Package 'D3mirt'

August 8, 2023

Title Descriptive 3D Multidimensional Item Response Theory Modeling

Version 1.1.0

Description For identifying, estimating, and plotting descriptive multidimensional item response theory models, restricted to 3D and dichotomous or polytomous data that fit the two-parameter logistic model or the graded response model. The method is foremost explorative and centered around the plot function that exposes item characteristics and constructs, represented by vector arrows, located in a three-dimensional interactive space. The results can be useful for item-level analysis as well as test development.

License GPL (>= 3)

URL https://github.com/ForsbergPyschometrics/D3mirt

BugReports https://github.com/ForsbergPyschometrics/D3mirt/issues

Depends mirt, R (>= 3.5.0), rgl (>= 1.0.1)

Suggests knitr, rmarkdown, stats

Encoding UTF-8

RoxygenNote 7.2.3

LazyData true

VignetteBuilder knitr

NeedsCompilation no

Author Erik Forsberg [aut, cre, cph] (<https://orcid.org/0000-0002-5228-9729>)

Maintainer Erik Forsberg <forsbergpsychometrics@gmail.com>

Repository CRAN

Date/Publication 2023-08-08 08:40:11 UTC

R topics documented:

`anes0809offwaves` *Moral Items from The anes0809offwaves data set*

Description

A subset of data (*N* = 1046, *Mean age* = 51.33, *SD* = 14.56, 57% Female) from the ANES 2008-2009 Panel Study Off Wave Questionnaires, December 2009 (DeBell et al., 2010), with complete responses on a pilot version of the Judgment subscale from what became the Moral Foundations Questionnaire (Graham et al., 2011). Demographic variables include age and gender (two levels) and Likert-items positively scored on a scale from 1 = *Strongly Disagree* to 6 = *Strongly Agree*.

Usage

```
anes0809offwaves
```

Format

`anes0809offwaves`**:**

A data frame with 1046 rows and 22 columns:

W3Xage Age

W3XGENDER Gender (Male = 1, Female = 2)

W7Q1 When one of my loved ones needs my attention, I really try to slow down and give them the time and help they need

W7Q2 I am known by family and friends as someone who makes time to pay attention to others' problems

W7Q3 I'm the kind of person who is willing to go the "extra mile" to help take care of my friends, relatives, and acquaintances

W7Q4 When friends or family members experience something upsetting or discouraging I make a special point of being kind to them

W7Q5 When it comes to my personal relationships with others, I am a very generous person

W7Q6 It makes me very happy to give to other people in ways that meet their needs

W7Q7 It is just as important to me that other people around me are happy and thriving as it is that I am happy and thriving

W7Q8 My decisions are often based on concern for the welfare of others

W7Q9 I am usually willing to risk my own feelings being hurt in the process if I stand a chance of helping someone else in need

W7Q10 I make it a point to let my friends and family know how much I love and appreciate them

W7Q11 Compassion for those who are suffering is the most crucial virtue

W7Q12 One of the worst things a person could do is hurt a defenseless animal

W7Q13 When the government makes laws, the number one principle should be ensuring that everyone is treated fairly

W7Q14 Justice is the most important requirement for a society

W7Q15 I am proud of my country's history

W7Q16 People should be loyal to their family members, even when they have done something wrong

W7Q17 Respect for authority is something all children need to learn

W7Q18 Men and women each have different roles to play in society

W7Q19 People should not do things that are disgusting, even if no one is harmed

W7Q20 I would call some acts wrong on the grounds that they are unnatural ...

Source

https://electionstudies.org/data-center/2008-2009-panel-study/

References

DeBell, M., Krosnick, J. A., & Lupia, A. (2010). *Methodology Report and User's Guide for the 2008–2009 ANES Panel Study*. Palo Alto, CA, and Ann Arbor, MI: Stanford University and the University of Michigan.

Graham, J., Nosek, B. A., Haidt, J., Iyer, R., Koleva, S., & Ditto, P. H. (2011). Mapping the moral domain. *Journal of Personality and Social Psychology, 101*(2), 366–385. https://doi.org/10.1037/a0021847

Examples

```
data(anes0809offwaves)
```

`D3mirt` *3D DMIRT Model Estimation*

Description

Descriptive multidimensional item response theory model estimation (DMIRT; Reckase, 2009, 1985, Reckase & McKinley, 1991) for dichotomous and polytomous items restricted to three dimensions.

