

Package: aiDIF (via r-universe)

September 20, 2026

Type Package

Title Differential Item Functioning for AI-Scored Assessments

Version 0.2.0

Description Detects and quantifies differential item functioning (DIF) in AI-scored educational and psychological assessments. Provides a fully self-contained robust DIF engine (M-estimation via iteratively re-weighted least squares with the bi-square loss) alongside the Differential AI Scoring Bias (DASB) test, which detects item-level scoring shifts that differ across subgroups when comparing human and AI scoring conditions. Supports independent and paired scoring designs, robust linking of the cross-condition contrast, multiplicity control, conversion of fitted 'mirt' models to package inputs, simulation utilities, anchor weight diagnostics, and an AI-effect classification framework. Methods follow Halpin (2024) <[doi:10.1007/s11336-024-09957-6](https://doi.org/10.1007/s11336-024-09957-6)>.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Matrix, stats, graphics, utils

Suggests mirt, testthat (>= 3.1.5), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

URL <https://github.com/causalfragility-lab/aiDIF>

BugReports <https://github.com/causalfragility-lab/aiDIF/issues>

Config/roxygen2/version 8.0.0

Repository <https://subirhait.r-universe.dev>

Date/Publication 2026-09-04 16:13:07 UTC

RemoteUrl <https://github.com/subirhait/aidif>

RemoteRef HEAD

RemoteSha 6668f5f135463a547248481e859474bc83d20cfc

Contents

ai_effect_summary	2
anchor_weights	3
as_aidif	4
estimate_robust_scale	5
fit_aidif	6
make_aidif_eg	7
plot.aidif	8
print.aidif	9
print.summary.aidif	10
read_ai_scored	10
scoring_bias_test	11
simulate_aidif_data	13
summary.aidif	14
Index	16

ai_effect_summary	<i>Summarise the effect of AI scoring on DIF flagging</i>
-------------------	---

Description

Compares the DIF flagging patterns from the human and AI scoring conditions and labels each item "stable_clean" (not flagged in either), "stable_dif" (flagged in both, same direction), "introduced" (flagged only under AI), "masked" (flagged only under human), or "new_direction" (flagged in both, opposite directions).

Usage

```
ai_effect_summary(dif_human, dif_ai = NULL, alpha = 0.05)
```

Arguments

- dif_human A DIF table from [fit_aidif](#), or an aidif object.
- dif_ai A DIF table from [fit_aidif](#), or NULL when dif_human is a complete aidif object.
- alpha Flagging threshold. Default 0.05.

Value

A data.frame with columns human_delta, ai_delta, human_flag, ai_flag, and status. Values are unrounded.

Interpretation

These labels are **descriptive only**. They are derived from a pair of significance flags, and a difference in significance is not itself a significant difference (Gelman and Stern, 2006). An item can move from "not flagged" to "flagged" through sampling variation alone. Use [scoring_bias_test](#) to test whether the group contrast actually changed; use these labels to describe the pattern afterwards.

References

Gelman, A. and Stern, H. (2006). The difference between "significant" and "not significant" is not itself statistically significant. *The American Statistician*, 60(4), 328–331.

See Also

[scoring_bias_test](#), [fit_aidif](#)

Examples

```
eg <- make_aidif_eg()
mod <- fit_aidif(eg$human, eg$ai)
ai_effect_summary(mod)
```

anchor_weights	<i>Anchor item weights from the robust AI-DIF procedure</i>
----------------	---

Description

Returns the bi-square weights assigned to each item under each scoring condition. Items with weight near zero are effectively excluded from the robust scaling estimate, indicating likely DIF contamination.

Usage

```
anchor_weights(object)
```

Arguments

object An aidif object from [fit_aidif](#).

Value

A data.frame with column human_weight and, when AI data were supplied, ai_weight. Values are unrounded.

Examples

```
eg <- make_aidif_eg()
mod <- fit_aidif(eg$human, eg$ai)
anchor_weights(mod)
```

as_aidif

Convert a fitted IRT model into aiDIF input

Description

Extracts group-specific item parameter estimates and their asymptotic covariance matrices from a fitted multiple-group IRT model and returns the mle list structure that [fit_aidif](#) and [scoring_bias_test](#) expect.

