Package ‘thurstonianIRT’

September 26, 2021

Encoding UTF-8
Type Package
Title Thurstonian IRT Models
Version 0.12.1
Date 2021-09-26

Description Fit Thurstonian Item Response Theory (IRT) models in R. This package supports fitting Thurstonian IRT models and its extensions using 'Stan', 'lavaan', or 'Mplus' for the model estimation. Functionality for extracting results, making predictions, and simulating data is provided as well. References:

License GPL (>= 3)
LazyData true
ByteCompile true
Depends R (>= 3.5.0), Rcpp (>= 0.12.16), methods
Imports dplyr (>= 0.6.0), knitr, lavaan (>= 0.6-1), magrittr,
  MplusAutomation, mvtnorm, RcppParallel (>= 5.0.1), rlang, rstan
  (>= 2.18.1), rstantools (>= 2.1.1), stats, tibble (>= 1.3.1),
  tidyrr, utils
Suggests testthat (>= 0.9.1), rmarkdown
LinkingTo BH (>= 1.66.0-1), Rcpp (>= 0.12.16), RcppEigen (>=
  0.3.3.4.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1),
  StanHeaders (>= 2.18.0)
VignetteBuilder knitr
SystemRequirements GNU make

URL https://github.com/paul-buerkner/thurstonianIRT

BugReports https://github.com/paul-buerkner/thurstonianIRT/issues

NeedsCompilation yes

RoxygenNote 7.1.1
The 'thurstonianIRT' package.

Description

This package fits Thurstonian Item Response Theory (IRT) models using 'Stan', 'lavaan', or 'Mplus'. To bring your data into the right format, use the `make_TIRT_data` function. Models can then be fitted via `fit_TIRT_stan`, `fit_TIRT_lavaan`, or `fit_TIRT_mplus` depending on the desired model fitting engine. Data from Thurstonian IRT models can be simulated via `sim_TIRT_data`.

References


cor_matrix

Set up Correlation Matrices

Description
Set up Correlation Matrices

Usage
cor_matrix(cors, dim, dimnames = NULL)

Arguments
- cors: vector of unique correlations
- dim: Dimension of the correlation matrix
- dimnames: Optional dimnames of the correlation matrix

Value
A correlation matrix of dimension dim.

Examples
cor_matrix(c(0.2, 0.3, 0.5), dim = 3)

fit_TIRT_lavaan
Fit Thurstonian IRT models in lavaan

Description
Fit Thurstonian IRT models in lavaan

Usage
fit_TIRT_lavaan(data, estimator = "ULSMV", ...)

Arguments
- data: An object of class 'TIRTdata'. see make_TIRT_data for documentation on how to create one.
- estimator: Name of the estimator that should be used. See lavOptions.
- ...: Further arguments passed to lavaan.
Value

A 'TIRTfit' object.

Examples

# load the data
data("triplets")

# define the blocks of items
blocks <-
  set_block(c("i1", "i2", "i3"), traits = c("t1", "t2", "t3"),
           signs = c(1, 1, 1)) +
  set_block(c("i4", "i5", "i6"), traits = c("t1", "t2", "t3"),
           signs = c(-1, 1, 1)) +
  set_block(c("i7", "i8", "i9"), traits = c("t1", "t2", "t3"),
           signs = c(1, 1, -1)) +
  set_block(c("i10", "i11", "i12"), traits = c("t1", "t2", "t3"),
           signs = c(1, -1, 1))

# generate the data to be understood by 'thurstonianIRT'
triplets_long <- make_TIRT_data(
  data = triplets, blocks = blocks, direction = "larger",
  format = "pairwise", family = "bernoulli", range = c(0, 1)
)

# fit the data using lavaan
fit <- fit_TIRT_lavaan(triplets_long)
print(fit)
predict(fit)


# fit_TIRT_stan

## Description

Fit Thurstonian IRT models in Stan

## Usage

```r
fit_TIRT_stan(data, init = 0, ...)
```

## Arguments

- **data**: An object of class 'TIRTdata'. see `make_TIRT_data` for documentation on how to create one.
- **init**: Initial values of the parameters. Defaults to 0 as it proved to be most stable.
- **...**: Further arguments passed to `rstan::sampling`.