Usage

```
D3mirt(x, constructs = NULL)
```

Arguments

`x`	A data frame with rows for items and columns for model parameters or an S4 object of class 'SingleGroupClass' exported from mirt::mirt (Chalmers, 2012). Regarding the data frame, the number of columns must be more than or equal to 4, i.e., three columns with (a) parameters and at least one column for difficulty (d) parameters.
`constructs`	Optional. Nested lists with integer values indicating construct. The default is `constructs = NULL`.

Details

The `D3mirt()` function takes in model parameters from a compensatory three-dimensional multidimensional two-parameter logistic model (M2PL) or a multidimensional graded response model (MGRM), either in the form of a data frame or an S4 object of class 'SingleGroupClass' exported from mirt::mirt (Chalmers, 2012) function fitted in accordance with the descriptive item response theory model specifications described below. The function returns DMIRT estimates that can be visualized with plot that graph vector arrows representing item response characteristics in a three-dimensional space. Regarding the former, this includes visualization of the single multidimensional discrimination (MDISC) parameter and the multidimensional difficulty (MDIFF) parameters (Reckase, 2009, 1985; Reckase & McKinley, 1991).

The user has the option of including constructs in the estimation. Constructs, in this context, refer to the assumption that a subset of items can measure a higher-order latent variable. To include constructs, the user must create one or more nested lists that indicate what items belong to what construct (from one item up to all items in the set; see the examples section below). From this, the `D3mirt()` function calculates the average direction by adding and normalizing the direction cosines using the items in the nested lists. Constructs are visualized when plotting as solid black arrows running across the model space. In addition, if constructs are used the output will also contain the directional discrimination (DDISC) parameters for all items assessed in the direction indicated by the construct vectors. This makes it possible to compare item discrimination under the assumption that they measure the same latent variable.

Note, model parameters from the multidimensional M2PL or MGRM must be assessed prior to using the `D3mirt()` function (see examples section below or the package vignette). This means that the model must first be identified (see modid for more on model identification). For more on theory and how to interpret statistical estimates, please see the package vignette.

Value

A S3 object of class `D3mirt` with lists of *a* and *d* parameters from the M2PL or MGRM estimation, multidimensional difficulty (MDIFF), multidimensional discrimination (MDISC), direction cosines and degrees for vector angles, construct lists, and vector coordinates.

Author(s)

Erik Forsberg

References

Chalmers, R., P. (2012). mirt: A Multidimensional Item Response Theory Package for the R Environment. *Journal of Statistical Software, 48*(6), 1-29.

Reckase, M. D. (2009). *Multidimensional Item Response Theory*. Springer.

Reckase, M. D. (1985). The Difficulty of Test Items That Measure More Than One Ability. *Applied Psychological Measurement, 9*(4), 401-412. https://doi-org.ezp.sub.su.se/10.1177/014662168500900409

Reckase, M. D., & McKinley, R. L. (1991). The Discriminating Power of Items That Measure More Than One Dimension. *Applied Psychological Measurement, 15*(4), 361-373. https://doi-org.ezp.sub.su.se/10.1177/014662169101500407

Examples

```
# Load data
data("anes0809offwaves")
x <- anes0809offwaves
x <- x[,3:22] # Remove columns for age and gender

# Fit a three-dimensional graded response model with orthogonal factors
# Example below uses Likert items from the built-in data set "anes0809offwaves"
# Item W7Q3 and item W7Q20 was selected with `modid()`
# The model specification set all items in the data set (1-20)
# to load on all three factors (F1-F3)
# The START and FIXED commands are used on the two items to identify the DMIRT model
spec <- '  F1 = 1-20
           F2 = 1-20
           F3 = 1-20

           START=(W7Q3,a2,0)
           START=(W7Q3,a3,0)

           START=(W7Q20,a3,0)

           FIXED=(W7Q3,a2)
           FIXED=(W7Q3,a3)

           FIXED=(W7Q20,a3) '


mod1 <- mirt::mirt(x,
                   spec,
                   itemtype = 'graded',
                   SE = TRUE,
                   method = 'QMCEM')

# Optional: Load the mod1 data as a data frame directly from the package file
# load(system.file("extdata/mod1.Rdata", package = "D3mirt"))

# Call D3mirt() using the mod1 as input
g <- D3mirt(mod1)

# Show summary of results
summary(g)

# Call to D3mirt(), including optional nested lists for three constructs
# Item W7Q16 is not included in any construct because of model violations
# The model violations for the item W7Q16 can be seen when plotting the model
c <- list(list(1,2,3,4,5,6,7,8,9,10),
          list(11,12,13,14),
          list(15,17,18,19,20))
g <- D3mirt(mod1, c)

# Show summary of results
```

```
summary(g)
```