Usage

```
as_aidif(object, groups = NULL, ...)

## S3 method for class 'MultipleGroupClass'
as_aidif(object, groups = NULL, ...)
```

Arguments

object	A fitted model. Currently only multiple-group <code>MultipleGroupClass</code> objects from mirt are supported, fitted with <code>SE = TRUE</code> so that a covariance matrix is available.
groups	Optional character vector of length two giving the reference and focal group names, in that order. Defaults to the first two groups in the fitted object, in the order mirt stores them. Supply this explicitly: silently taking the first two groups is the kind of default that produces a valid-looking but reversed contrast.
...	Passed to methods.

Details

This removes the manual extraction step that 0.1.0 required, in which users had to slice a joint covariance matrix and reassemble parameter data frames by hand.

Value

A list with components `par.names`, `est`, and `var.cov`, suitable as `human_mle` or `ai_mle`.

See Also

[read_ai_scored](#), [fit_aidif](#)

Examples

```
if (requireNamespace("mirt", quietly = TRUE)) {
  grp <- rep(c("reference", "focal"), each = 196)
  mod <- mirt::multipleGroup(mirt::Science, 1, group = grp, SE = TRUE,
                             verbose = FALSE)

  dat <- as_aidif(mod, groups = c("reference", "focal"))
  str(dat$est$group.1)

  # In practice the two arguments to fit_aidif() come from two separate
  # calibrations, one per scoring condition:
  # fit <- fit_aidif(as_aidif(human_mod), as_aidif(ai_mod))
}
```

estimate_robust_scale *Robust DIF scale estimation via IRLS*

Description

Estimates a robust location parameter for the vector of IRT scaling functions using iteratively re-weighted least squares (IRLS) with the bi-square loss. This is the core estimation engine of **aiDIF**.

Usage

```
estimate_robust_scale(
  mle,
  alpha = 0.05,
  scale_by = "pooled",
  tol = 1e-07,
  maxit = 100L
)
```

Arguments

mle	A validated mle list.
alpha	Significance level controlling the bi-square tuning parameter $k = z_{1-\alpha/2}$. Default 0.05.
scale_by	Scaling denominator; passed to compute_scaling_fn . Default "pooled".
tol	Convergence tolerance. Default 1e-7.
maxit	Maximum IRLS iterations. Default 100.

Value

A list of class `rdif` with elements:

`est` Estimated robust scale parameter.

`weights` Bi-square item weights.

`rho_value` Value of objective at solution.

`n_iter` Number of iterations used.

`k` Tuning parameter used.

`y` Raw scaling function values.

`vcov_est` Covariance matrix of `y` at solution.

`dif_test` Wald item-level DIF test (`data.frame`).

`dtf_test` Wald test of differential test functioning.

Examples

```
dat <- simulate_aidif_data(n_items = 5, seed = 1)
fit <- estimate_robust_scale(dat$human)
print(fit$est)
```

fit_aidif

Fit the AI-DIF model

Description

The primary estimation function of **aiDIF**. Runs the robust DIF procedure under both human and AI scoring using the built-in IRLS engine ([estimate_robust_scale](#)), then tests for Differential AI Scoring Bias (DASB).

Usage

```
fit_aidif(
  human_mle,
  ai_mle = NULL,
  alpha = 0.05,
  scale_by = c("pooled", "ref", "focal"),
  design = c("independent", "paired"),
  cross_cov = NULL,
  metric = c("common", "linked"),
  adjust = "none",
  tol = 1e-07,
  maxit = 100L
)
```

Arguments

human_mle	A validated mle list for the human scoring condition, or an aidif_data object from read_ai_scored (in which case ai_mle is taken from it and must not be supplied).
ai_mle	A validated mle list for the AI scoring condition, or NULL to fit the human condition only.
alpha	Significance level. Default 0.05.
scale_by	Denominator for standardising intercept differences: "pooled" (default), "ref", or "focal".
design	Passed to scoring_bias_test : either "independent" (default) or "paired".
cross_cov	Cross-condition covariance list, required when design = "paired". See scoring_bias_test .
metric	Passed to scoring_bias_test : "common" (default) or "linked".
adjust	Multiplicity adjustment across items for the DASB tests. Any p.adjust method, or "none" (default).
tol	IRLS convergence tolerance. Default 1e-7.
maxit	Maximum IRLS iterations. Default 100.

