

Package: IRTC (via r-universe)

September 26, 2026

Type Package

Title Marginal Maximum Likelihood Estimation for Item Response Models

Version 1.1.2

Date 2026-08-27

Description Self-contained marginal maximum likelihood (MML) estimation for unidimensional and multidimensional item response models, including the Rasch / one-parameter logistic, partial credit, rating scale, two-parameter logistic and generalised partial credit models, with latent regression, multiple groups and case weights. A parallelised, dimension-factorised streaming estimation engine supports large between-item (simple-structure) multidimensional models with bounded memory and an opt-in controlled-accuracy quadrature mode that reports a measured approximation error. A usability layer serves non-specialists and automated pipelines: one-stop estimation from common file formats ('Excel', delimited text, 'SPSS', 'Stata', 'SAS') with automatic cleaning and answer-key scoring, pre-estimation data checks, classical item statistics and item fit, plain-language quality ratings, bilingual (English/Chinese) output, spreadsheet exports for item banking and cross-year linking, audience-specific 'Word'/'HTML' reports, and machine-readable results with structured error conditions. Methods follow Adams, Wilson and Wang (1997) <doi:10.1177/0146621697211001>.

License GPL (>= 2)

Copyright See file inst/COPYRIGHTS.

URL <https://github.com/weandata/IRTC>

BugReports <https://github.com/weandata/IRTC/issues>

Depends R (>= 3.5.0)

Imports Rcpp, stats, utils, methods, MASS, tools, graphics, grDevices

Suggests testthat (>= 3.0.0), mytnorm, sfsmisc, readxl, writextl, haven, openxlsx, officer, jsonlite

LinkingTo Rcpp, RcppArmadillo
Encoding UTF-8
NeedsCompilation yes
Repository <https://weiandata.r-universe.dev>
Date/Publication 2026-08-27 04:40:29 UTC
RemoteUrl <https://github.com/weiandata/irtc>
RemoteRef HEAD
RemoteSha e44f0bd4d931f90ffcc404dd300305349fee134c

Contents

IRTC-package	2
data.gpcm	3
data.mc	4
data.sim.rasch	4
irtc	5
irtc.mml	7
irtc.mml.2pl	10
irtc_audit_scoring	13
irtc_check_data	14
irtc_ckt	15
irtc_excel	17
irtc_itemfit	18
irtc_lang	19
irtc_quality	20
irtc_read	21
irtc_read_q	23
irtc_report	24
irtc_results	26
irtc_score	27
plain_summary	29
plot.irtc	30
Index	31

IRTC-package	<i>Marginal Maximum Likelihood Estimation for Item Response Models</i>
--------------	--

Description

A self-contained marginal maximum likelihood estimation package for unidimensional and multi-dimensional item response models. It supports Rasch / 1PL, partial credit, rating scale, 2PL and generalized partial credit models, latent regression, multiple groups, case weights, and a bounded-memory streaming engine for large simple-structure models.

Details

Two entry points are provided.

For statisticians, the expert functions `irtc.mml` and `irtc.mml.2pl` return an object of class `irtc` with `summary`, `print`, `logLik`, `anova` and `plot` methods.

For non-specialists and automated pipelines, the one-stop function `irtc` reads data from common file formats (`irtc_read`), scores raw responses (`irtc_score`), checks the data (`irtc_check_data`) and enriches the fitted model with classical statistics (`irtc_ctt`), item fit (`irtc_itemfit`) and plain-language quality ratings (`irtc_quality`). Results can be exported as three Excel workbooks (`irtc_excel`), audience-specific Word/HTML reports (`irtc_report`), a layered text summary (`plain_summary`) or machine-readable structures (`irtc_results`, `irtc_json`; schema reference in `inst/llms.txt`). User-facing output is bilingual; see `irtc_lang`.

Author(s)

Kunxiang Ma and WEIAN DATA TECH (Beijing) Co., Ltd.

See Also

`irtc`, `irtc.mml`, `irtc.mml.2pl`

data.gpcm

Simulated Polytomous Item Response Data

Description

A simulated dataset of polytomous item responses suitable for fitting partial credit and generalised partial credit models, provided as an example dataset for **IRTC**.

Usage

```
data(data.gpcm)
```

Format

A data frame with 392 persons (rows) and 3 polytomously scored items (columns `Comfort`, `Work`, `Benefit`), each scored in four ordered categories 0, 1, 2, 3.

Details

The data were simulated from a partial credit model with three step parameters per item. The generating script is `'scripts/gen_data.R'` in the package sources.

Source

Simulated by IRTC. See `'scripts/gen_data.R'` in the package sources for the reproducible generating code.

data.mc	<i>Simulated Multiple-Choice Item Response Data</i>
---------	---

Description

A simulated dataset of multiple-choice item responses provided in both raw and scored form, provided as an example dataset for **IRTC**.

Usage

```
data(data.mc)
```

Format

A list with two components:

raw A character matrix (143 persons x 30 items) of raw multiple-choice responses coded "A"–"D", with "9" denoting an omitted or not-reached response.

scored A data frame of the corresponding dichotomously scored (0/1) responses, where 1 marks the keyed (correct) option.

Details

Correctness was simulated under a Rasch model; raw responses were then built from a randomly assigned answer key with distractors drawn at random. The generating script is 'scripts/gen_data.R' in the package sources.

Source

Simulated by IRTC. See 'scripts/gen_data.R' in the package sources for the reproducible generating code.

data.sim.rasch	<i>Simulated Dichotomous Rasch Data</i>
----------------	---

Description

A simulated dataset of dichotomous (0/1) item responses generated under the Rasch model, provided as an example dataset for **IRTC**.

Usage

```
data(data.sim.rasch)
```

Format

A matrix with 2000 persons (rows) and 40 dichotomous items (columns), with entries coded 0 or 1.

Details

The data were simulated with person abilities drawn from a standard normal distribution and item difficulties spread evenly over $[-2.2, 2.2]$. The generating script is ‘scripts/gen_data.R’ in the package sources.

Source

Simulated by IRTC. See ‘scripts/gen_data.R’ in the package sources for the reproducible generating code.

 irtc

One-Stop Item Response Model Estimation

Description

Reads data from a file or R object, cleans it, optionally scores raw multiple-choice responses against an answer key, checks it, estimates the requested item response model and enriches the result with classical item statistics, item fit and plain-language quality ratings.

Designed for non-specialists and automated callers; professional users keep full control because all additional arguments are passed unchanged to [irtc.mm1](#) (for "1PL", "PCM", "PCM2", "RSM") or [irtc.mm1.2pl](#) (for "2PL", "GPCM").