`modid` *D3mirt Model Identification*

Description

`modid()` performs model identification for descriptive multidimensional item response theory (DMIRT) models by indicating what items, from a set or scale, to use to identify the DMIRT model.

Usage

```
modid(
  x,
  efa = TRUE,
  factors = 3,
  lower = 0.5,
  upper = 0.1,
  fac.order = NULL,
  itemtype = "graded",
  method = "EM",
  rotate = "oblimin",
  ...
)
```

Arguments

`x`	A data frame with item data or item factor loadings that fit the multidimensional graded response model (MGRM) or the multidimensional 2-parameter logistic model (M2PL).
`efa`	Logical, if the data should be explored with exploratory factor analysis (EFA). The default is `efa = TRUE`.
`factors`	The number of factors for the exploratory factor analysis. The default is `factors = 3`.
`lower`	The lower bound for the item pool, calculated using the standard deviation of scaled item factor loadings. The default is `lower = 0.5`.
`upper`	The upper bound for the filtering of absolute sum scores less than or equal to the indicated value. The default is `upper = .10`
`fac.order`	Optional. Users can override the automatic sorting of factors by manually indicating factor order with integer values, e.g., `c(2,1,3)` to start with the second factor (or column) in data frame x, followed by the first factor (or column) in x, and then lastly the third factor (or column). The default is `fac.order = NULL`.
`itemtype`	The item model for the exploratory factor analysis. Note, only item type 'graded' (for the MGRM) or '2PL' (for the M2PL) are allowed. The default is `itemtype = "graded"`. See mirt::mirt (Chalmers, 2012) for more on item models.

`method`	A string indicating what integration algorithm to use for the EFA. The default is `method = 'QMCEM'`. See mirt::mirt (Chalmers, 2012) for more on methods.
`rotate`	A string indicating what rotation method to use for the EFA. The default is `rotate = "oblimin"`. See mirt::mirt (Chalmers, 2012) for more on rotations.
`...`	Any additional arguments passed to mirt().

Details

Before performing DMIRT analysis it is necessary to identify the compensatory model (Reckase, 2009). For a three-dimensional model, this entails that two items must be chosen and their loadings restricted as follows. The first item is fixed not to load on the second and third axes (y and z), while the second item is fixed not to load on the third axis (z). If this can be empirically achieved, it will be possible to create the orthogonal structure necessary for a three-dimensional DMIRT model.

The `modid()` function can help by suggesting what items to use for this purpose. This is done by the function by first performing an EFA on the data and then selecting the strongest loading items, following the order of strength of the factors, in accordance with the statistical assumptions described above. This orders the entire model so that the strongest loading item, from the strongest factor, always aligns with the x-axis and the other items follow thereon. Note, the `modid()` function is not limited to three-dimensional analysis and can be used to identify a DMIRT model on any number of dimensions.

Because `D3mirt` analysis is based on the M2PL and the MGRM, it is recommended to use multidimensional item response theory EFA methods, such as the EFA option in mirt::mirt (Chalmers, 2012) with `itemtype = 'graded'` or `'2PL'`, so that the EFA is performed with the proper item model. For this reason, the `mirt()` function is integrated into `modid()` so that the user needs only to provide the data frame containing empirical item data in the first argument in the call to the function. Accordingly, in the default mode (`efa = TRUE`), using raw item data, the function performs an EFA with three factors as default (`factors = 3`), and then finishes with the model identification.

However, it is also possible to use the `modid()` function without performing the EFA by setting `efa = FALSE`, if, for instance, a factor loading data frame is already available. This allows the function to move directly to the model identification step.