Value

An object of class "aidif", a list with components human_fit and ai_fit (objects of class rdif), dif_human and dif_ai (Wald DIF tables), scoring_bias (a dasb table), ai_effect (classification table), data, and the settings used.

See Also

[estimate_robust_scale](#), [scoring_bias_test](#), [as_aidif](#)

Examples

```
dat <- simulate_aidif_data(n_items = 6, seed = 1)
mod <- fit_aidif(dat$human, dat$ai)
print(mod)

# Robustly centred DASB with Benjamini-Hochberg adjustment
mod2 <- fit_aidif(dat$human, dat$ai, metric = "linked", adjust = "BH")
```

make_aidif_eg

Built-in example dataset for aiDIF

Description

Constructs and returns the built-in example dataset: paired human and AI item parameter estimates for 6 items in two groups, with known DIF and DASB planted at specific items.

Usage

```
make_aidif_eg()
```

Details

The data-generating model includes:

- **Item 1:** DIF under human scoring (intercept +0.5 in focal group).
- **Item 3:** Differential AI Scoring Bias (DASB) — AI scoring adds +0.4 to the focal-group intercept only.
- **Impact:** 0.5 SD (focal group higher on latent trait).
- **AI drift:** uniform +0.1 calibration offset on all items in both groups.

Value

A list with elements `human` and `ai`, each a validated mle list (see [simulate_aidif_data](#) for format details).

See Also

[simulate_aidif_data](#), [fit_aidif](#)

Examples

```
eg <- make_aidif_eg()
mod <- fit_aidif(eg$human, eg$ai)
summary(mod)
```

plot.aidif

S3 plot method for class "aidif"

Description

S3 plot method for class "aidif"

Usage

```
## S3 method for class 'aidif'
plot(x, type = "dif_forest", ...)
```

Arguments

x	An object of class "aidif".
type	Character. One of: "dif_forest" Forest plot of DIF estimates with confidence intervals for both scoring conditions (default). "dasb" Bar chart of DASB estimates. "weights" Dot plot of bi-square anchor weights. "rho" Bi-square objective for human scoring.
...	Additional graphical parameters.

Value

x, invisibly.

print.aidif	<i>S3 print method for class "aidif"</i>
-------------	--

Description

Prints a compact summary of the estimated robust scaling parameters and, when available, the number of items flagged for DIF and DASB.

Usage

```
## S3 method for class 'aidif'
print(x, digits = 4, ...)
```

Arguments

x	An object of class "aidif".
digits	Number of digits to display. Default 4.
...	Further arguments (currently ignored).

Value

x, invisibly.

```
print.summary.aidif
```

Print method for "summary.aidif" objects

Description

Print method for "summary.aidif" objects

Usage

```
## S3 method for class 'summary.aidif'
print(x, digits = 4, ...)
```

Arguments

<code>x</code>	An object of class "summary.aidif".
<code>digits</code>	Number of digits to display. Default 4.
<code>...</code>	Further arguments (currently ignored).

Value

`x`, invisibly.

```
read_ai_scored
```

Validate and bundle paired human/AI parameter estimates

Description

Takes two mle lists (one per scoring condition) and returns a validated `aidif_data` object for use in `fit_aidif`.

Usage

```
read_ai_scored(human_mle, ai_mle)
```

Arguments

<code>human_mle</code>	An mle list for human-scored data. Must contain <code>est</code> (a named list group.1, group.2 of data.frames with columns <code>a1</code> , <code>d1</code>) and <code>var.cov</code> (matching list of covariance matrices).
<code>ai_mle</code>	An mle list for AI-scored data in the same format.

Value

A list of class "aidif_data" with elements `human` and `ai`.