Usage

```
irtc(data, model, key = NULL, rules = NULL, q = NULL,
      on_mismatch = c("warn", "error"),
      rare_categories = c("collapse", "prior"), id = NULL, weights = NULL,
      sheet = 1, missing_codes = c(-9, -99, 99, 999), check = TRUE,
      quality = TRUE, verbose = TRUE, ...)
```

Arguments

data	A file path (.xlsx, .xls, .csv, .tsv, .txt, .dat, .sav, .por, .dta, .sas7bdat, .xpt), a data frame, a matrix, or an object returned by irtc.read .
model	Required. One of "1PL" (alias "Rasch"), "2PL", "PCM", "PCM2", "RSM", "GPCM". Case-insensitive.
key	An optional answer key for raw responses: a named vector or a key-file path (optionally with a partial-answer column); see irtc.score .
rules	An optional partial-credit scoring table or a path to one; see irtc.score .
q	An optional Q (item-by-dimension) matrix: a file path, data frame, matrix, or an object from irtc.read.q . The Q matrix is aligned with the data via irtc.align.q and passed to the estimation as Q; its dimension names label the person-level output. If Q is also supplied through ..., the explicit Q wins and a warning (W419) is raised.
on_mismatch	How to handle items that differ between the Q matrix and the data; see irtc.align.q .

<code>rare_categories</code>	How to handle score categories nobody reached (nobody fully correct / partially correct / wrong). "collapse" (default): merge unobserved categories into their lower neighbour; the mapping is logged (W425) and annotated in the item output (categories_unobserved, categories_collapsed, declared vs observed maximum score). "prior": keep the observed category structure; thresholds of in-range empty categories get a $N(0, 4)$ prior (models "1PL", "PCM") or are fixed at the prior mean 0 (models "2PL", "GPCM", whose estimation core does not evaluate priors); such thresholds are marked prior_dominated in the output. Categories above the observed maximum have no parameter in either mode and are reported as a reduced maximum score. Items answered by nobody are removed (W410) but keep an annotated row in <code>irtc_results</code> .
<code>id</code>	An optional person identifier column (name or index); passed to <code>irtc_read</code> .
<code>weights</code>	An optional sampling weight column (name or index); passed to <code>irtc_read</code> . Common weight column names are also detected automatically. The weights are used as pweights in the estimation; if pweights is also supplied through ..., the explicit pweights wins and a warning (W418) is raised.
<code>sheet</code>	Excel sheet to read (number or name).
<code>missing_codes</code>	Numeric values recoded to NA during cleaning; see <code>irtc_read</code> .
<code>check</code>	Logical: stop with a structured error if <code>irtc_check_data</code> finds any error-level issue?
<code>quality</code>	Logical: attach CTT statistics, item fit and quality ratings to the result?
<code>verbose</code>	Logical: print estimation progress?
<code>...</code>	Further arguments passed to <code>irtc.mml</code> or <code>irtc.mml.2pl</code> (e.g. group, pweights, Q, control).

Details

Items without any observed responses or without variance are removed before estimation (with a warning of class `irtc_warning_estimation`). All errors raised by the usability layer are structured conditions with fields code, reason and fix; see the package file `inst/llms.txt` for the full list of codes.

The language of user-facing output is controlled by `options(irtc.lang = "zh")` (default) or `"en"`.

Value

An object of class `irtc` (see `irtc.mml`) with an additional list component `usability` containing model, `data_log` (cleaning log), `check` (the `irtc_check_data` result), `removed_items`, `ctt`, `itemfit` and `quality`.

See Also

`irtc_read`, `irtc_check_data`, `plain_summary`, `irtc_excel`, `irtc_report`, `irtc_results`

Examples

```
set.seed(1)
theta <- rnorm(150)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(150) < plogis(theta - b))
}))
colnames(resp) <- paste0("I", 1:5)

mod <- irtc(resp, model = "1PL", verbose = FALSE)
plain_summary(mod, lang = "en")
mod$usability$quality
```

irtc.mml	<i>Marginal Maximum Likelihood Estimation of Fixed-Slope Item Response Models</i>
----------	---

Description

Estimates unidimensional and multidimensional fixed-slope item response models by marginal maximum likelihood (MML). Supported models include the Rasch / 1PL model, the partial credit model ("PCM", "PCM2") and the rating scale model ("RSM"). Latent regression on covariates, multiple groups, case weights and user-specified design matrices are supported.

Usage

```
irtc.mml(resp, Y = NULL, group = NULL, irtmodel = "1PL",
  formulaY = NULL, dataY = NULL, ndim = 1, pid = NULL,
  xsi.fixed = NULL, xsi.inits = NULL, beta.fixed = NULL, beta.inits = NULL,
  variance.fixed = NULL, variance.inits = NULL, est.variance = TRUE,
  constraint = "cases", A = NULL, B = NULL, B.fixed = NULL, Q = NULL,
  est.slopegroups = NULL, E = NULL, pweights = NULL,
  userfct.variance = NULL, variance.Npars = NULL, item.elim = TRUE,
  verbose = TRUE, control = list())

## S3 method for class 'irtc'
summary(object, file = NULL, ...)

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

## S3 method for class 'irtc'
logLik(object, ...)

## S3 method for class 'irtc'
anova(object, ...)
```

Arguments

<code>resp</code>	A data frame or matrix of polytomous or dichotomous item responses. Missing responses are coded NA.
<code>Y</code>	An optional matrix of covariates for the latent regression.
<code>group</code>	An optional vector of integer group identifiers for multiple-group estimation.
<code>irtmodel</code>	The estimated item response model. One of "1PL", "PCM", "PCM2" or "RSM".
<code>formulaY</code>	An optional formula for the latent regression in conjunction with <code>dataY</code> .
<code>dataY</code>	An optional data frame of covariates used together with <code>formulaY</code> .
<code>ndim</code>	The number of latent dimensions (used when no <code>Q</code> or <code>B</code> is supplied).
<code>pid</code>	An optional vector of person identifiers.
<code>xsi.fixed</code>	An optional matrix with two columns for fixing item parameters: parameter index and fixed value.
<code>xsi.inits</code>	An optional matrix of initial item parameter values.
<code>beta.fixed</code>	An optional matrix for fixing regression coefficients, or FALSE to estimate all.
<code>beta.inits</code>	An optional matrix of initial regression coefficients.
<code>variance.fixed</code>	An optional matrix for fixing entries of the covariance matrix.
<code>variance.inits</code>	An optional initial covariance matrix.
<code>est.variance</code>	Logical: should the covariance matrix be estimated?
<code>constraint</code>	Identification constraint, either "cases" or "items".
<code>A</code>	An optional design array for the item intercepts.
<code>B</code>	An optional design array for the (fixed) item slopes / scoring.
<code>B.fixed</code>	An optional matrix for fixing entries of the loading array <code>B</code> .
<code>Q</code>	An optional loading (Q-) matrix mapping items to dimensions.
<code>est.slopegroups</code>	An optional vector grouping items with common slope design (advanced use).
<code>E</code>	An optional design array used together with <code>est.slopegroups</code> .
<code>pweights</code>	An optional vector of person (case) weights.
<code>userfct.variance</code>	An optional user function imposing constraints on the covariance matrix.
<code>variance.Npars</code>	The number of estimated parameters in a user-constrained covariance matrix.
<code>item.elim</code>	Logical: should items with no observed responses in a category be eliminated?
<code>verbose</code>	Logical: print progress during estimation?
<code>control</code>	A list of control arguments for the estimation (e.g. <code>maxiter</code> , <code>conv</code> , <code>snodes</code> , <code>nodes</code>). Performance-related elements: <code>n_threads</code> (threads for the parallel E-step; default and maximum = 2), <code>fast</code> (logical, default FALSE; opt-in approximate mode that prunes negligible-weight quadrature nodes), and <code>fast_threshold</code> (pruning cutoff, default 1e-6).
<code>object, x</code>	An object of class <code>irtc</code> as produced by <code>irtc.mml</code> .
<code>file</code>	An optional file name; if supplied, summary output is written there.
<code>...</code>	Further arguments passed to methods.