Note, the EFA is only used to find model identification items that meet the necessary DMIRT model specification requirements. The EFA model itself is discarded after this step in the procedure and the user can, therefore, try different types of rotation methods and compare the results.

Running the function prints the number of items and factors together with the suggested model identification items to the console and the summary function is used to inspect the full results. The latter includes data frames that hold all the model identification items (`Item.1...Item.n`) selected by `modid()` together with the items absolute sum score (`ABS`), one frame for the sum of squares for factors sorted in descending order, and one frame for item factor loadings. The order of the factors follows the model identification items so that item 1 comes from the strongest factor, item 2 from the second strongest factor, and so on.

Model identification items should preferably (a) have an absolute sum score of less than or equal to .10 and (b) have the highest factor loading scores on the factor of interest. Of these two criteria, (a) should be given the strongest weight in the selection decision. If these conditions cannot be met, the user is advised to proceed with caution since the loading scores, therefore, imply that an adequate orthogonal structure may not be empirically attainable. For more details on the model identification process and troubleshooting please see the package vignette.

Value

A S3 object of class `modid` with lists of items and absolute sum scores, sorted by the latter, and sum of squared factor loadings and frame with raw factor loadings with columns ordered on explained variance (high to low) or according to user settings.

Author(s)

Erik Forsberg

References

Chalmers, R., P. (2012). mirt: A Multidimensional Item Response Theory Package for the R Environment. *Journal of Statistical Software, 48*(6), 1-29.

Reckase, M. D. (2009). *Multidimensional Item Response Theory*. Springer.

Examples

```
# Load data
data("anes0809offwaves")
x <- anes0809offwaves
x <- x[,3:22] # Remove columns for age and gender

# Identify the DMIRT model using a three-factor EFA
g <- modid(x)

# Optional: Load the EFA data for this example directly from the package file
# load(system.file("extdata/efa.Rdata", package = "D3mirt"))

# Call to summary
summary(g)

# Call to modid with increased lower and upper bound
modid(x, lower = 1, upper = 1 )

# Override factor order by reversing columns in the original data frame
modid(x, fac.order = c(3,2,1))
```

---

plot.D3mirt *Plot Method for Objects of Class* D3mirt

Description

For graphing of objects of class `D3mirt` from the `D3mirt()` function using the rgl 3D visualization device system (Adler & Murdoch, 2022).

Usage

```
## S3 method for class 'D3mirt'
plot(
  x,
  scale = FALSE,
  hide = FALSE,
  diff.level = NULL,
  items = NULL,
  item.names = TRUE,
  item.lab = NULL,
  constructs = FALSE,
  construct.lab = NULL,
  adjust.lab = c(0.5, -0.8),
  x.lab = "X",
  y.lab = "Y",
  z.lab = "Z",
  title = "",
  line = -5,
  axis.scalar = 1.1,
  axis.length = NULL,
  axis.col = "black",
  axis.points = "black",
  points = TRUE,
  axis.ticks = TRUE,
  nticks = 4,
  width.rgl.x = 1040,
  width.rgl.y = 1040,
  view = c(15, 20, 0.6),
  show.plane = TRUE,
  plane.col = "grey80",
  background = "white",
  type = "rotation",
  col = c("black", "grey20", "grey40", "grey60", "grey80"),
  arrow.width = 0.6,
  n = 20,
  theta = 0.2,
  barblen = 0.03,
  c.scalars = c(1, 1),
  c.type = "rotation",
  c.col = "black",
  c.arrow.width = 0.6,
  c.n = 20,
  c.theta = 0.2,
  c.barblen = 0.03,
  profiles = NULL,
  levels = NULL,
  sphere.col = c("black", "grey20", "grey40", "grey60", "grey80"),
  spheres.r = 0.05,
```

```
  ci = FALSE,
  ci.level = 0.95,
  ellipse.col = "grey80",
  ellipse.alpha = 0.2,
  ...
)
```