See Also

[fit_aidif](#), [make_aidif_eg](#), [simulate_aidif_data](#)

scoring_bias_test	<i>Differential AI Scoring Bias (DASB) test</i>
-------------------	---

Description

For each item, computes the change in item intercept from human to AI scoring within each group, then tests whether that scoring shift differs across groups. A nonzero DASB indicates that the move from human to AI scoring changed the group contrast for that item, over and above any uniform recalibration that affects both groups equally.

Usage

```
scoring_bias_test(
  human_mle,
  ai_mle,
  design = c("independent", "paired"),
  cross_cov = NULL,
  metric = c("common", "linked"),
  adjust = "none",
  alpha = 0.05,
  tune = NULL
)
```

Arguments

human_mle	An mle list for the human scoring condition.
ai_mle	An mle list for the AI scoring condition, with the same item and group structure.
design	Either "independent" (default) or "paired". "paired" requires cross_cov.
cross_cov	Optional list with elements group.1 and group.2, each a matrix of cross-condition covariances $\text{Cov}(\hat{\theta}^H, \hat{\theta}^A)$ for that group, with the same dimension and parameter ordering as the entries of human_mle\$var.cov. Only the diagonal is used in this release. Ignored when design = "independent".
metric	Either "common" (default) or "linked". See Identification.
adjust	Multiplicity adjustment applied across items within this DASB analysis. Any method accepted by p.adjust , or "none" (default). The adjustment family is the set of items in this call; it does not span the human-DIF and AI-DIF tests.
alpha	Confidence level complement for the returned intervals. Default 0.05.
tune	Bi-square tuning constant used when metric = "linked". Defaults to $qnorm(1 - \alpha/2)$, matching the convention used by estimate_robust_scale for within-condition scaling, so that both robust steps downweight on the same scale. Do not change this without changing the engine to match.

Value

A data frame of class "dasb", one row per item (per threshold for polytomous items), with columns shift_g1, shift_g2, DASB, se, lower, upper, z, p_val, and — when adjust != "none" — p_adj. Values are unrounded. Attributes record design, metric, adjust, alpha, and, when linked, the estimated offset and its bi-square offset_weights.

Estimand

Let d_{igs} denote the intercept for item i , group $g \in \{R, F\}$, under scoring condition $s \in \{H, A\}$. The within-group scoring shift is

$$\delta_{ig} = d_{igA} - d_{igH}$$

and

$$DASB_i = \delta_{iF} - \delta_{iR} = DIF_i^A - DIF_i^H.$$

Identification

If the four calibrations each carry their own arbitrary metric origin c_{gs} , those constants contribute $(c_{FA} - c_{FH} - c_{RA} + c_{RH})$ to *every* item's DASB. Raw DASB is therefore identified only up to an item-common additive offset, which is perfectly confounded with a uniform DASB affecting all items equally. Two options:

metric = "common" Assumes the four calibrations are already on a commensurate metric, so the offset is zero. This is the 0.1.0 behaviour and is appropriate only when linking was established upstream.

metric = "linked" Estimates the offset from the data with the same bi-square M-estimator the package uses for within-condition robust scaling, and reports DASB relative to it. This assumes the majority of items have no scoring-condition-by-group effect — the same majority assumption that underlies robust DIF scaling.

Inference

With design = "independent" the variance is the sum of the four marginal variances. This is the 0.1.0 formula. It is appropriate when the human and AI calibrations come from disjoint respondent samples.

With design = "paired" — the same responses scored twice — the cross-condition covariance within each group is subtracted:

$$\text{Var}(DASB_i) = \sum_{g,s} \text{Var}(d_{igs}) - 2\text{Cov}(d_{iFH}, d_{iFA}) - 2\text{Cov}(d_{iRH}, d_{iRA}).$$

Cross-group covariances are taken to be zero, since the reference and focal groups comprise disjoint respondents.

Because that cross-condition covariance is typically positive, the independence variance is an *upper bound* in paired designs: using design = "independent" on paired data yields conservative (too large) p-values, not anti-conservative ones.

