

# Package: iDIFr (via r-universe)

June 9, 2026

**Type** Package

**Title** Intersectional Differential Item Functioning Analysis

**Version** 1.0.1

**Description** A toolkit for detecting Differential Item Functioning (DIF) using Logistic Regression (LR) as described in Swaminathan and Rogers (1990) <[doi:10.1111/j.1745-3984.1990.tb00754.x](https://doi.org/10.1111/j.1745-3984.1990.tb00754.x)>, the IRT Likelihood Ratio Test (LRT) following Thissen, Steinberg & Wainer (1993, ISBN:0-8058-0972-4), and model-based recursive partitioning (MOB) as implemented in 'strucchange' following Strobl, Kopf and Zeileis (2015) <[doi:10.1007/s11336-013-9388-3](https://doi.org/10.1007/s11336-013-9388-3)>. Designed for both standard two-group and intersectional multi-group designs, 'iDIFr' prioritises effect size reporting alongside statistical significance, clear guidance on group construction, and interpretable output suitable for applied testing contexts. Built-in Intersectional Contrast Analysis (ICA) classifies items as amplified, pure-intersection, obscured, or none by comparing single-variable and intersectional analyses.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**Depends** R (>= 4.1.0)

**Imports** Rcpp (>= 1.0.0), generics, parallel, stats, cli, dplyr, ggplot2, rlang, strucchange

**LinkingTo** Rcpp

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, openxlsx

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**URL** <https://github.com/thmsrgrs/iDIFr>

**BugReports** <https://github.com/thmsrgrs/iDIFr/issues>

**Config/roxygen2/version** 8.0.0

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

**Date/Publication** 2026-06-02 10:59:08 UTC

**RemoteUrl** https://github.com/thmsrgs/idifr

**RemoteRef** HEAD

**RemoteSha** 1808d163d37fd3105f187d075f9203a807f3300a

## Contents

check_groups . . . . .	2
cross_details . . . . .	3
export_results . . . . .	4
fit_2pl . . . . .	5
group_details . . . . .	6
idifr . . . . .	7
item_loglik . . . . .	9
item_loglik_mg . . . . .	10
merge_groups . . . . .	11
plot.idifr . . . . .	11
print.idifr . . . . .	12
simulate_dif . . . . .	13
summary.idifr . . . . .	14
tidy . . . . .	15
tidy.idifr . . . . .	16
<b>Index</b>	<b>18</b>

---

check_groups	<i>Check group structure and cell sizes before running DIF analysis</i>
--------------	---

---

### Description

Provides a concise summary of the group structure defined by your demographic variables. Reports how many groups meet the recommended minimum cell size, optionally checks which levels of specified variables are fully crossed, and points to `group_details()` and `cross_details()` for full breakdowns.

### Usage

```
check_groups(data, group, min_cell_size = 50, cross_by = NULL, plot = TRUE)
```

**Arguments**

data	A data frame containing demographic variables.
group	A one-sided formula specifying the grouping variable(s), using the same syntax as <code>idifr()</code> . Example: <code>~ gender * nationality</code> .
min_cell_size	Minimum recommended group size. Default is 50.
cross_by	Optional character vector of variable name(s) to check for complete crossing. For each unique value of the named variable(s), the function checks whether every intersectional cell containing that value meets <code>min_cell_size</code> . Example: <code>cross_by = "nationality"</code> reports which nationalities are fully crossed across all other demographic variables. Multiple variables can be supplied: <code>cross_by = c("nationality", "gender")</code> .
plot	Logical. If TRUE (default), prints a heatmap of cell sizes. Only applies when there are at least two grouping variables.

**Value**

An object of class `idifr_groups` (invisibly), which can be passed to `merge_groups()`, `group_details()`, or `cross_details()`.