Arguments

x	A S3 object of class D3mirt.
scale	Logical, if item vector arrow length should visualize the MDISC. If set to FALSE, the vector arrow length will be of one unit length. The default is scale = FALSE.
hide	Logical, if items should be plotted. The default is hide = FALSE.
diff.level	Optional. Plotting of a single level of difficulty indicated by an integer.
items	Optional. The user can input a list of integers indicating what item vector arrows will be visible while the remaining item vector arrows are hidden.
item.names	Logical, if item labels should be plotted. The default is item.names = TRUE.
item.lab	Optional. String vector of item names that will override row names extracted from the data frame. Note, row names are not overwritten. Instead, the string vector in item.lab prints item labels on the item vector arrows currently displayed following the order of item vector arrows in the graphical output. For example, when plotting in the default mode (plotting all item vectors) the labels will follow the order of the items in the data frame. If a selection of items is plotted with items, e.g., items = c(24,34,25), then the item labels will be displayed following the order of the vector in items left to right. In this case, item label 1 will be printed on item 24, item label 2 on item 34, and item label 3 on item 25, and so on.
constructs	Logical, if construct vector arrows should be plotted. The default is constructs = FALSE.
construct.lab	Optional. String vector of names for constructs, similar to item.lab.
adjust.lab	Vector of parameters for the position of the item and construct labels for the text3d function. The first value is for horizontal adjustment and the second is for vertical adjustment. The default is adjust.lab = c(0.5, -0.8).
x.lab	Labels for x-axis, the default is x.lab = "X".
y.lab	Labels for y-axis, the default is y.lab = "Y".
z.lab	Labels for y-axis, the default is z.lab = "Z".
title	The main title for the graphical device, plotted with the title3d() function. The default is no title.
line	Title placement for title3d(). The default is line = -5.
axis.scalar	Scalar multiple for adjusting the length of all axes (x, y, z) in the 3D model proportionally. The default is axis.scalar = 1.1.
axis.length	Optional. For adjusting the length of the axis manually by entering a numeric or a numeric vector. For instance, c(3,2,4,3,3,2) indicate axis coordinates x = 3, -x = 3, y = 4, -y = 3, z = 3, -z = 2. Note, a symmetric model can be created easily by

	adding a single numeric in the `axis.length` argument (e.g., `axis.length = 4`) because the function repeats the last value in the vector to cover all axis points. The default is `axis.length = NULL`.
`axis.col`	Color of axis for the `segment3D()`function, the default is `axis.col = "black"`.
`axis.points`	Color of axis points for the `points3d()` function. The default is `axis.points = "black"`.
`points`	Logical, if axis from `points3d()` should have end points. The default is `points = TRUE`.
`axis.ticks`	Logical, if axis ticks from the `axis3d()` function should be plotted. The default is `axis.ticks = TRUE`.
`nticks`	Number of ticks for `axis3d()`. The function repeats the last numeric value in the vector to cover all axis. The user can, therefore, adjust the number of ticks with one numeric value (e.g., `nticks = 6`) or up to three (e.g., `nticks = c(6,4,8)` corresponding to the for the x, y, and z axes respectively. The default is `nticks = 4`.
`width.rgl.x`	Width in the x direction for `par3d()`. The default is `width.rgl.x = 1040`.
`width.rgl.y`	Width in the y direction for `par3d()`. The default is `width.rgl.y = 1040`.
`view`	Vector with polar coordinates and zoom factor for the `view3d` function. The default is `view = c(15,20, 1)`.
`show.plane`	Logical, if xz-plane should be visible in the graphical device. The default is `show.plane = TRUE`.
`plane.col`	Color of the plane, the default is `plane.col = "grey80"`.
`background`	Set background color for the graphical device, the default is `background = "white"`.
`type`	Type of vector arrow for items, the default is `type = "rotation"`. See rgl::arrow3d for more options regarding arrow types.
`col`	Vector of colors representing difficulty levels for item response functions used in `arrow3d()`. The default is `col = c("black", "grey20", "grey40", "grey60", "grey80")`.
`arrow.width`	Width of vector arrows for `arrow3d()`. The default is `arrow.width = 0.6`.
`n`	Number of barbs for the vector arrows from `arrow3d()`. The default is `n = 20`.
`theta`	Opening angle of barbs for vector arrows from `arrow3d()`. The default is `theta = 0.2`.
`barblen`	The length of the barbs for vector arrows from `arrow3d()`. The default is `barblen = 0.03`.
`c.scalars`	Set of scalars for adjusting construct arrow length proportionally. The first numeric adjusts the length in the positive direction and the second numeric the length in the negative direction. The default is `c.scalars = c(1,1)`.
`c.type`	Type of vector arrow for constructs. See rgl::arrow3d for more options regarding arrow types. The default is `c.type = "rotation"`.
`c.col`	Color of construct vector arrows from `arrow3d()`, the default is `c.col = "black"`.
`c.arrow.width`	Width of construct vector arrows for `arrow3d()`. The default is `c.arrow.width = 0.6`.