---

**Value**

A 'TIRTfit' object.

**Examples**

```r
# load the data
data("triplets")

# define the blocks of items
blocks <-
set_block(c("i1", "i2", "i3"), traits = c("t1", "t2", "t3"),
  signs = c(1, 1, 1)) +
set_block(c("i4", "i5", "i6"), traits = c("t1", "t2", "t3"),
  signs = c(-1, 1, 1)) +
set_block(c("i7", "i8", "i9"), traits = c("t1", "t2", "t3"),
  signs = c(1, 1, -1)) +
set_block(c("i10", "i11", "i12"), traits = c("t1", "t2", "t3"),
  signs = c(1, -1, 1))

# generate the data to be understood by 'thurstonianIRT'
triplets_long <- make_TIRT_data(
  data = triplets, blocks = blocks, direction = "larger",
  format = "pairwise", family = "bernoulli", range = c(0, 1))

# fit the data using Mplus
fit <- fit_TIRT_mplus(triplets_long)
print(fit)
predict(fit)
```
Value

A 'TIRTfit' object.

Examples

```r
# load the data
data("triplets")

# define the blocks of items
blocks <-
  set_block(c("i1", "i2", "i3"), traits = c("t1", "t2", "t3"),
            signs = c(1, 1, 1)) +
  set_block(c("i4", "i5", "i6"), traits = c("t1", "t2", "t3"),
            signs = c(-1, 1, 1)) +
  set_block(c("i7", "i8", "i9"), traits = c("t1", "t2", "t3"),
            signs = c(1, 1, -1)) +
  set_block(c("i10", "i11", "i12"), traits = c("t1", "t2", "t3"),
            signs = c(1, -1, 1))

# generate the data to be understood by 'thurstonianIRT'
triplets_long <- make_TIRT_data(
  data = triplets, blocks = blocks, direction = "larger",
  format = "pairwise", family = "bernoulli", range = c(0, 1)
)

# fit the data using Stan
fit <- fit_TIRT_stan(triplets_long, chains = 1)
print(fit)
predict(fit)
```

---

gof.TIRTfit

**Extract corrected goodness of fit statistics**

**Description**

By default lavaan will return a value for degrees of freedom that ignores redundancies amongst the estimated model thresholds. This function corrects the degrees of freedom, and then recalculates the associated chi-square test statistic p-value and root mean square error of approximation (RMSEA).

**Usage**

```r
## S3 method for class 'TIRTfit'
gof(object, ...)

gof(object, ...)
```
make_lavaan_code

Generate lavaan code for Thurstonian IRT models

Description

Generate lavaan code for Thurstonian IRT models

Usage

make_lavaan_code(data)
**make_mplus_code**

Generate Mplus code for Thurstonian IRT models

**Description**

Generate Mplus code for Thurstonian IRT models

**Usage**

```r
make_mplus_code(data, iter = 1000, eta_file = "eta.csv")
```

**Arguments**

- `data` An object of class 'TIRTdata'. see `make_TIRT_data` for documentation on how to create one.
- `iter` Maximum number of iterations of the model fitting algorithm.
- `eta_file` optional file name in which predicted trait scores should be stored.

**Value**

A list of Mplus code snippets to be interpreted by the `MplusAutomation` package.
Examples

```r
sim_data <- sim_TIRT_data(
  npersons = 100,
  ntraits = 3,
  nblocks_per_trait = 4,
  gamma = 0,
  lambda = c(runif(6, 0.5, 1), runif(6, -1, -0.5)),
  Phi = diag(3)
)

# show the created Mplus code
lapply(make_mplus_code(sim_data), cat)
```

make_sem_data

Prepare data for Thurstonian IRT models fitted with lavaan or Mplus

Description

Prepare data for Thurstonian IRT models fitted with lavaan or Mplus

Usage

```r
make_sem_data(data)
```

Arguments

- data: An object of class 'TIRTdata'. see `make_TIRT_data` for documentation on how to create one.