Value

An object of class `irtc`, a list with (among others) the components:

<code>xsi</code>	A data frame of estimated item parameters and standard errors.
<code>beta</code>	The estimated latent regression coefficients.
<code>variance</code>	The estimated covariance matrix of the latent dimensions.
<code>person</code>	A data frame of person parameter estimates (EAP) and related statistics.
<code>EAP.rel</code>	The EAP reliability per dimension.
<code>deviance</code>	The deviance ($-2 \log L$) at the solution.
<code>ic</code>	A list of information criteria (AIC, BIC, ...).
<code>CALL</code>	The matched call.

References

Adams, R. J., Wilson, M., & Wang, W. C. (1997). The multidimensional random coefficients multinomial logit model. *Applied Psychological Measurement*, 21(1), 1–23. doi:10.1177/0146621697211001

See Also

[irtc.mml.2pl](#) for models with estimated item slopes.

Examples

```
#####
# EXAMPLE 1: Unidimensional Rasch model
#####

data(data.sim.rasch)
mod1 <- irtc.mml(resp = data.sim.rasch)
summary(mod1)

#####
# EXAMPLE 2: Partial credit model
#####

data(data.gpcm)
mod2 <- irtc.mml(resp = data.gpcm, irtmodel = "PCM")
summary(mod2)
```

irtc.mml.2pl	<i>Marginal Maximum Likelihood Estimation of Slope (Discrimination) Item Response Models</i>
--------------	--

Description

Estimates unidimensional and multidimensional item response models with estimated item slopes (discriminations) by marginal maximum likelihood (MML). Supported models are the two-parameter logistic model ("2PL"), the generalized partial credit model ("GPCM"), their design-matrix and slope-group variants ("GPCM.design", "2PL.groups", "GPCM.groups").

Usage

```
irtc.mml.2pl(resp, Y = NULL, group = NULL, irtmodel = "2PL",
  formulaY = NULL, dataY = NULL, ndim = 1, pid = NULL,
  xsi.fixed = NULL, xsi.inits = NULL, beta.fixed = NULL, beta.inits = NULL,
  variance.fixed = NULL, variance.inits = NULL, est.variance = FALSE,
  A = NULL, B = NULL, B.fixed = NULL, Q = NULL, est.slopegroups = NULL,
  E = NULL, gamma.init = NULL, pweights = NULL, userfct.variance = NULL,
  variance.Npars = NULL, item.elim = TRUE, verbose = TRUE,
  method = c("auto", "grid", "streaming"), control = list())
```

Arguments

resp	A data frame or matrix of item responses (NA = missing).
Y	Optional covariate matrix for the latent regression.
group	Optional integer vector of group identifiers for multiple groups.
irtmodel	The estimated model: one of "2PL", "GPCM", "GPCM.design", "2PL.groups", "GPCM.groups".
formulaY	Optional formula for the latent regression (with dataY).
dataY	Optional covariate data frame used with formulaY.
ndim	Number of latent dimensions (when no Q/B is given).
pid	Optional vector of person identifiers.
xsi.fixed	Optional matrix (index, value) fixing item intercept parameters.
xsi.inits	Optional matrix of initial item parameters.
beta.fixed	Optional matrix fixing regression coefficients, or FALSE.
beta.inits	Optional matrix of initial regression coefficients.
variance.fixed	Optional matrix fixing covariance entries.
variance.inits	Optional initial covariance matrix.
est.variance	Logical: estimate the covariance matrix? (default FALSE).
A	Optional design array for item intercepts.
B	Optional design array for item slopes.

B.fixed	Optional matrix fixing entries of the slope array B.
Q	Optional loading (Q-) matrix mapping items to dimensions.
est.slopegroups	Optional vector grouping items that share a common estimated slope.
E	Optional design array used with est.slopegroups / GPCM.design.
gamma.init	Optional initial values for the estimated slope parameters.
pweights	Optional vector of person (case) weights.
userfct.variance	Optional user function constraining the covariance matrix.
variance.Npars	Number of estimated parameters in a user-constrained covariance.
item.elim	Logical: eliminate items with empty response categories?
verbose	Logical: print estimation progress?
method	Estimation engine. "grid" is the standard full-quadrature path. "streaming" routes to a dimension-factorized streaming engine for large between-item (simple-structure) multidimensional 2PL/GPCM models; it supports latent regression (Y), multiple groups (group, with per-group covariances), and case weights (pweights), but not the advanced item-design arguments (A, B, xsi.fixed, etc.), which force "grid". "auto" (default) predicts which engine is faster from (N, items, dims, nodes, unique covariate patterns) and routes accordingly (falling back to "grid" for unidimensional/small/unsupported models); the chosen engine and reasoning are returned in the routing component of the result. With case weights the result also reports the Kish effective sample size in n_eff.
control	List of control arguments (e.g. maxiter, conv, nodes, Msteps). Performance-related elements: n_threads (threads for the parallel E-step and slope M-step; default and maximum = 2), fast (logical, default FALSE; approximate mode pruning negligible-weight quadrature nodes), fast_threshold (pruning cutoff, default 1e-6), and acceleration ("none" default, or "squarem"/"Ramsay"/"Yu" to accelerate the grid EM). The streaming engine additionally accepts reg for opt-in statistical regularization (slope_penalty, slope_max, sigma_shrink, and sigma_shrink_pooled for shrinking per-group covariances toward the pooled estimate; default off), and group_structure ("full" = independent per-group covariances, default; "mean" = shared covariance with free group means). When fast=TRUE the streaming engine prunes quadrature nodes by a calibrated mass_budget (default 1e-3): after a few full-grid burn-in iterations it drops the lowest-importance nodes holding at most that fraction of the combined prior-and-posterior mass, retained per (covariate, group) pattern and unioned so extreme responders and small groups are protected. mass_budget is a pruning budget, <i>not</i> a guaranteed error. The real accuracy is <i>measured</i>: with verify="stratified" (default) a stratified subsample (groups, high weights, extreme responders, high-missingness; verify_n, verify_seed) is run through both the full and pruned grids at the final parameters, and the per-metric error quantiles (deviance / EAP / posterior moments / one-M-step parameter change, against tol_deviance/tol_eap/tol_moment/tol_pa are returned in accuracy_report with a met verdict. refine=TRUE refits with a tighter budget until the tolerances are met.

Value

An object of class `irtc` (see [irtc.mml](#)) with, in addition, the estimated slope array `B` and the slope parameter vector. `ic` is a one-row data frame from either engine. The person table carries the caller's pid, the case weights normalised to mean 1, the EAP estimates and their posterior standard errors (`SD.EAP.Dim1, ...`). Streaming-engine fits also carry a routing component (chosen engine and reasoning) and, when `fast=TRUE` with verification on, an `accuracy_report`: a measured, stratified approximation check (per-metric error quantiles plus a met verdict), *not* a strict mathematical error bound.