c.n	Number of barbs for the construct vector arrows from the arrow3d() function. The default is c.n = 20.
c.theta	Opening angle of barbs for construct vector arrows from arrow3d(). The default is c.theta = 0.2.
c.barblen	The length of the barbs for construct vector arrows from arrow3d(). The default is c.barblen = 0.03.
profiles	Data frame with coordinates for spheres representing respondent scores. The default is profiles = NULL.
levels	Optional. A column with values indicating levels for sphere colors from the sphere.col vector. The default is levels = NULL.
sphere.col	Color vector for spheres3d(). The default is sphere.col = c("black", "grey20", "grey40", "grey60", "grey80").
spheres.r	Radius of the spheres for spheres3d(). The default is spheres.r = 0.05.
ci	Logical, if spheres should include an ellipsoid outlining a confidence region returned from the ellipse3d() function. The default is ci = FALSE.
ci.level	Level of confidence for ellipse3d(), the default is ci.level = 0.95.
ellipse.col	Color of the ellipse from ellipse3d(). The default is ellipse.col = "grey80".
ellipse.alpha	Opacity for the confidence region from ellipse3d(). The default is ellipse.alpha = 0.20.
...	Additional arguments passed to RGL or methods.

Details

The plotting function allows plotting of all items, a selection of items as well as plotting a single item. Length of the vector arrows can be set to one unit length across all item vector arrows by setting scale = TRUE. This removes the visualization of the MDISC parameter. Note, when scaling items with scale = TRUE, the plot() function does not change the length of the model axes. This often means that the axes of the model may need to be adjusted, which can be achieved proportionally with axis.scalar or manually with axis.length.

The user also has the option of adding constructs to the graphical output with constructs = TRUE (see the documentation for D3mirt or the package vignette regarding constructs). Other options include plotting one level of difficulty at a time with the diff.level argument if polytomous items are used in the model. Item row names are displayed by default, but the user has the option of adding new item labels for the items with item.lab, as well as labeling constructs with construct.lab.

Regarding the interpretation of results, the angle of the vector arrows indicates what traits, located along the orthogonal axes, an item can be said to describe (Reckase, 2009, 1985, Reckase & McKinley, 1991). For instance, an item located at 0° seen from the x-axis, and 90° as seen from the y and z-axis, only describes trait x. Such an item is unidimensional since its direction vector lies parallel and on the x-axis. In contrast, an item located at 45° between all three axes in a three-dimensional model describes all three traits in the model equally well. Such an item is within-multidimensional with respect to all three latent traits used in the analysis because its direction vector points in a neutral direction in the model.

When plotting the D3mirt model with plot(), it is possible to visually observe statistical violations in the graphical output returned. For instance, shorter vector arrows indicate weaker discrimination

and therefore also higher amounts of statistical violations. Moreover, if a polytomous item struggles or even fail to describe any of the latent variables in the model, it can often lead to an extreme stretch of the MDIFF range. This is comparable to trace lines turning horizontal in a unidimensional item response theory model.

The plot function can also display respondent scores in the three-dimensional model space, represented as spheres whose coordinates are derived from the respondent's factor scores. This allows for a profile analysis in which respondent rows are separated or selected conditioned on some external criteria. To do this, the user must first extract respondent factor scores with mirt::fscores(Chalmers, 2012) and then use some selection process to separate or subset respondent rows. The resulting data frame is used in the `profiles` argument. If desired, a confidence interval can be added to the spheres by setting `ci = TRUE`. A general advice is also to hide vector arrows with `hide = TRUE` when analyzing respondent profiles to avoid visual cluttering. For more on profile analysis (e.g., preparation and examples), see package vignette.

The returned RGL device can, for example, be exported to the R console and be saved as an interactive html file or as a still shoot (see examples below). In the case of the latter, the model perspective in the still shoot can be manually adjusted by changing the `view` argument for the function.