**See Also**

[group\\_details\(\)](#), [cross\\_details\(\)](#), [merge\\_groups\(\)](#), [idifr\(\)](#)

**Examples**

```
dat <- simulate_dif(300, 10,
  demo_vars = list(nationality = c("UK", "DE", "FR")), seed = 1)
grp <- check_groups(dat, group = ~ group * nationality,
  cross_by = "nationality")
```

---

cross\_details

*Full crossing breakdown for a demographic variable*

---

**Description**

For each unique level of the specified variable, shows whether every intersectional cell containing that level meets the minimum cell size. One row per level, showing how many cells are adequate and the smallest cell size observed.

**Usage**

```
cross_details(grp, cross_by, min_cell_size = NULL)
```

**Arguments**

grp	An idifr_groups object from check_groups().
cross_by	Character vector of variable name(s) to check. Must match variables in the group formula.
min_cell_size	Minimum recommended group size. Overrides the stored value if supplied.

**Value**

The idifr\_groups object, invisibly.

**See Also**

[check\\_groups\(\)](#), [group\\_details\(\)](#)

**Examples**

```
dat <- simulate_dif(300, 10,
  demo_vars = list(nationality = c("UK", "DE", "FR")), seed = 1)
grp <- check_groups(dat, group = ~ group * nationality,
  cross_by = "nationality", plot = FALSE)
cross_details(grp, cross_by = "nationality")
```

---

 export\_results

*Export iDIFr results to Excel*


---

**Description**

Writes an idifr result object to a formatted .xlsx workbook. Each requested sheet is written as an Excel table so that column headers are bold, filters are enabled, and values are properly typed.

Only columns that are actually present in the result object are written; columns listed in the per-method definitions that were not produced by the current run are silently omitted.

**Usage**

```
export_results(x, file, sheets = NULL, overwrite = TRUE)
```

**Arguments**

x	An idifr object returned by <a href="#">idifr()</a> .
file	Path to the output .xlsx file (character string).
sheets	Character vector of sheet keys to include. Valid keys: "summary", "lr", "lrt", "mob", "direction", "ica", "groups". Pass NULL (default) to include all available sheets.
overwrite	Logical. If TRUE (default) an existing file is silently overwritten.

**Value**

x invisibly (so the call can be piped).

**Examples**

```
if (requireNamespace("openxlsx", quietly = TRUE)) {
  dat <- simulate_dif(300, 10, dif_items = c(3, 7), seed = 1)
  result <- idifr(dat, 1:10, ~ group, method = "LR", verbose = FALSE)
  export_results(result, tempfile(fileext = ".xlsx"))
}
```

---

fit\_2pl

*Fit a 2PL IRT model via marginal maximum likelihood (EM)*


---

**Description**

Fit a 2PL IRT model via marginal maximum likelihood (EM)

**Usage**

```
fit_2pl(
  resp,
  group = NULL,
  constrain = "items",
  n_nodes = 15,
  max_iter = 200,
  tol = 1e-04,
  start = NULL,
  verbose = FALSE
)
```

**Arguments**

resp	Integer matrix (0/1/NA). Rows=persons, cols=items.
group	Character/factor vector of group membership (length=nrow(resp)). NULL for single-group calibration.
constrain	Parameter constraint across groups: <ul style="list-style-type: none"> <li>• "items" — a and b equal across groups (DIF null hypothesis)</li> <li>• "none" — all parameters free (separate group calibrations)</li> <li>• "alpha" — a fixed across groups, b free (uniform DIF test)</li> <li>• "beta" — b fixed across groups, a free (non-uniform DIF test)</li> </ul>
n_nodes	Number of quadrature nodes. Default 15. Values of 11-21 are appropriate for DIF detection; use 21 for publication- quality parameter estimates.

max_iter	Maximum EM iterations. Default 200.
tol	Convergence tolerance on log-likelihood change. Default 1e-4.
start	Optional list with elements a and b (numeric vectors of length = number of items) to warm-start the EM loop. If NULL (default), the usual data-driven starting values are used.
verbose	Print iteration log. Default FALSE.

**Value**

Object of class irt\_2pl.

---

group_details	<i>Full per-group cell size breakdown</i>
---------------	---

---

**Description**

Prints a detailed table showing the cell size for every intersectional group, flagging those below the recommended minimum. This is the full breakdown that `check_groups()` summarises in a single line.