Missing responses

Both engines treat NA as missing at random: the item contributes a likelihood factor of 1 for that person and no expected counts, so no case has to be dropped. Persons with no responses at all are kept and sit at the prior mean.

Comparing models

The reported deviance, `loglik` and the information criteria in `ic` are marginal log-probabilities of the discretised model, on the same scale for both engines. They may therefore be compared across the "grid" and "streaming" paths, across different `ndim`, and across different `control$nodes` settings, up to the quadrature accuracy itself. Use [logLik](#), AIC, BIC or `anova` on the fitted objects.

Two cautions. First, quadrature accuracy is finite: comparisons are only as good as the node grid, so keep the nodes fixed when comparing models. Second, when an estimated latent correlation reaches the boundary the model has degenerated to a lower-dimensional one; a warning with code W427 is raised, and the criteria and EAP reliabilities of such a fit are computed on a grid that can no longer resolve the collapsed covariance.

Model choice for dichotomous items

For strictly dichotomous (0/1) data `irtmodel = "GPCM"` is mathematically equivalent to `irtmodel = "2PL"`: with two categories the generalised partial credit model reduces to the two-parameter logistic. Switching to "GPCM" therefore does not by itself change the model; only `ndim / Q` do.

Memory

The grid engine materialises an `N` by `nodes^ndim` posterior matrix. When the prediction exceeds the memory available to the session, estimation stops with an error of class `irtc_error_estimation` (code E409) naming the predicted size and pointing at `method = "streaming"`, rather than failing later inside an allocation. `method = "auto"` already routes such models to the streaming engine.

See Also

[irtc.mml](#) for fixed-slope (Rasch / PCM / RSM) models.

Examples

```
data(data.sim.rasch)
mod1 <- irtc.mml.2pl(resp = data.sim.rasch, irtmodel = "2PL")
summary(mod1)
```

```
data(data.gpcm)
mod2 <- irtc.mml.2pl(resp = data.gpcm, irtmodel = "GPCM")
summary(mod2)
```

irtc_audit_scoring *Audit a Saved Analysis for the Pre-1.1.2 Answer-Key Defect*

Description

Checks whether a saved analysis was hit by the answer-key defect fixed in IRTC 1.1.2, without re-running it.

Up to and including 1.1.1, [irtc_read](#) renumbered an item's categories to consecutive scores starting at 0 (a file coded 1..5 became 0..4) and [irtc_score](#) then matched the answer key against the renumbered values. A key written in the coding of the data file therefore selected the wrong option. Nothing warned, and the resulting 0/1 matrix looks entirely plausible, so an affected analysis cannot be spotted by inspecting its results.

The risk condition is narrow and mechanically checkable: an item was scored against a key or a rules table *and* its categories were renumbered. Both facts are recorded in the cleaning log that every fit carries, so an archived analysis can be triaged from the saved object alone.

Usage

```
irtc_audit_scoring(x)

## S3 method for class 'irtc_audit'
print(x, lang = irtc_lang(), ...)
```

Arguments

x	A fitted irtc model, an irtc_data object, the list returned by irtc_results , or a cleaning-log data frame (the <code>cleaning_log</code> element, which also travels in the exported workbooks and reports). For the print method: an irtc_audit object.
lang	Output language, "zh" or "en".
...	Ignored.

Value

An object of class `irtc_audit`: a list with `status`, `affected_items`, an items data frame (`item`, `categories_renumbered`, `original_categories`), `package_version`, `n_scored` and `n_recoded`. `status` is one of:

"not_scored" No key or rules were used: the responses were already scored, so the defect cannot apply.

"not_affected" A key was used, but no scored item had its categories renumbered, so the key applied to the intended options.

"ok_fixed" Produced by IRTC 1.1.2 or later, where the key is applied in the coding of the data file itself.

"affected" Scored against a key after renumbering, by a version known to predate the fix. Re-run the analysis.

"affected_if_before_1.1.2" The same risk condition, but the recording version is not stamped on the object (fits made before 1.1.2 carry no stamp). Establish which version produced it.

See Also

[irtc_score](#), [irtc_read](#)

Examples

```
set.seed(1)
n <- 120
raw <- data.frame(pid = seq_len(n),
  Q1 = sample(1:4, n, TRUE), Q2 = sample(1:4, n, TRUE),
  Q3 = sample(1:3, n, TRUE), Q4 = sample(1:3, n, TRUE))
f <- tempfile(fileext = ".csv")
utils::write.csv(raw, f, row.names = FALSE)

mod <- irtc(f, model = "2PL", key = c(Q1 = 2, Q2 = 3, Q3 = 1, Q4 = 2),
  id = "pid", verbose = FALSE)
irtc_audit_scoring(mod)

## an archived fit from an older version reports the items to re-run
archived <- mod
archived$usability$package_version <- "1.1.1"
irtc_audit_scoring(archived)$affected_items

unlink(f)
```

irtc_check_data

Pre-Estimation Data Diagnostics

Description

Checks a response data set before model estimation and returns both a human-readable report and a machine-readable issue table, so that survey staff and automated callers (e.g. AI agents) can verify data quality and act on concrete fixes.

Checked, among others: number of items and persons, non-numeric (raw) response columns, negative or non-integer values, items without responses or without variance, extreme missingness, sparse categories, persons without any responses and duplicated person IDs.

Usage

```
irtc_check_data(x, key = NULL, verbose = TRUE)

## S3 method for class 'irtc_check'
print(x, lang = irtc_lang(), ...)
```

Arguments

x	An irtc_data object, a data frame or a matrix. For the print method: an irtc_check object.
key	If an answer key will be used for scoring, pass it here so that raw text columns are reported as scorable instead of as errors.
verbose	Logical: print the result?
lang	Output language, "zh" or "en".
...	Ignored.

Value

An object of class irtc_check: a list with ok (logical: no error-level issues), n_persons, n_items, n_errors, n_warnings and issues, a data frame with columns code, severity ("error", "warning", "info"), scope, where, message_en, message_zh, fix_en, fix_zh.

See Also

[irtc_read](#), [irtc](#)

Examples

```
d <- data.frame(i1 = c(0, 1, 1, 0), i2 = c(1, 1, 1, 1),
  i3 = c(0, NA, 1, -9))
chk <- irtc_check_data(d, verbose = FALSE)
chk$ok
chk$issues[, c("code", "severity", "where")]
print(chk, lang = "en")
```

Description

Computes classical item statistics: item means and difficulty (share of the maximum score), corrected item-total (item-rest) correlations, Cronbach's alpha and alpha-if-item-deleted.

Usage

```
irtc_ctt(x, key = NULL, weights = NULL)

## S3 method for class 'irtc_ctt'
print(x, lang = irtc_lang(), ...)
```

Arguments

<code>x</code>	An <code>irtc_data</code> object, a data frame, a matrix, or a fitted <code>irtc</code> model (its stored response data are used). For the print method: an <code>irtc_ctt</code> object.
<code>key</code>	An optional answer key; raw responses are scored first via irtc_score .
<code>weights</code>	Optional case (sampling) weights, one per person. All statistics – item means, difficulties, item-rest correlations and Cronbach’s alpha – are then computed on the weighted sample, which is what a complex-sample survey needs so that the classical statistics describe the same population as the weighted IRT estimates. Constant weights reproduce the unweighted results exactly.
<code>lang</code>	Output language, "zh" or "en".
<code>...</code>	Ignored.

