and Behavioral Statistics, 45*(1), 5-31. doi:10.3102/1076998619864550

Templin, J., & Bradshaw, L. (2013). Measuring the reliability of diagnostic classification model examinee estimates. *Journal of Classification, 30*(2), 251-275. doi:10.1007/s0035701391294

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)

measr_extract(rstn_mdm_lcdm, "strc_param")
```

`model_evaluation` *Add model evaluation metrics model objects*

## Description

Add model evaluation metrics to fitted model objects. These functions are wrappers around other functions that compute the metrics. The benefit of using these wrappers is that the model evaluation metrics are saved as part of the model object so that time-intensive calculations do not need to be repeated. See Details for specifics.

## Usage

```
add_criterion(
  x,
  criterion = c("loo", "waic"),
  overwrite = FALSE,
  save = TRUE,
  ...,
  r_eff = NA
)

add_reliability(x, overwrite = FALSE, save = TRUE)

add_fit(
  x,
  method = c("m2", "ppmc"),
  overwrite = FALSE,
  save = TRUE,
  ...,
  ci = 0.9
)

add_respondent_estimates(
  x,
  probs = c(0.025, 0.975),
  overwrite = FALSE,
  save = TRUE
)
```

## Arguments

| | |
|---|---|
| `x` | A measrfit object. |
| `criterion` | A vector of criteria to calculate and add to the model object. |
| `overwrite` | Logical. Indicates whether specified elements that have already been added to the estimated model should be overwritten. Default is `FALSE`. |

| | |
|---|---|
| save | Logical. Only relevant if a file was specified in the measrfit object passed to x. If TRUE (the default), the model is re-saved to the specified file when new criteria are added to the R object. If FALSE, the new criteria will be added to the R object, but the saved file will not be updated. |
| ... | Additional arguments passed relevant methods. See Details. |
| r_eff | Vector of relative effective sample size estimates for the likelihood (exp(log_lik)) of each observation. This is related to the relative efficiency of estimating the normalizing term in self-normalized importance sampling when using posterior draws obtained with MCMC. If MCMC draws are used and r_eff is not provided then the reported PSIS effective sample sizes and Monte Carlo error estimates can be over-optimistic. If the posterior draws are (near) independent then r_eff=1 can be used. r_eff has to be a scalar (same value is used for all observations) or a vector with length equal to the number of observations. The default value is 1. See the relative_eff() helper functions for help computing r_eff. |
| method | A vector of model fit methods to evaluate and add to the model object. |
| ci | The confidence interval for the RMSEA, computed from the M2 |
| probs | The percentiles to be computed by the [stats::quantile()] function to summarize the posterior distributions of each person parameter. Only relevant if method = "mcmc" was used to estimate the model. |

## Details

For add_respondent_estimates(), estimated person parameters are added to the $respondent_estimates element of the fitted model.

For add_fit(), model and item fit information are added to the $fit element of the fitted model. This function wraps fit_m2() to calculate the $M_2$ statistic (Hansen et al., 2016; Liu et al., 2016) and/or fit_ppmc() to calculate posterior predictive model checks (Park et al., 2015; Sinharay & Almond, 2007; Sinharay et al., 2006; Thompson, 2019), depending on which methods are specified. Additional arguments supplied to ... are passed to fit_ppmc().

For add_criterion(), relative fit criteria are added to the $criteria element of the fitted model. This function wraps loo() and/or waic(), depending on which criteria are specified, to calculate the leave-one-out (LOO; Vehtari et al., 2017) and/or widely applicable information criteria (WAIC; Watanabe, 2010) to fitted model objects. Additional arguments supplied to ... are passed to loo::loo.array() or loo::waic.array().

For add_reliability(), reliability information is added to the $reliability element of the fitted model. Pattern level reliability is described by Cui et al. (2012). Classification reliability and posterior probability reliability are described by Johnson & Sinharay (2018, 2020), respectively. This function wraps reliability().

## Value

A modified measrfit object with the corresponding slot populated with the specified information.

## References


Cui, Y., Gierl, M. J., & Chang, H.-H. (2012). Estimating classification consistency and accuracy for cognitive diagnostic assessment. *Journal of Educational Measurement, 49*(1), 19-38. doi:10.1111/j.17453984.2011.00158.x

Hansen, M., Cai, L., Monroe, S., & Li, Z. (2016). Limited-information goodness-of-fit testing of diagnostic classification item response models. *British Journal of Mathematical and Statistical Psychology, 69*(3), 225-252. doi:10.1111/bmsp.12074

Johnson, M. S., & Sinharay, S. (2018). Measures of agreement to assess attribute-level classification accuracy and consistency for cognitive diagnostic assessments. *Journal of Educational Measurement, 55*(4), 635-664. doi:10.1111/jedm.12196

Johnson, M. S., & Sinharay, S. (2020). The reliability of the posterior probability of skill attainment in diagnostic classification models. *Journal of Educational and Behavioral Statistics, 45*(1), 5-31. doi:10.3102/1076998619864550

Liu, Y., Tian, W., & Xin, T. (2016). An application of $M_2$ statistic to evaluate the fit of cognitive diagnostic models. *Journal of Educational and Behavioral Statistics, 41*(1), 3-26. doi:10.3102/1076998615621293

Park, J. Y., Johnson, M. S., Lee, Y-S. (2015). Posterior predictive model checks for cognitive diagnostic models. *International Journal of Quantitative Research in Education, 2*(3-4), 244-264. doi:10.1504/IJQRE.2015.071738

Sinharay, S., & Almond, R. G. (2007). Assessing fit of cognitive diagnostic models. *Educational and Psychological Measurement, 67*(2), 239-257. doi:10.1177/0013164406292025

Sinharay, S., Johnson, M. S., & Stern, H. S. (2006). Posterior predictive assessment of item response theory models. *Applied Psychological Measurement, 30*(4), 298-321. doi:10.1177/0146621605285517

Thompson, W. J. (2019). *Bayesian psychometrics for diagnostic assessments: A proof of concept* (Research Report No. 19-01). University of Kansas; Accessible Teaching, Learning, and Assessment Systems. doi:10.35542/osf.io/jzqs8

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing, 27*(5), 1413-1432. doi:10.1007/s1122201696964

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of Machine Learning Research, 11*(116), 3571-3594. https://jmlr.org/papers/v11/watanabe10a.html


## Examples