Value

A data.frame ready to be passed to `lavaan` or `Mplus`.

Examples

```r
# simulate some data
sdata <- sim_TIRT_data(
  npersons = 100,
  ntraits = 3,
  nblocks_per_trait = 4,
  gamma = 0,
  lambda = c(runif(6, 0.5, 1), runif(6, -1, -0.5)),
  Phi = diag(3)
)

# create data ready for use in SEM software
sem_data <- make_sem_data(sdata)
head(sem_data)
```
**make_stan_data**

Prepare data for Thurstonian IRT models fitted with Stan

**Description**

Prepare data for Thurstonian IRT models fitted with Stan

**Usage**

```r
make_stan_data(data)
```

**Arguments**

- `data`: An object of class `data.frame` containing data of all variables used in the model.

**Value**

A list of data ready to be passed to Stan.

```r
make_TIRT_data
```

**Description**

Prepare data for Thurstonian IRT models

**Usage**

```r
make_TIRT_data(
  data, blocks, direction = c("larger", "smaller"), format = c("ranks", "pairwise"), family = "bernoulli", partial = FALSE, range = c(0, 1)
)
```
Arguments

data  An object of class data.frame containing data of all variables used in the model.

blocks  Object of class TIRTBblocks generated by set_block indicating which items belong to which block, trait and more. Ignored if data already contains information on the blocks.

direction  Indicates if "larger" (the default) or "smaller" input values are considered as indicating the favored answer.

format  Format of the item responses. Either "ranks" for responses in ranked format or "pairwise" for responses in pairwise comparison format. If "ranks", each item must have its own column in the data frame which contains its ranks within the block. If "pairwise", each existing item combination must have its own column named after the combination of the two compared items.

family  Name of assumed the response distribution. Either "bernoulli", "cumulative", or "gaussian".

partial  A flag to indicate whether partial comparisons are allowed for responses stored in the "ranks" format.

range  Numeric vector of length two giving the range of the responses when using the "pairwise" format. Defaults to c(0, 1) for use with dichotomous responses.

Value

A data.frame in a specific format and with attributes ready for use with other functions of the ThurstonianIRT package.

Examples

```r
# load the data
data("triplets")

# define the blocks of items
blocks <-
    set_block(c("i1", "i2", "i3"), traits = c("t1", "t2", "t3"), signs = c(1, 1, 1)) +
    set_block(c("i4", "i5", "i6"), traits = c("t1", "t2", "t3"), signs = c(-1, 1, 1)) +
    set_block(c("i7", "i8", "i9"), traits = c("t1", "t2", "t3"), signs = c(1, 1, -1)) +
    set_block(c("i10", "i11", "i12"), traits = c("t1", "t2", "t3"), signs = c(1, -1, 1))

# generate the data to be understood by 'thurstonianIRT'
triplets_long <- make_TIRT_data(
    data = triplets, blocks = blocks, direction = "larger",
    format = "pairwise", family = "bernoulli", range = c(0, 1)
)
```
# fit the data using Stan
fit <- fit_TIRT_stan(triplets_long, chains = 1)
print(fit)
predict(fit)

---

**predict.TIRTfit**  
*Predict trait scores of Thurstonian IRT models*

**Description**

Predict trait scores of Thurstonian IRT models

**Usage**

```r
## S3 method for class 'TIRTfit'
predict(object, newdata = NULL, ...)
```

**Arguments**

- `object`  
  An object of class TIRTfit.

- `newdata`  
  Optional TIRTdata object (created via `make_TIRT_data`) containing data of new persons for which trait scores should be predicted based on the fitted model. If NULL (the default), trait scores are predicted for the persons whose data was used to originally fit the model.

- `...`  
  Further arguments passed to the underlying methods.