Value

An object of class `irtc_ctt`: a list with items (data frame with columns `item`, `N`, `miss_rate`, `max_score`, `N_weighted`, `M`, `pvalue`, `discr`, `alpha_if_deleted`), `alpha`, `n_persons`, `n_items` and `weighted` (whether case weights were applied).

See Also

[irtc_quality](#), [irtc_itemfit](#)

Examples

```
set.seed(1)
theta <- rnorm(200)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 6), function(b) {
  as.numeric(runif(200) < plogis(theta - b))
}))
ctt <- irtc_ctt(resp)
ctt$alpha
ctt$items

## with sampling weights
w <- runif(200, 0.5, 3)
irtc_ctt(resp, weights = w)$items$pvalue
```

Description

One call writes three separate .xlsx files (requires the optional **openxlsx** package):

1. <prefix>_item_quality.xlsx - a plain-language item quality table (score rate, difficulty label, discrimination, fit, colour-coded overall rating, reasons and advice) plus a notes sheet explaining every column, for judging item quality without statistical training.
2. <prefix>_item_parameters.xlsx - the IRT difficulty / discrimination table with a frozen column schema (schema_version, analysis_id, model, item_id, n_obs, p_value, slope_a, difficulty_b, se_b, tau_k), designed for cross-year anchor-item linking: files from different administrations can be merged directly on item_id.
3. <prefix>_person_ability.xlsx - a flat person ability table (ID, number answered, raw score, EAP, standard error, percentile, T score) whose rows stay in input order so it can be pasted next to a master sample sheet.

Usage

```
irtc_excel(mod, dir = ".", prefix = "IRT", lang = irtc_lang(),
  resp = NULL, overwrite = FALSE, verbose = TRUE)
```

```
irtc_param_table(mod, resp = NULL, weights = NULL)
```

```
irtc_person_table(mod, lang = irtc_lang(), weights = NULL)
```

Arguments

mod	A fitted irtc model object.
dir	Output directory (created if necessary).
prefix	File name prefix.
lang	Language of headers and notes, "zh" or "en". The linking schema of file 2 is language-independent.
resp	The response data; only needed when the model object does not store it (streaming engine).
weights	Optional case (sampling) weights, one per person. For irtc_param_table they weight the p_value column so it shares the basis of the weighted IRT parameters beside it; for irtc_person_table they weight the norm-referenced percentile and T_score columns, so those describe the population the weights represent rather than the achieved sample. Both default to the case weights stored on the model, and constant weights reproduce the unweighted values exactly.
overwrite	Logical: overwrite existing files? Otherwise an error of class irtc_error_export (code E501) is raised.
verbose	Logical: print a completion message?

Value

`irtc_excel` invisibly returns a named character vector with the three file paths (quality, parameters, ability). `irtc_param_table` and `irtc_person_table` return the underlying data frames and work without **openxlsx**.

For a multidimensional model `irtc_param_table` adds a dimension column naming the latent dimension each item loads on (from the Q-matrix column headers when available), and `n_loadings`, the number of dimensions the item loads on. `slope_a` and `difficulty_b` are read from that dimension. An item loading on several dimensions at once (within-item structure) has no single slope: `slope_a` is then the sum of its loadings and `n_loadings` is above 1, flagging the value as a composite.

The person-level percentile and `T_score` columns are norm-referenced: with case weights they are computed on the weighted sample, so a percentile is the share of the represented population below that person. The ability estimates and their standard errors are person-level quantities and are never weighted.

See Also

[irtc](#), [irtc_report](#), [irtc_results](#)

Examples

```
set.seed(1)
theta <- rnorm(150)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(150) < plogis(theta - b))
}))
mod <- irtc(resp, model = "1PL", verbose = FALSE)

## data frames behind the workbooks (no extra packages needed)
irtc_param_table(mod)
head(irtc_person_table(mod, lang = "en"))

if (requireNamespace("openxlsx", quietly = TRUE)) {
  out <- irtc_excel(mod, dir = tempdir(), overwrite = TRUE,
    verbose = FALSE)
  basename(out)
}
```

irtc_itemfit

Item Fit Statistics (Infit and Outfit)

Description

Residual-based item fit for a fitted model: infit and outfit mean squares with Wilson-Hilferty t statistics. Values near 1 indicate good fit; the conventional normal range is 0.7 to 1.3.

The residuals are evaluated at the *EAP* (expected a posteriori) person estimates – worth knowing when comparing the numbers against software that uses WLE estimates instead.

Usage

```
irtc_itemfit(mod, resp = NULL, weights = NULL)

## S3 method for class 'irtc_itemfit'
print(x, lang = irtc_lang(), ...)
```

Arguments

mod	A fitted irtc model object.
resp	The response data; only needed when the model object does not store it (streaming engine).
weights	Optional case (sampling) weights, one per person: each person then contributes w_i standardised residuals rather than one, so the fit statistics rest on the same weighted population as the item parameters they judge. Defaults to the case weights stored on the model; constant weights reproduce the unweighted statistics exactly.
x	An irtc_itemfit object.
lang	Output language, "zh" or "en".
...	Ignored.

Value

A data frame of class irtc_itemfit with columns item, N, outfit, outfit_t, infit, infit_t.

See Also

[irtc_quality](#), [irtc_ctt](#)

Examples

```
set.seed(1)
theta <- rnorm(200)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(200) < plogis(theta - b))
}))
mod <- irtc.mml(resp = resp, verbose = FALSE)
irtc_itemfit(mod)
```

 irtc_lang

Output Language of the IRTC Usability Layer

Description

Returns the current output language, "zh" (Chinese, the default) or "en" (English). Set it globally with `options(irtc.lang = "en")`. The language affects user-facing text (messages, printed summaries, Excel headers, reports); machine-readable schemas ([irtc_results](#), the linking table of [irtc_excel](#)) never depend on it.

Usage

```
irtc_lang()
```

Value

"zh" or "en".

Examples

```
old <- options(irtc.lang = "en")
irtc_lang()
options(old)
```

irtc_quality	<i>Plain-Language Item Quality Ratings</i>
--------------	--

Description

Combines classical difficulty and discrimination ([irtc_ctt](#)) with IRT item fit ([irtc_itemfit](#)) into a four-level quality rating per item - "good", "acceptable", "review", "revise" - with bilingual reasons and advice that non-specialists can act on. Negative discrimination (usually a wrong answer key) always leads to "revise".