Value

A RGL graphical device.

Author(s)

Erik Forsberg

References

Adler, D., & Murdoch, D. (2022). *Rgl: 3d Visualization Using OpenGL* Computer software.

Chalmers, R., P. (2012). mirt: A Multidimensional Item Response Theory Package for the R Environment. *Journal of Statistical Software, 48*(6), 1-29.

Reckase, M. D. (2009). *Multidimensional Item Response Theory*. Springer.

Reckase, M. D. (1985). The Difficulty of Test Items That Measure More Than One Ability. *Applied Psychological Measurement, 9*(4), 401-412. https://doi-org.ezp.sub.su.se/10.1177/014662168500900409

Reckase, M. D., & McKinley, R. L. (1991). The Discriminating Power of Items That Measure More Than One Dimension. *Applied Psychological Measurement, 15*(4), 361-373. https://doi-org.ezp.sub.su.se/10.1177/014662169101500407

Examples

```
# To plot, the data must be prepared with mirt::mirt and the D3mirt() function
# Load data
data("anes0809offwaves")
x <- anes0809offwaves
x <- x[,3:22] # Remove columns for age and gender

# Fit a three-dimensional graded response model with orthogonal factors
spec <- '  F1 = 1-20
```

```
            F2 = 1-20
            F3 = 1-20

            START=(W7Q3,a2,0)
            START=(W7Q3,a3,0)

            START=(W7Q20,a3,0)

            FIXED=(W7Q3,a2)
            FIXED=(W7Q3,a3)

            FIXED=(W7Q20,a3) '


mod1 <- mirt::mirt(x,
                   spec,
                   itemtype = 'graded',
                   SE = TRUE,
                   method = 'QMCEM')

# Optional: Load the mod1 data for this example directly from the package file
# load(system.file("extdata/mod1.Rdata", package = "D3mirt"))

# Call D3mirt() with mod1 and constructs assigned to c
c <- list(list(1,2,3,4,5,6,7,8,9,10),
          list(11,12,13,14),
          list(15,17,18,19,20))
g <- D3mirt(mod1, c)
plot(g)

# Plot RGL device with constructs visible and named
plot(g, constructs = TRUE,
     construct.lab = c("Compassion", "Fairness", "Conformity"))

# Item W7Q16 has location 6 in the data set (gender and age excluded)
# The item is plotted together with construct to aid the visual interpretation
plot(g, constructs = TRUE,
     items = 6,
     construct.lab = c("Compassion", "Fairness", "Conformity"))

# Plot RGL device on item difficulty level 5
plot(g, diff.level = 5)

# A selection of Conformity items from the model plotted with constructs
plot(g, constructs = TRUE,
     items = c(5,7,8,9,10),
     construct.lab = c("Compassion", "Fairness", "Conformity"))

# Plot RGL device with scaled items and constructs visible and named
plot(g, scale = TRUE,
     constructs = TRUE,
     construct.lab = c("Compassion", "Fairness", "Conformity"))
```

```
# Profile Analysis
# Extract respondent factor scores from mod1 (see D3mirt()) with fscores()
f <- mirt::fscores(mod1,
                   method="EAP",
                   full.scores = TRUE,
                   full.scores.SE = FALSE, QMC = TRUE)

# Optional: Load the respondent factor scores for this example directly from the package file
# load(system.file("extdata/fscores.Rdata", package = "D3mirt"))

# Attach f to the gender variable (column 2 from anes0809offwaves data set; "W3XGENDER")
# Use cbind with fscores() output attached first
x <- anes0809offwaves
z <- data.frame(cbind(f, x[,2]))

# Plot profiles with item vector arrows hidden
# Score levels: 1 = Blue ("male") and 2 = Red ("female")
plot(g, hide = TRUE,
     profiles = z,
     levels = z[,4],
     sphere.col = c("blue", "red"),
     x.lab = "Compassion",
     y.lab="Conformity",
     z.lab="Fairness")

# Add a 95% CI to respondent factor scores on <= 30 y.o.
# Column bind fscores() with age variable ("W3Xage")
y <- data.frame(cbind(f, x[,1]))

# Subset data frame y conditioned on age <= 30
z1 <- subset(y, y[,4] <= 30)

# Use rep() to create a color vector to color groups based on the nlevels() output
# z1 has 14 factor levels
colvec <- c(rep("red", 14))

# Call plot() with profile data on age with item vector arrows hidden
plot(g, hide = TRUE,
     profiles = z1,
     levels = z1[,4],
     sphere.col = colvec,
     x.lab = "Compassion",
     y.lab="Conformity",
     z.lab="Fairness",
     ci = TRUE,
     ci.level = 0.95,
     ellipse.col = "orange")

## Not run:
# Export an open RGL device to the console to be saved as html or image file
plot(g, constructs = TRUE)
s <- scene3d()
rgl::rglwidget(s,
```

```
                width = 1040,
                height = 1040)

# Export a snap shoot of an open RGL device directly to file
plot(g, constructs = TRUE)
rgl::rgl.snapshot('RGLdevice.png',
                  fmt = 'png')

## End(Not run)
```