**Usage**

```
group_details(grp, min_cell_size = NULL)
```

**Arguments**

`grp` An `idifr_groups` object from `check_groups()`.  
`min_cell_size` Minimum recommended group size. Overrides the stored value if supplied.

**Value**

The `idifr_groups` object, invisibly.

**See Also**

[check\\_groups\(\)](#), [cross\\_details\(\)](#)

**Examples**

```
dat <- simulate_dif(300, 10,
  demo_vars = list(nationality = c("UK", "DE", "FR")), seed = 1)
grp <- check_groups(dat, group = ~ group * nationality, plot = FALSE)
group_details(grp)
```

---

idifr *Run intersectional DIF analysis*

---

## Description

The main entry point for `idifr`. Detects Differential Item Functioning (DIF) using one or more statistical methods, with full support for intersectional group structures defined by crossing multiple demographic variables.

Effect sizes are reported alongside significance for all methods. Groups with small cell sizes trigger a warning. Use `exclude_below_min` and `fully_crossed` to control whether those groups are included in the analysis.

## Usage

```
idifr(
  data,
  items,
  group,
  method,
  ica = FALSE,
  min_cell_size = 50,
  exclude_below_min = FALSE,
  fully_crossed = NULL,
  value_selection = NULL,
  anchor = NULL,
  alpha = 0.05,
  p_adjust = "BH",
  nonuniform_es = "MAPPD",
  verbose = TRUE
)
```

## Arguments

<code>data</code>	A data frame containing item responses and demographic variables.
<code>items</code>	A numeric vector of column indices, or a character vector of column names, identifying the item response columns. Items must be dichotomously scored (0/1).
<code>group</code>	A one-sided formula specifying the grouping variable(s). Use <code>~ var</code> for a single demographic (2+ groups) or <code>~ var1 * var2</code> for intersectional groups. Example: <code>~ gender * nationality * age_band</code> .
<code>method</code>	A character vector specifying which DIF method(s) to use. Must be one or more of "LR" (Logistic Regression), "LRT" (IRT Likelihood Ratio Test), or "MOB" (model-based recursive partitioning). No default – the user must choose.
<code>ica</code>	Logical. If TRUE and the group formula contains two or more variables, runs an Intersectional Contrast Analysis (ICA) after the main analysis: one <code>idifr()</code> per

demographic variable is run silently, and each item is classified as "amplified", "pure\_intersection", "obscured", or "none" based on where it was flagged. The ICA table is stored in `result$ica` and printed by `print()`. If the formula has only one variable, a message is printed and ICA is skipped. Default FALSE.

<code>min_cell_size</code>	Minimum acceptable group size. Groups below this threshold trigger a warning. Also used as the crossing criterion when <code>exclude_below_min = TRUE</code> or <code>fully_crossed</code> is supplied. Default is 50.
<code>exclude_below_min</code>	Logical. If TRUE, any intersectional group with fewer than <code>min_cell_size</code> respondents is excluded from the analysis entirely. If FALSE (default), all groups are included and small groups trigger a warning only.
<code>fully_crossed</code>	A character vector of variable name(s). Only levels of the named variable(s) that are fully crossed – meaning every intersectional cell for that level meets <code>min_cell_size</code> – are included in the analysis. Respondents belonging to levels that are not fully crossed are excluded. Default is NULL (no crossing filter applied). Example: <code>fully_crossed = "nationality"</code> keeps only nationalities where every gender x age_band cell meets <code>min_cell_size</code> .
<code>value_selection</code>	A named list for filtering specific values of demographic variables before analysis. Each element should be named after a grouping variable and contain a character vector of values to keep. Variables not mentioned are left unchanged (all values included). Default is NULL. Example: <code>value_selection = list(country = c("UK", "France"), age_band = c("Young", "Old"))</code> .
<code>anchor</code>	A numeric or character vector identifying anchor items (items assumed to be DIF-free) for IRT scaling. If NULL (default), all items are used as anchors in the first pass.
<code>alpha</code>	Significance level for DIF flagging. Default is 0.05.
<code>p_adjust</code>	Method for p-value adjustment across items. Passed to <code>stats::p.adjust()</code> . Default is "BH" (Benjamini-Hochberg). Use "none" to skip adjustment.
<code>nonuniform_es</code>	Character. The effect size metric to use for non-uniform DIF detection when method includes "LR". One of: "MAPPD" (default) — Maximum Absolute Predicted Probability Difference (probability scale, threshold 0.05); "delta_r2" — Nagelkerke $\Delta R^2$ for the interaction component (threshold 0.035); "chi_sq" — chi-square statistic for the interaction term (threshold 3.84). MAPPD is always computed and stored regardless of this setting.
<code>verbose</code>	Logical. If TRUE (default), prints progress and group information during the analysis.