**Details**

When predicting trait scores of new persons (via `newdata`), posterior medians of item parameters are used for predictions. This implies that the uncertainty in the new trait scores is underestimated as the uncertainty in the (posterior distribution of) item parameters is ignored.

**Value**

A data frame with predicted trait scores.
**set_block**  

Prepare blocks of items

**Description**

Prepare blocks of items and incorporate information about which item belongs to which trait. A block of items is a set of two or more items presented and answered together by fully ranking them or selecting the most and/or least favorit in a forced choice format. A whole test usually contains several blocks and items may reappear in different blocks.

**Usage**

```r
set_block(items, traits, names = items, signs = 1)
empty_block()
```

**Arguments**

- **items**: Names of item comparisons to be combined into one block. Should correspond to variables in the data.
- **traits**: Names of the traits to which each item belongs
- **names**: Optional names of the items in the output. Can be used to equate parameters of items across blocks, if the same item was used in different blocks.
- **signs**: Expected signs of the item loadings (1 or -1).

**See Also**

- `set_blocks_from_df`

**Examples**

```r
set_block(
  items = c("i1", "i2", "i3"),
  traits = c("A", "B", "C")
) +
set_block(
  items = c("i4", "i5", "i6"),
  traits = c("A", "B", "C")
)
```
set_blocks_from_df

Prepare blocks of items from a data frame

Description

Prepare blocks of items and incorporate information about which item belongs to which trait from a pre-existing dataframe. This is a wrapper function for set_block, eliminating the need to manually set each item, trait, name and sign (loading) info per block.

Usage

set_blocks_from_df(
  data,
  blocks = "block",
  items = "item",
  traits = "trait",
  names = items,
  signs = "sign"
)

Arguments

data A data.frame containing all the required columns (see the arguments below) to specify the item blocks.
blocks Name of column vector denoting the block each item corresponds to. Each block must have an equal number of items.
items Name of column vector denoting items to be combined into one block. Should correspond to variables in the data.
traits Names of column vector denoting the traits to which each item belongs.
names Optional column vector of item names in the output. Can be used to equate parameters of items across blocks, if the same item was used in different blocks.
signs Name of column vector with expected signs of the item loadings (1 or -1).

Details

A block of items is a set of two or more items presented and answered together by fully ranking them or selecting the most and/or least favorite in a forced choice format. A whole test usually contains several blocks and items may reappear in different blocks.

See Also

set_block
Examples

```r
block_info <- data.frame(
  block = rep(1:4, each = 3),
  items = c("i1", "i2", "i3", "i4", "i5", "i6",
             "i7", "i8", "i9", "i10", "i11", "i12"),
  traits = rep(c("t1", "t2", "t3"), times = 4),
  signs = c(1, 1, 1, -1, 1, 1, 1, -1, 1, -1, 1)
)
blocks <- set_blocks_from_df(
  data = block_info,
  blocks = "block",
  items = "items",
  traits = "traits",
  signs = "signs"
)
```

Description

Simulate Thurstonian IRT data

Usage

```r
sim_TIRT_data(
  npersons,
  ntraits,
  lambda,
  gamma,
  psi = NULL,
  Phi = NULL,
  eta = NULL,
  family = "bernoulli",
  nblocks_per_trait = 5,
  nitems_per_block = 3,
  comb_blocks = c("random", "fixed")
)
```

Arguments

- `npersons` Number of persons.
- `ntraits` Number of traits.
- `lambda` Item factor loadings.
**gamma**
Baseline attractiveness parameters of the first item versus the second item in the pairwise comparisons. Can be thought of as intercept parameters.

**psi**
Optional item uniquenesses. If not provided, they will be computed as psi = 1 - lambda^2 in which case lambda are taken to be the standardized factor loadings.