See Also

[fit_aidif](#), [ai_effect_summary](#)

Examples

```
eg <- make_aidif_eg()

# 0.1.0-equivalent behaviour
scoring_bias_test(eg$human, eg$ai)

# Robustly centred, with Benjamini-Hochberg adjustment
scoring_bias_test(eg$human, eg$ai, metric = "linked", adjust = "BH")
```

simulate_aidif_data	<i>Simulate item parameter estimates for the AI-DIF model.</i>
---------------------	--

Description

Generates a synthetic aidif_data-compatible list suitable for benchmarking and method evaluation. The data-generating model contains: classical DIF in the human scoring condition (controlled via dif_items and dif_mag), differential AI scoring bias (controlled via dasb_items and dasb_mag), and a latent group mean difference (impact).

Usage

```
simulate_aidif_data(
  n_items = 10L,
  n_obs = 500L,
  impact = 0.5,
  dif_items = 1L,
  dif_mag = 0.5,
  dasb_items = 3L,
  dasb_mag = 0.4,
  ai_drift = 0.1,
  seed = 42L
)
```

Arguments

n_items	Integer. Number of items. Default: 10.
n_obs	Integer. Approximate number of observations per group, used to scale the covariance matrices. Default: 500.
impact	Numeric. Latent mean difference (group 2 minus group 1) in SD units. Default: 0.5.
dif_items	Integer vector. Indices of items with DIF in the human scoring condition (intercept shift added to group 2). Default: 1.
dif_mag	Numeric. Magnitude of the intercept DIF effect (in IRT metric). Default: 0.5.
dasb_items	Integer vector. Indices of items where AI scoring introduces differential bias (intercept shift added to group 2 in the AI condition only). Default: 3.

dasb_mag	Numeric. Magnitude of the DASB effect. Default: 0.4.
ai_drift	Numeric. Uniform intercept shift applied to ALL items in BOTH groups under AI scoring (simulates AI calibration offset). Default: 0.1.
seed	Integer seed for reproducibility, or NULL. Default: 42.

Details

Rather than simulating item responses and refitting IRT models (which requires additional dependencies), this function directly simulates maximum-likelihood estimates and their asymptotic covariance matrices, consistent with a 2PL model fitted to `n_obs` observations per group.

Value

A list with elements `human` and `ai`, each formatted identically to the output of `read_ai_scored`. Can be passed directly to `fit_aidif`.

Examples

```
dat <- simulate_aidif_data(
  n_items = 8,
  n_obs   = 600,
  dif_items = c(1, 2),
  dasb_items = 5
)
mod <- fit_aidif(dat$human, dat$ai)
summary(mod)
```

summary.aidif	<i>S3 summary method for class "aidif"</i>
---------------	--

Description

Assembles the DIF test tables for each scoring condition, the DASB table, and the AI-effect classification into a single object.

Usage

```
## S3 method for class 'aidif'
summary(object, adjust = NULL, ...)
```

Arguments

<code>object</code>	An object of class <code>"aidif"</code> .
<code>adjust</code>	Optional multiplicity adjustment for the DASB p-values, recomputed from the unadjusted values. Any <code>p.adjust</code> method, or NULL (default) to keep whatever <code>fit_aidif</code> produced.
<code>...</code>	Further arguments (currently ignored).

Value

An object of class "summary.aidif". Print it to see the report, or extract components directly.

Examples

```
eg <- make_aidif_eg()
mod <- fit_aidif(eg$human, eg$ai)
s <- summary(mod, adjust = "BH")
s$dasb
```

Index

`ai_effect_summary`, [2](#), [12](#)
`anchor_weights`, [3](#)
`as_aidif`, [4](#), [7](#)

`compute_scaling_fn`, [5](#)

`estimate_robust_scale`, [5](#), [6](#), [7](#), [11](#)

`fit_aidif`, [2–4](#), [6](#), [8](#), [10–12](#), [14](#)

`make_aidif_eg`, [7](#), [11](#)

`p.adjust`, [7](#), [11](#), [14](#)
`plot.aidif`, [8](#)
`print.aidif`, [9](#)
`print.summary.aidif`, [10](#)

`read_ai_scored`, [4](#), [7](#), [10](#), [14](#)

`scoring_bias_test`, [3](#), [4](#), [7](#), [11](#)
`simulate_aidif_data`, [8](#), [11](#), [13](#)
`summary.aidif`, [14](#)