`print.D3mirt` *Print Method for S3 Objects of Class* `D3mirt`

Description

The print method for the `D3mirt()` function.

Usage

```
## S3 method for class 'D3mirt'
print(x, ...)
```

Arguments

`x`	A S3 object of class `D3mirt`.
`...`	Additional arguments.

Value

A printed message reporting the number of items, levels of difficulty, the number of construct vectors and the row names of the respective items contained in each construct.

Author(s)

Erik Forsberg

Examples

```
## Not run:
# Call D3mirt()
# The first argument can be data frame with model parameters
# or an S4 object of class 'SingleGroupClass' exported from mirt::mirt
g <- D3mirt(mod1)

# Print model summary
print(g)

## End(Not run)
```

print.modid *Print Method for S3 Objects of Class* modid

Description

The print method for the modid() function.

Usage

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

Arguments

x	A S3 object of class modid.
...	Additional arguments.

Value

A printed message reporting the number of factors and the suggested model identification items.

Examples

```
## Not run:
# Identify the DMIRT model using a three-factor EFA with modid()
# x can be a data frame with item data or item factor loadings.
# In the case of the latter, set argument 'efa' to 'FALSE'
g <- modid(x)

# Print model identification summary
print(g)

## End(Not run)
```

summary.D3mirt *Summary Method for S3 Objects of Class* D3mirt

Description

The summary method for the D3mirt() function.

Usage

```
## S3 method for class 'D3mirt'
summary(object, ..., digits = 4)
```

Arguments

`object`	A S3 object of class `D3mirt`.
`...`	Additional arguments.
`digits`	The number of digits shown per estimate. The default is `digits = 4`.

Value

Tables containing *a* and *d* parameters, multidimensional discrimination (MDISC), multidimensional item difficulty (MDIFF), direction cosines, and degrees for vector angles for items. If constructs were used in the estimation process, the summary function will also show tables for direction cosines, and degrees for construct vectors as well as directional discrimination (DDISC) parameters.

Author(s)

Erik Forsberg

Examples

```
## Not run:
# Call D3mirt() and create list of constructs
# The first argument can be data frame with model parameters
# or an S4 object of class 'SingleGroupClass' exported from mirt::mirt
c <- list(list(1,2,3,4,5,6,7,8,9,10),
          list(11,12,13,14),
          list(15,17,18,19,20))
g <- D3mirt(mod1, c)

# Call to summary
summary(g)

#' # Call to summary rounded off to 2 digits
summary(g, digits = 2)

## End(Not run)
```

`summary.modid` *Summary Method for S3 Objects of Class* `modid`

Description

The summary method for the `modid()` function.

Usage

```
## S3 method for class 'modid'
summary(object, ..., digits = 4)
```

Arguments

object	A S3 object of class modid.
...	Additional arguments.
digits	The number of digits shown per estimate. The default is digits = 4.

Value

Model identification items (one less than the number of factors), factor loadings and absolute sum score for model identification items, squared factor loadings and factor loadings for all items.

Author(s)

Erik Forsberg

Examples

```
## Not run:
# Identify the DMIRT model using a three-factor EFA with modid()
# x can be a data frame with item data or item factor loadings.
# In the case of the latter, set argument 'efa' to 'FALSE'
g <- modid(x)

# Call to summary
summary(x)

# Call to summary rounded off to 2 digits
summary(x, digits = 2)

## End(Not run)
```

Index