**Phi**
Optional trait correlation matrix from which to sample person factor scores. Only used if `eta` is not provided.

**eta**
Optional person factor scores. If provided, argument `Phi` will be ignored.

**family**
Name of assumed the response distribution. Either "bernoulli", "cumulative", or "gaussian".

**nblocks_per_trait**
Number of blocks per trait.

**nitems_per_block**
Number of items per block.

**comb_blocks**
Indicates how to combine traits to blocks. "fixed" implies a simple non-random design that may combine certain traits which each other disproportionally often. We thus recommend to use a "random" block design (the default) that combines all traits with all other traits equally often on average.

**Value**
A data.frame of the same structure as returned by `make_TIRT_data`. Parameter values from which the data were simulated are stored as attributes of the returned object.

**Examples**

```r
# simulate some data
sdata <- sim_TIRT_data(
  npersons = 100,
  ntraits = 3,
  nblocks_per_trait = 4,
  gamma = 0,
  lambda = c(runif(6, 0.5, 1), runif(6, -1, -0.5)),
  Phi = diag(3)
)

# take a look at the data
head(sdata)
str(attributes(sdata))

# fit a Thurstonian IRT model using lavaan
fit <- fit_TIRT_lavaan(sdata)
print(fit)
```
Description

This data set contains synthetic data of 200 participants on 4 triplets. In each triplet, participants had to rank the three alternative items according to their preference. Responses were then converted into a set of dichotomous pairwise responses between all the three alternatives. More details can be found in Brown and Maydeu-Olivares (2011).

Usage

triplets

Format

A data frame of 200 observations containing information on 12 variables. Overall, the 12 items measure 3 different traits. Items 1, 4, 7, and 10 load on trait 1, items 2, 5, 8, and 11 load on trait 2, and items 3, 6, 9, and 12 load on trait 3. Moreover, items 4, 9, and 11 are inverted.

i1i2 Response preferences between item 1 and 2.
i1i3 Response preferences between item 1 and 3.
i2i3 Response preferences between item 2 and 3.
i4i5 Response preferences between item 4 and 5.
i4i6 Response preferences between item 4 and 6.
i5i6 Response preferences between item 5 and 6.
i7i8 Response preferences between item 7 and 8.
i7i9 Response preferences between item 7 and 9.
i8i9 Response preferences between item 8 and 9.
i10i11 Response preferences between item 10 and 11.
i10i12 Response preferences between item 10 and 12.
i11i12 Response preferences between item 11 and 12.

Source

Examples

```r
# load the data
data("triplets")

# define the blocks of items
blocks <-
  set_block(c("i1", "i2", "i3"), traits = c("t1", "t2", "t3"),
            signs = c(1, 1, 1)) +
  set_block(c("i4", "i5", "i6"), traits = c("t1", "t2", "t3"),
            signs = c(-1, 1, 1)) +
  set_block(c("i7", "i8", "i9"), traits = c("t1", "t2", "t3"),
            signs = c(1, 1, -1)) +
  set_block(c("i10", "i11", "i12"), traits = c("t1", "t2", "t3"),
            signs = c(1, -1, 1))

# generate the data to be understood by 'thurstonianIRT'
tdat <- make_TIRT_data(
  triplets, blocks, direction = "larger",
  format = "pairwise", family = "bernoulli", range = c(0, 1))

# fit the data using Stan
fit <- fit_TIRT_stan(tdat, chains = 1)
print(fit)
predict(fit)
```
Index

* datasets
  triplets, 17

cor_matrix, 3

empty_block (set_block), 13

fit_TIRT_lavaan, 2, 3
fit_TIRT_mplus, 2, 4
fit_TIRT_stan, 2, 5

gof (gof.TIRTfit), 6
gof.TIRTfit, 6

lavaan, 3
lavOptions, 3

make_lavaan_code, 7
make_mplus_code, 8
make_sem_data, 9
make_stan_data, 10
make_TIRT_data, 2–5, 8, 9, 10, 12, 16
mplusModeler, 4

predict.TIRTfit, 12
rstan::sampling, 5

set_block, 11, 13, 14
set_blocks_from_df, 13, 14
sim_TIRT_data, 2, 15

thurstonianIRT
  (thurstonianIRT-package), 2
thurstonianIRT-package, 2
triplets, 17