Usage

```
irtc_quality(mod, resp = NULL, thresholds = NULL, weights = NULL)
```

```
irtc_quality_thresholds()
```

```
## S3 method for class 'irtc_quality'
print(x, lang = irtc_lang(), ...)
```

Arguments

mod	A fitted irtc model object.
resp	The response data; only needed when the model object does not store it (streaming engine).
weights	Optional case (sampling) weights, one per person, passed on to irtc_ctt and irtc_itemfit so that both halves of the rating rest on the same population. Defaults to the case weights stored on the model.
thresholds	An optional named list overriding entries of <code>irtc_quality_thresholds()</code> : <code>p_hard</code> (default 0.10), <code>p_easy</code> (0.95), <code>discr_poor</code> (0.15), <code>discr_weak</code> (0.25), <code>fit_mild</code> (c(0.70, 1.30)), <code>fit_severe</code> (c(0.50, 1.50)).
x	An irtc_quality object.
lang	Output language, "zh" or "en".
...	Ignored.

Value

A data frame of class `irtc_quality` with columns `item`, `N`, `pvalue`, `discr`, `outfit`, `infit`, `rating`, `reasons_en`, `reasons_zh`, `advice_en`, `advice_zh`, and attributes `alpha` (Cronbach's alpha) and `thresholds`. `irtc_quality_thresholds()` returns the default threshold list.

See Also

[irtc](#), [irtc_excel](#)

Examples

```
set.seed(1)
theta <- rnorm(300)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 6), function(b) {
  as.numeric(runif(300) < plogis(theta - b))
}))
resp$noise <- rbinom(300, 1, 0.5) # a broken item
mod <- irtc.mml(resp = resp, verbose = FALSE)
qual <- irtc_quality(mod)
qual[, c("item", "pvalue", "discr", "rating")]
print(qual, lang = "en")
```

 irtc_read

Read and Clean Item Response Data from Many Formats

Description

Unified data import with automatic cleaning. Supports Excel (`.xlsx`, `.xls`; requires **readxl**), delimited text (`.csv`, `.tsv`, `.txt`, `.dat`; delimiter and UTF-8/GBK encoding are detected automatically), SPSS/Stata/SAS files (`.sav`, `.por`, `.dta`, `.sas7bdat`, `.xpt`; requires **haven**) and in-memory data frames, matrices and tibbles.

Every cleaning action is recorded in a bilingual log attached to the returned object, so results remain traceable.

Usage

```
irtc_read(x, sheet = 1, id = NULL, weights = NULL,
  missing_codes = c(-9, -99, 99, 999),
  na_strings = c("", "NA", "N/A", "n/a", ".", "*"),
  "\u7f3a\u5931", "\u65e0", "\u7a7a"),
  guess_id = TRUE, guess_weights = TRUE, clean = TRUE, recode = TRUE,
  verbose = TRUE)

## S3 method for class 'irtc_data'
print(x, lang = irtc_lang(), ...)
```

Arguments

<code>x</code>	A file path or a data frame / matrix. For the print method: an <code>irtc_data</code> object.
<code>sheet</code>	Excel sheet to read (number or name).
<code>id</code>	Person identifier column(s), by name or index. The first is kept as the person ID; the rest are set aside. If <code>NULL</code> and <code>guess_id</code> is <code>TRUE</code> , common ID column names (<code>id</code> , <code>pid</code> , student number, name, etc., in English and Chinese) are detected automatically.
<code>missing_codes</code>	Numeric codes recoded to NA. Negative codes are always treated as missing; a positive code is only recoded in a column when it lies clearly above the observed response range, which protects e.g. a legitimate score of 99 on a 0-100 scale.
<code>na_strings</code>	Strings treated as missing in text columns.
<code>weights</code>	Sampling weight column, by name or index. If <code>NULL</code> and <code>guess_weights</code> is <code>TRUE</code> , common weight column names (<code>weight</code> , <code>wt</code> , <code>pweight</code> , " <code>\u6743\u91cd</code> ", " <code>\u6837\u672c\u6743\u91cd</code> ", etc., in English and Chinese) are detected automatically. The column is set aside as <code>\$weights</code> and later used as <code>pweights</code> by irtc . Weights must be positive numbers; missing weights are set to 1 with a warning.
<code>guess_id</code>	Logical: detect the person ID column automatically?
<code>guess_weights</code>	Logical: detect the weight column automatically?
<code>clean</code>	Logical: apply the cleaning steps (drop empty rows/columns, convert numeric-like text, recode missing codes)?
<code>recode</code>	Logical: recode integer response categories to consecutive scores starting at 0 (e.g. a 1-5 Likert item becomes 0-4)?
<code>verbose</code>	Logical: print the object (including the cleaning log)?
<code>lang</code>	Output language, " <code>zh</code> " or " <code>en</code> ".
<code>...</code>	Ignored.

Value

An object of class `irtc_data`: a list with components `resp` (cleaned response data frame), `pid` (person IDs or `NULL`), `weights` (sampling weights or `NULL`), `dropped` (set-aside non-response columns), `log` (cleaning log data frame with columns `step`, `code`, `message_en`, `message_zh`) and `source`.

Category recoding

With `recode = TRUE` (the default) integer response columns are renumbered to consecutive scores starting at 0, which is what the estimation needs. The original categories are kept in the `recode_map` component so that an answer key written in the file's own coding still applies; see [irtc_score](#).

See Also

[irtc](#), [irtc_score](#), [irtc_check_data](#)

Examples

```
path <- tempfile(fileext = ".csv")
writeLines(c("id,i1,i2,i3", "s1,0,1,99", "s2,1,-9,1", "s3,1,0,0"), path)
d <- irtc_read(path, verbose = FALSE)
d$resp
d$pid
print(d, lang = "en")
unlink(path)
```

 irtc_read_q

Read a Q Matrix from a File and Align It with the Response Data

Description

`irtc_read_q` reads a Q (item-by-dimension) matrix from any file format supported by `irtc_read` (Excel, delimited text, SPSS/Stata/SAS) or from a data frame / matrix. The dimension column headers become the dimension names used in all downstream person-level output (ability and standard-error headers). An optional partial-credit column declares which items are scored in multiple levels, and optionally their maximum scores.

`irtc_align_q` compares the Q-matrix items with the response-data items. By default mismatches raise warnings (W420, W421) and the analysis continues with the shared items; `on_mismatch = "error"` turns any mismatch into an error (E422).

Usage

```
irtc_read_q(x, sheet = 1)

irtc_align_q(data, q, on_mismatch = c("warn", "error"))

## S3 method for class 'irtc_qmatrix'
print(x, lang = irtc_lang(), ...)
```

Arguments

<code>x</code>	A file path, data frame, or numeric matrix with item rownames. For the print method: an <code>irtc_qmatrix</code> object.
<code>sheet</code>	Excel sheet to read (number or name).
<code>data</code>	An <code>irtc_data</code> object (from <code>irtc_read</code>), data frame or matrix of item responses.
<code>q</code>	An <code>irtc_qmatrix</code> object, or anything accepted by <code>irtc_read_q</code> .
<code>on_mismatch</code>	"warn" (default): report item mismatches as warnings and continue with the shared items. "error": stop on any mismatch.
<code>lang</code>	Output language, "zh" or "en".
<code>...</code>	Ignored.