## Value

An object of class `idifr` containing:

**results** A data frame with one row per item per method, including test statistics, p-values, adjusted p-values, effect sizes, and DIF classification (negligible/moderate/large for all methods).

**groups** An `idifr_groups` object describing the group structure, cell sizes, and any small-cell warnings.

- method** Character vector of methods used.
- call** The matched call.
- items** Character vector of item names analysed.
- alpha** The significance level used.
- p\_adjust** The p-value adjustment method used.
- excluded\_groups** Character vector of group labels excluded by `exclude_below_min` or `fully_crossed`, or NULL if no exclusions.
- excluded\_values** Named list of `value_selection` filters applied, or NULL if none.
- ica** Data frame of ICA classifications (one row per item per method) when `ica = TRUE` and the design is intersectional, otherwise NULL. Columns: `item`, `method`, `ica_class`, `marginal_vars`, `intersectional_flag`.

### See Also

[check\\_groups\(\)](#) for exploring group structure before analysis; [group\\_details\(\)](#) and [cross\\_details\(\)](#) for full breakdowns; [merge\\_groups\(\)](#) for combining sparse cells.

### Examples

```
# Basic two-group analysis using synthetic data
dat <- simulate_dif(300, 10, dif_items = c(3, 7), seed = 1)
result <- idifr(dat, 1:10, ~ group, method = "LR")
print(result)

# Intersectional analysis with ICA
dat_ix <- simulate_dif(500, 10,
  demo_vars = list(nationality = c("UK", "DE", "FR")),
  seed = 2)
result_ix <- idifr(dat_ix, 1:10, ~ group * nationality,
  method = "LR", ica = TRUE)
```

---

item_loglik	<i>Compute per-item log-likelihood contributions from a fitted irt_2pl model</i>
-------------	--

---

### Description

Uses local independence to decompose total LL into item contributions:  $LL_j = \sum_i \log P(x_{ij} | \text{posterior}_i)$  where  $P(x_{ij} | \text{posterior}_i) = \sum_k \text{posterior}[i,k] * P(x_{ij} | \theta_k)$

### Usage

```
item_loglik(model, resp = NULL, post = NULL, gi = 1)
```

**Arguments**

model	An irt_2pl object.
resp	Response matrix (0/1/NA). Defaults to model\$resp.
post	Posterior matrix (persons x nodes). Defaults to model\$posterior.
gi	Group index (integer). Used to select group-specific item params and ability nodes. Default 1.

**Value**

Numeric vector of length n\_items.

---

item_loglik_mg	<i>Per-item LL for a multigroup constrained model</i>
----------------	---

---

**Description**

For the constrained model, each person uses shared item params but their own group-specific ability nodes.

**Usage**

```
item_loglik_mg(model, resp = NULL, post = NULL)
```

**Arguments**

model	An irt_2pl object with constrain != "none".
resp	Response matrix. Defaults to model\$resp.
post	Posterior matrix. Defaults to model\$posterior.

**Value**

Numeric vector of length n\_items.

---

merge_groups	<i>Merge sparse groups</i>
--------------	----------------------------

---

**Description**

Combines sparse intersectional cells by collapsing levels of one or more demographic variables. Returns a modified data frame ready to pass back to `idifr()` or `check_groups()`.