```
cmds_mdm_dina <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "dina",
  method = "optim", seed = 63277, backend = "rstan",
  prior = c(prior(beta(5, 17), class = "slip"),
            prior(beta(5, 17), class = "guess"))
)

cmds_mdm_dina <- add_reliability(cmds_mdm_dina)
cmds_mdm_dina <- add_fit(cmds_mdm_dina, method = "m2")
cmds_mdm_dina <- add_respondent_estimates(cmds_mdm_dina)
```

predict.measrdcm *Posterior draws of respondent proficiency*

## Description

Calculate posterior draws of respondent proficiency. Optionally retain all posterior draws or return only summaries of the distribution for each respondent.

## Usage

```
## S3 method for class 'measrdcm'
predict(
  object,
  newdata = NULL,
  resp_id = NULL,
  missing = NA,
  summary = TRUE,
  probs = c(0.025, 0.975),
  force = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| object | An object of class measrdcm. Generated from measr_dcm(). |
| newdata | Optional new data. If not provided, the data used to estimate the model is scored. If provided, newdata should be a data frame with 1 row per respondent and 1 column per item. All items that appear in newdata should appear in the data used to estimate object. |
| resp_id | Optional. Variable name of a column in newdata that contains respondent identifiers. NULL (the default) indicates that no identifiers are present in the data, and row numbers will be used as identifiers. If newdata is not specified and the data used to estimate the model is scored, the resp_id is taken from the original data. |
| missing | An R expression specifying how missing data in data is coded (e.g., NA, ".", -99, etc.). The default is NA. |
| summary | Should summary statistics be returned instead of the raw posterior draws? Only relevant if the model was estimated with method = "mcmc". Default is FALSE. |
| probs | The percentiles to be computed by the [stats::quantile()] function. Only relevant if the model was estimated with method = "mcmc". Only used if summary is TRUE. |
| force | If respondent estimates have already been added to the model object with add_respondent_estimates() should they be recalculated. Default is FALSE. |
| ... | Unused. |

### Value

A list with two elements: `class_probabilities` and `attribute_probabilities`.