Details

Expected table layout for `irtc_read_q`:

- An item-ID column named `item`, `item_id`, `"\u9898\u76ee"`, `"\u9898\u53f7"` (etc.); if no such name is found, the first non-numeric column is used.
- One or more numeric dimension columns; each header is a dimension name (e.g. `"\u4ee3\u6570"`, algebra). Values are usually 0/1 loadings. Empty cells count as 0. Every item must load on at least one dimension.
- Optionally one partial-credit column named `partial`, `max_score`, `"\u5206\u90e8\u8ba1\u5206"`, `"\u6ee1\u5206"` (etc.). Values may be 0/1, TRUE/FALSE, yes/no, `"\u662f"/"\u5426"`, or integer maximum scores (a value ≥ 2 declares a partial-credit item and records its maximum score).

Value

`irtc_read_q` returns an object of class `irtc_qmatrix`: a list with components `Q` (numeric item-by-dimension matrix with dimnames), `partial` (named logical vector), `max_score` (named integer vector, NA when not declared), `log` and `source`.

`irtc_align_q` returns a list with components `data` (the input data restricted to the shared items), `q` (the aligned `irtc_qmatrix`, rows in data column order), `common`, `q_only` and `data_only` (character vectors of item names).

See Also

[irtc](#), [irtc_read](#)

Examples

```
qdf <- data.frame(item = c("I1", "I2", "I3"),
  algebra = c(1, 1, 0), geometry = c(0, 0, 1),
  partial = c(0, 1, 0))
qm <- irtc_read_q(qdf)
print(qm, lang = "en")

resp <- data.frame(I1 = c(0, 1), I2 = c(2, 0), I4 = c(1, 1))
al <- suppressWarnings(irtc_align_q(resp, qm))
al$q_only
al$data_only
```

Description

Writes an analysis report with figures (Wright map, ability distribution, item quality summary, item characteristic curves) and tables. Three audience layouts are available: "decision" (a short executive summary with recommendations), "survey" (a plain-language full report) and "stat" (a complete technical report including item parameters, item fit and model information). Every layout includes a *Model diagnostics* section (convergence, information criteria with interpretation, EAP reliability band, item-fit reading) and a *Data processing transparency* section (weights summary, Q-matrix item alignment, unobserved-category collapses, dropped items, scoring summary, and - for "stat"/"survey" - the full cleaning log), detailed for statisticians and condensed for other audiences.

HTML reports are fully self-contained single files (images embedded); Word reports require the optional **officer** package.

Usage

```
irtc_report(mod, file, format = NULL,
  audience = c("survey", "decision", "stat"), lang = irtc_lang(),
  resp = NULL, title = NULL, overwrite = FALSE, verbose = TRUE)
```

Arguments

mod	A fitted irtc model object.
file	Output file path ending in .docx or .html.
format	"docx" or "html"; inferred from file when NULL.
audience	Report layout; see Description.
lang	Output language, "zh" or "en".
resp	The response data; only needed when the model object does not store it (streaming engine).
title	Report title; a bilingual default is used when NULL.
overwrite	Logical: overwrite an existing file?
verbose	Logical: print a completion message?

Value

Invisibly, the output file path.

See Also

[irtc](#), [plain_summary](#), [irtc_excel](#), [plot.irtc](#)

Examples

```
set.seed(1)
theta <- rnorm(150)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(150) < plogis(theta - b))
})))
```

```
mod <- irtc(resp, model = "1PL", verbose = FALSE)

file <- tempfile(fileext = ".html")
irtc_report(mod, file, audience = "decision", lang = "en",
  verbose = FALSE)
file.exists(file)
unlink(file)
```

irtc_results

Machine-Readable Results with a Stable Schema

Description

Extracts fitted-model results as tidy data frames with a stable, documented schema aimed at automated callers (pipelines, AI agents). Column names are fixed English snake_case identifiers and never depend on the language option. `irtc_json` exports the same content as JSON (requires the optional **jsonlite** package).

Schema v1.2 components: `model_info` (one row: schema and package version, model, sizes, deviance, AIC/BIC, EAP reliability, iterations, and `dimension_names` when a Q matrix was supplied), `items` (per item: parameters, classical statistics, fit, quality rating and bilingual reasons/advice, plus a status column, for polytomous items semantic step difficulties and rare-category annotations, and for multidimensional models `dimension` and `n_loadings`), `persons` (per person: ID, number answered, raw score, EAP, SE, percentile, T score; ability/SE headers use the Q dimension names when available), `cleaning_log` and `check_issues` (present when the model was fitted via `irtc`). Schema 1.2 is additive over 1.1: existing column names and meanings are unchanged.

Sampling weights given to `irtc` are honoured throughout: the item statistics share one population with the IRT parameters, and the norm-referenced person columns `percentile` and `t_score` are computed on the weighted sample, so a percentile is the share of the represented population below that person. Pass weights to override the weights stored on the model. The ability estimates and their standard errors are person-level quantities and are never weighted. Items declared in the Q matrix but not estimated keep a row with status `"dropped_no_response"` or `"declared_not_estimated"`. See the package file `inst/llms.txt` for the full schema reference.

Usage

```
irtc_results(mod, resp = NULL, weights = NULL)

irtc_json(mod, file = NULL, resp = NULL, pretty = TRUE)

## S3 method for class 'irtc_results'
print(x, lang = irtc_lang(), ...)
```

Arguments

<code>mod</code>	A fitted <code>irtc</code> model object; <code>irtc_json</code> also accepts an <code>irtc_results</code> object.
<code>resp</code>	The response data; only needed when the model object does not store it (streaming engine).

weights	Optional case (sampling) weights, one per person, used for the norm-referenced percentile and t_score columns and for the item statistics. Defaults to the case weights stored on the model; constant weights reproduce the unweighted values exactly.
file	Optional path; when given, the JSON is written there (UTF-8) and the path is returned invisibly.
pretty	Logical: pretty-print the JSON?
x	An irtc_results object.
lang	Output language, "zh" or "en".
...	Ignored.

Value

irtc_results returns an object of class irtc_results, a list of data frames as described above.
 irtc_json returns a JSON string, or invisibly the file path when file is given.

See Also

[irtc](#), [irtc_excel](#)

Examples

```
set.seed(1)
theta <- rnorm(150)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(150) < plogis(theta - b))
})))
mod <- irtc(resp, model = "1PL", verbose = FALSE)

res <- irtc_results(mod)
res$model_info
head(res$items)

if (requireNamespace("jsonlite", quietly = TRUE)) {
  json <- irtc_json(mod)
  substr(json, 1, 80)
}
```

 irtc_score

Score Raw Responses with an Answer Key or Scoring Rules

Description

Converts raw multiple-choice responses (e.g. "A", "B", "C", "D") to 0/1 scores using an answer key, or maps responses to partial-credit scores through a rules table. Responses are normalised before matching: whitespace is trimmed, case is ignored and full-width characters are converted to half-width.

Both key and rules may also be file paths in any format supported by [irtc_read](#). A key file needs an item column (item, "\u9898\u76ee", ...) and an answer column (answer, "\u7b54\u6848", ...); an optional partial-answer column (partial_answer, "\u90e8\u5206\u6b63\u786e\u7b54\u6848", ...) lists answers worth partial credit, several separated by |, ;, , or "\u3001". Items with partial answers are scored full = 2, partial = 1, other = 0; items without stay 0/1. For more than three score levels supply an explicit rules table instead.