**Usage**

```
merge_groups(groups, grp_formula = NULL, ..., min_cell_size = 50)
```

**Arguments**

<code>groups</code>	An <code>idifr_groups</code> object from <code>check_groups()</code> , or a data frame (in which case <code>grp_formula</code> must also be supplied).
<code>grp_formula</code>	A formula, required only if <code>groups</code> is a raw data frame.
<code>...</code>	Named arguments specifying merge rules. Each should be named after a demographic variable, with a named list mapping new level names to vectors of old level names.
<code>min_cell_size</code>	Minimum cell size to validate against after merging.

**Value**

The original data frame with recoded grouping variable(s).

**Examples**

```
dat <- simulate_dif(300, 10,
  demo_vars = list(nationality = c("UK", "DE", "FR", "ES")), seed = 1)
grp <- check_groups(dat, group = ~ group * nationality, plot = FALSE)
merged <- merge_groups(grp,
  nationality = list("Other" = c("DE", "FR", "ES")))
```

---

plot.idifr	<i>Plot method for idifr objects</i>
------------	--------------------------------------

---

**Description**

Plot method for `idifr` objects

**Usage**

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

**Arguments**

x	An idifr object.
type	Plot type: "items" (default, one row per item showing effect sizes across methods), "concordance" (method agreement heatmap), or "groups" (cell size heatmap from the group structure).
...	Ignored.

**Value**

No return value, called for side effects.

---

print.idifr	<i>Print method for idifr objects</i>
-------------	---------------------------------------

---

**Description**

Print method for idifr objects

**Usage**

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

**Arguments**

x	An idifr object.
...	Ignored.

**Value**

No return value, called for side effects.

---

simulate_dif	<i>Generate synthetic DIF data for testing and simulation</i>
--------------	---

---

### Description

Generates synthetic dichotomous item response data with a known DIF structure. Supports three DIF patterns: standard group DIF ("standard"), DIF confined to a single intersectional cell ("intersection"), and a mixture of both ("mixed").

### Usage

```
simulate_dif(
  n_persons = 500,
  n_items = 20,
  n_groups = 2,
  dif_items = c(3, 7),
  dif_effect = 0.8,
  dif_type = "uniform",
  dif_structure = "standard",
  dif_group = NULL,
  demo_vars = NULL,
  seed = NULL
)
```

### Arguments

n_persons	Integer. Total number of respondents.
n_items	Integer. Number of items. Default 20.
n_groups	Integer. Number of groups. Default 2.
dif_items	Which items have DIF. For dif_structure = "standard" or "intersection", an integer vector (e.g. c(3, 7)). For dif_structure = "mixed", either a named list list(standard = c(3,7), intersection = c(12,15)) or a plain integer vector (first half gets standard DIF, second half intersection DIF). Default c(3, 7).
dif_effect	Numeric. DIF shift size in logits. Default 0.8.
dif_type	"uniform" (difficulty shift only, default) or "nonuniform" (both difficulty and discrimination shifted).
dif_structure	One of "standard" (default), "intersection", or "mixed". "standard" replicates the original behaviour. "intersection" applies DIF only to the specific intersectional cell in dif_group. "mixed" applies standard DIF to some items and intersection DIF to others.
dif_group	Named list identifying the target intersectional cell for intersection DIF. Variable names must match demo_vars or "group". Example: list(group = "G1", nationality = "UK", age_band = "Young"). Required when dif_structure is "intersection" or "mixed".

demo_vars	Named list of additional demographic variables to add, with their levels. Persons are assigned randomly with uniform probability. Example: <code>list(nationality = c("UK", "DE", "FR"), age_band = c("Young", "Old"))</code> . Required when <code>dif_structure</code> is "intersection" or "mixed".
seed	Integer random seed for reproducibility.

### Value

A data frame with item response columns (`item_1`, `item_2`, ...), a group column, and any additional columns specified in `demo_vars`. True item parameters and DIF metadata are stored as attributes.