If summary is `FALSE`, each element is a tibble with the number of rows equal to the number of draws in `object` with columns: `.chain`, `.iteration`, `.draw`, the respondent identifier, and one column of probabilities for each of the possible classes.

If summary is `TRUE`, each element is a tibble with one row per respondent and class or attribute, and columns of the respondent identifier, `class` or `attribute`, `mean`, and one column for every value specified in `probs`.

---

## `reliability` *Estimate the reliability of psychometric models*

---

### Description

For diagnostic classification models, reliability can be estimated at the pattern or attribute level. Pattern-level reliability represents the classification consistency and accuracy of placing students into an overall mastery profile. Rather than an overall profile, attributes can also be scored individually. In this case, classification consistency and accuracy should be evaluated for each individual attribute, rather than the overall profile. This is referred to as the *maximum a posteriori* (MAP) reliability. Finally, it may be desirable to report results as the probability of proficiency or mastery on each attribute instead of a proficient/not proficient classification. In this case, the reliability of the posterior probability should be reported. This is the *expected a posteriori* (EAP) reliability.

### Usage

```
reliability(model, ...)

## S3 method for class 'measrdcm'
reliability(model, ..., threshold = 0.5, force = FALSE)
```

### Arguments

| | |
|---|---|
| `model` | The estimated model to be evaluated. |
| `...` | Unused. For future extensions. |
| `threshold` | For `map_reliability`, the threshold applied to the attribute-level probabilities for determining the binary attribute classifications. |
| `force` | If reliability information has already been added to the model object with `add_reliability()`, should it be recalculated. Default is `FALSE`. |

### Details

The pattern-level reliability (`pattern_reliability`) statistics are described in Cui et al. (2012). Attribute-level classification reliability statistics (`map_reliability`) are described in Johnson & Sinharay (2018). Reliability statistics for the posterior mean of the skill indicators (i.e., the mastery or proficiency probabilities; `eap_reliability`) are described in Johnson & Sinharay (2019).

## Value

For class `measrdcm`, a list with 3 elements:

- `pattern_reliability`: The pattern-level accuracy (`p_a`) and consistency (`p_c`) described by Cui et al. (2012).
- `map_reliability`: A list with 2 elements: `accuracy` and `consistency`, which include the attribute-level classification reliability statistics described by Johnson & Sinharay (2018).
- `eap_reliability`: The attribute-level posterior probability reliability statistics described by Johnson & Sinharay (2020).

## Methods (by class)

- `reliability(measrdcm)`: Reliability measures for diagnostic classification models.

## References

Cui, Y., Gierl, M. J., & Chang, H.-H. (2012). Estimating classification consistency and accuracy for cognitive diagnostic assessment. *Journal of Educational Measurement, 49*(1), 19-38. doi:10.1111/j.17453984.2011.00158.x

Johnson, M. S., & Sinharay, S. (2018). Measures of agreement to assess attribute-level classification accuracy and consistency for cognitive diagnostic assessments. *Journal of Educational Measurement, 55*(4), 635-664. doi:10.1111/jedm.12196

Johnson, M. S., & Sinharay, S. (2020). The reliability of the posterior probability of skill attainment in diagnostic classification models. *Journal of Educational and Behavioral Statistics, 45*(1), 5-31. doi:10.3102/1076998619864550

## Examples

```
rstn_mdm_lcdm <- measr_dcm(
  data = mdm_data, missing = NA, qmatrix = mdm_qmatrix,
  resp_id = "respondent", item_id = "item", type = "lcdm",
  method = "optim", seed = 63277, backend = "rstan"
)

reliability(rstn_mdm_lcdm)
```

---

# waic.measrfit *Widely applicable information criterion (WAIC)*

---

## Description

A `loo::waic()` method that is customized for `measrfit` objects. This is a simple wrapper around `loo::waic.array()`. See the **loo** package vignettes for details.

## Usage

```
## S3 method for class 'measrfit'
waic(x, ..., force = FALSE)
```

## Arguments

| | |
|---|---|
| x | A measrfit object. |
| ... | Additional arguments passed to loo::waic.array(). |
| force | If the WAIC criterion has already been added to the model object with add_criterion(), should it be recalculated. Default is FALSE. |

## Value

The object returned by loo::waic.array().

# Index