Usage

```
irtc_score(resp, key = NULL, rules = NULL, na_as_wrong = FALSE,
           sheet = 1)
```

Arguments

resp	An irtc_data object (from irtc_read), a data frame or a matrix of raw responses.
key	A named vector mapping item names to correct answers, e.g. c(Q1 = "A", Q2 = "C"), or a path to a key file (see Description). An unnamed vector is matched to the item columns by position. Items not named in the key are left unchanged.
rules	A data frame with columns item, response, score defining partial-credit scoring, or a path to such a table (Chinese column aliases "\u9898\u76ee", "\u4f5c\u7b54" and "\u5206\u503c" are accepted). Observed responses without a rule are set to NA with a warning of class irtc_warning_scoring.
na_as_wrong	Logical: score missing responses as 0 instead of NA? (Key scoring only.)
sheet	Excel sheet of the key / rules file (number or name).

Value

The same type as the input: an irtc_data object with scored resp and an extended log, or a data frame with a "scoring_log" attribute.

Answer coding

Write the key in the coding of your own data file. [irtc_read](#) may renumber an item's categories to consecutive scores starting at 0 (a file coded 1..5 becomes 0..4, logged as I124); it records the original categories, and irtc_score translates the key, the partial answers and a rules table through that record. An answer that never occurs among an item's responses is reported with a warning (code W205) rather than scoring every respondent wrong – which catches a mistyped key as well as an option nobody chose.

See Also

[irtc_read](#), [irtc](#)

Examples

```
resp <- data.frame(Q1 = c("A", "b", "C"), Q2 = c("c", "C", "D"),
                  stringsAsFactors = FALSE)
irtc_score(resp, key = c(Q1 = "A", Q2 = "C"))
```

```
rules <- data.frame(item = "Q1", response = c("A", "B", "C"),
  score = c(2, 1, 0))
irtc_score(resp["Q1"], rules = rules)
```

plain_summary

Plain-Language Layered Summary of a Fitted Model

Description

Prints a layered, plain-language summary of a fitted item response model: conclusion first, then analysis overview, item quality, ability distribution and suggested next steps. Intended for survey staff and decision makers; statisticians can use `summary(mod)` for the full technical output.

Usage

```
plain_summary(mod, lang = irtc_lang())
```

Arguments

<code>mod</code>	A fitted irtc model object (from irtc , irtc.mml or irtc.mml.2pl).
<code>lang</code>	Output language, "zh" or "en".

Value

Invisibly, the list of summary sections (each with `title` and `body`); the same content that [irtc_report](#) uses.

See Also

[irtc](#), [irtc_report](#)

Examples

```
set.seed(1)
theta <- rnorm(150)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(150) < plogis(theta - b))
}))
mod <- irtc(resp, model = "1PL", verbose = FALSE)
plain_summary(mod, lang = "en")
```

plot.irtc

*Diagnostic Plots for Fitted Item Response Models***Description**

Base-graphics plots for a fitted model: "wright" (item difficulty vs person ability map), "ability" (person ability histogram, one panel per dimension), "quality" (item quality rating summary) and "icc" (item characteristic curves; unidimensional models only). The same figures are embedded in [irtc_report](#) reports.

Usage

```
## S3 method for class 'irtc'
plot(x, type = c("wright", "ability", "quality", "icc"),
     lang = irtc_lang(), items = NULL, resp = NULL, ...)
```

Arguments

x	A fitted irtc model object.
type	Plot type; see Description.
lang	Label language, "zh" or "en". Chinese labels require a CJK-capable graphics font.
items	For type = "icc": item indices or names to plot (default: the first 12 items).
resp	The response data; only needed when the model object does not store it (streaming engine).
...	Ignored.

Value

Invisibly, x.

See Also

[irtc](#), [irtc_report](#)

Examples

```
set.seed(1)
theta <- rnorm(150)
resp <- as.data.frame(sapply(seq(-1, 1, length.out = 5), function(b) {
  as.numeric(runif(150) < plogis(theta - b))
}))
mod <- irtc(resp, model = "1PL", verbose = FALSE)
plot(mod, type = "wright", lang = "en")
plot(mod, type = "icc", lang = "en", items = 1:2)
```

Index

- * **datasets**
 - data.gpcm, [3](#)
 - data.mc, [4](#)
 - data.sim.rasch, [4](#)
- * **models**
 - irtc.mml, [7](#)
 - irtc.mml.2pl, [10](#)
- * **package**
 - IRTC-package, [2](#)

anova.irtc(irtc.mml), [7](#)

data.gpcm, [3](#)

data.mc, [4](#)

data.sim.rasch, [4](#)

IRTC (IRTC-package), [2](#)

irtc, [3](#), [5](#), [15](#), [18](#), [21](#), [22](#), [24–30](#)

IRTC-package, [2](#)

irtc.mml, [3](#), [5](#), [6](#), [7](#), [12](#), [29](#)

irtc.mml.2pl, [3](#), [5](#), [6](#), [9](#), [10](#), [29](#)

irtc_align_q, [5](#)

irtc_align_q(irtc_read_q), [23](#)

irtc_audit_scoring, [13](#)

irtc_check_data, [3](#), [6](#), [14](#), [22](#)

irtc_ctt, [3](#), [15](#), [19](#), [20](#)

irtc_excel, [3](#), [6](#), [17](#), [19](#), [21](#), [25](#), [27](#)

irtc_itemfit, [3](#), [16](#), [18](#), [20](#)

irtc_json, [3](#)

irtc_json(irtc_results), [26](#)

irtc_lang, [3](#), [19](#)

irtc_param_table(irtc_excel), [17](#)

irtc_person_table(irtc_excel), [17](#)

irtc_quality, [3](#), [16](#), [19](#), [20](#)

irtc_quality_thresholds(irtc_quality), [20](#)

irtc_read, [3](#), [5](#), [6](#), [13–15](#), [21](#), [23](#), [24](#), [28](#)

irtc_read_q, [5](#), [23](#)

irtc_report, [3](#), [6](#), [18](#), [24](#), [29](#), [30](#)

irtc_results, [3](#), [6](#), [13](#), [18](#), [19](#), [26](#)

irtc_score, [3](#), [5](#), [13](#), [14](#), [16](#), [22](#), [27](#)

logLik, [12](#)

logLik.irtc(irtc.mml), [7](#)

plain_summary, [3](#), [6](#), [25](#), [29](#)

plot, [3](#)

plot.irtc, [25](#), [30](#)

print.irtc(irtc.mml), [7](#)

print.irtc_audit(irtc_audit_scoring), [13](#)

print.irtc_check(irtc_check_data), [14](#)

print.irtc_ctt(irtc_ctt), [15](#)

print.irtc_data(irtc_read), [21](#)

print.irtc_itemfit(irtc_itemfit), [18](#)

print.irtc_qmatrix(irtc_read_q), [23](#)

print.irtc_quality(irtc_quality), [20](#)

print.irtc_results(irtc_results), [26](#)

summary.irtc(irtc.mml), [7](#)