### Examples

```
# Standard DIF
dat <- simulate_dif(500, 20, 2, c(3, 7), 1.0)

# Intersection-only DIF
dat_ix <- simulate_dif(
  n_persons = 500,
  n_items = 20,
  dif_items = c(5, 12),
  dif_effect = 1.5,
  dif_structure = "intersection",
  dif_group = list(group = "G1", nationality = "UK", age_band = "Young"),
  demo_vars = list(nationality = c("UK", "DE", "FR"),
                   age_band = c("Young", "Old")),
  seed = 42
)

# Mixed DIF
dat_mix <- simulate_dif(
  n_persons = 500,
  n_items = 20,
  dif_items = list(standard = c(3, 7), intersection = c(12, 15)),
  dif_effect = 1.0,
  dif_structure = "mixed",
  dif_group = list(group = "G1", nationality = "UK", age_band = "Young"),
  demo_vars = list(nationality = c("UK", "DE", "FR"),
                   age_band = c("Young", "Old")),
  seed = 42
)
```

### Description

Summary method for idifr objects

**Usage**

```
## S3 method for class 'idifr'  
summary(object, ...)
```

**Arguments**

object	An idifr object.
...	Ignored.

**Value**

No return value, called for side effects.

---

tidy	<i>Tidy an idifr object</i>
------	-----------------------------

---

**Description**

Re-exports [generics::tidy](#) so that `tidy()` is available after `library(iDIFr)` without loading **broom** or **generics** separately. For the iDIFr-specific method see [tidy.idifr](#).

**Usage**

```
tidy(x, ...)
```

**Arguments**

x	An object to tidy. When x is an idifr object the <a href="#">tidy.idifr</a> method is dispatched.
...	Additional arguments passed to the method.

**Value**

A data frame (exact structure depends on the method dispatched).

---

tidy.idifr	<i>Return tidy data frame of DIF results</i>
------------	--

---

### Description

Returns results as a tidy data frame suitable for use with `dplyr`, `ggplot2`, or for export. Use the `table` argument to choose which table to return. Implements the `tidy` generic from the `generics` package so that `tidy()` works correctly regardless of whether `broom` is also loaded.

### Usage

```
## S3 method for class 'idifr'
tidy(x, table = NULL, ...)
```

### Arguments

<code>x</code>	An idifr object.
<code>table</code>	Which table to return. <code>NULL</code> (default) returns the main results table. Other accepted values: <code>"results"</code> One row per item per method. Includes test statistics, p-values, effect sizes, and DIF classification. <code>"direction"</code> One row per group per flagged item. Shows direction and magnitude of DIF for each group. Only available when method includes <code>"LR"</code> . <code>"ica"</code> ICA classification table (one row per item per method). Only available when <code>idifr()</code> was called with <code>ica = TRUE</code> .
<code>...</code>	Ignored.

### Value

A data frame.

### Examples

```
dat <- simulate_dif(300, 10, dif_items = c(3, 7), seed = 1)
result <- idifr(dat, 1:10, ~ group, method = "LR")

# Item-level results (default)
tidy(result)
tidy(result, table = "results")

# Group direction table for flagged items
tidy(result, table = "direction")

# ICA classification table (requires ica = TRUE)
dat_ix <- simulate_dif(500, 10,
  demo_vars = list(nationality = c("UK", "DE")), seed = 2)
result_ix <- idifr(dat_ix, 1:10, ~ group * nationality,
```

```
tidy(result_ix, table = "ica",  
      method = "LR", ica = TRUE)
```

# Index

check\_groups, 2  
check\_groups(), 4, 6, 9  
cross\_details, 3  
cross\_details(), 3, 6, 9  
  
export\_results, 4  
  
fit\_2pl, 5  
  
generics::tidy, 15  
group\_details, 6  
group\_details(), 3, 4, 9  
  
idifr, 7  
idifr(), 3, 4  
item\_loglik, 9  
item\_loglik\_mg, 10  
  
merge\_groups, 11  
merge\_groups(), 3, 9  
  
plot.idifr, 11  
print.idifr, 12  
  
simulate\_dif, 13  
stats::p.adjust(), 8  
summary.idifr, 14  
  
tidy, 15  
tidy.idifr, 15, 16