

# Package: D3mirt (via r-universe)

June 12, 2026

**Title** Descriptive 3D Multidimensional Item Response Theory Modeling

**Version** 2.0.5.9000

**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 primarily exploratory and centered on the plot function, which exposes item characteristics and constructs, represented by vector arrows, within a three-dimensional interactive latent space. The results can be useful for item-level analysis as well as test development.

**License** GPL (>= 3)

**URL** <https://github.com/ForsbergPsychometrics/D3mirt>

**BugReports** <https://github.com/ForsbergPsychometrics/D3mirt/issues>

**Depends** R (>= 3.6.0)

**Imports** mirt, rgl (>= 1.3.1)

**Suggests** knitr, rmarkdown, stats, testthat, R.rsp

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**LazyData** true

**VignetteBuilder** R.rsp

**Config/testthat/edition** 3

**Config/pak/sysreqs** cmake libfreetype6-dev libglu1-mesa-dev make texlive libpng-dev libuv1-dev libgl1-mesa-dev libx11-dev zlib1g-dev

**Repository** <https://forsbergpsychometrics.r-universe.dev>

**Date/Publication** 2026-06-07 10:59:38 UTC

**RemoteUrl** <https://github.com/forsbergpsychometrics/d3mirt>

**RemoteRef** HEAD

**RemoteSha** ec77bd967b42a7be3ba8ceb12124b5b2657a63b2

## Contents

anes0809offwaves . . . . .	2
angles . . . . .	3
D3mirt . . . . .	4
modid . . . . .	7
plot.D3mirt . . . . .	10
print.D3mirt . . . . .	17
print.modid . . . . .	18
summary.D3mirt . . . . .	19
summary.modid . . . . .	20
<b>Index</b>	<b>22</b>

---

anes0809offwaves	<i>Moral Items from The anes0809offwaves data set</i>
------------------	---

---

### 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)
```

---

angles

*Standard Angles Data Frame*

---

### Description

A test unit data frame consisting of 42 rows and 6 columns with standard angles in Cartesian coordinates as item loadings (columns denoted a1, a2, and a3) oriented in both positive and negative directions in a three-dimensional space. The distance from the origin is set by  $d = 0, 5$  (4th column) on all rows, which refers to the parameter related to difficulty in the compensatory model. The last two columns contain the angles converted to spherical coordinates with Theta representing the polar angle and Phi representing the azimuthal angle. Running the data frame in `D3mirt()` converts the angles into spherical coordinates and can be used to check functionality in the package. Note, Nan, i.e., "not-a-number", appears in the `D3mirt()` output because the arctan function (used when changing to spherical coordinates) is not defined when cosine equals zero.

**Usage**

```
angles
```

**Format**

An object of class `data.frame` with 42 rows and 6 columns.

**Examples**

```
data(angles)
```

---

D3mirt

*3D DMIRT Model Estimation*


---

**Description**

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

**Usage**

```
D3mirt(
  x,
  modid = NULL,
  model = NULL,
  con.items = NULL,
  con.sphe = NULL,
  itemtype = "graded",
  method.mirt = "QMCEM",
  method.fscores = "EAP",
  QMC = TRUE
)
```

**Arguments**

<code>x</code>	A data frame with items in rows and model parameters in columns containing raw response data as integer values or factor loadings. Input can also be an S4 object of class 'SingleGroupClass' exported from <a href="#">mirt::mirt</a> (Chalmers, 2012). Regarding the data frame, the number of columns must be more than or equal to 4, i.e., three columns with ( <i>a</i> ) parameters and at least one column for difficulty ( <i>d</i> ) parameters.
<code>modid</code>	Use either the two model identification items from <a href="#">modid</a> as a combined vector or use nested list of item indicators to fit an orthogonal model (see examples below). The default is <code>modid = NULL</code> .
<code>model</code>	The user has the option of imputing a model specification schema used in the call to <a href="#">mirt::mirt</a> (Chalmers, 2012). The default is <code>model = NULL</code> .

<code>con.items</code>	Optional. Nested lists with integer values as item indicators to identify constructs. The default is <code>con.items = NULL</code> .
<code>con.sphe</code>	Optional. Nested lists of spherical angles to identify constructs. The default is <code>con.sphe = NULL</code> .
<code>itemtype</code>	What item type to use in the function call. Available options are "2PL" and "graded". The default is <code>itemtype = "graded"</code> .
<code>method.mirt</code>	Estimation algorithm for <code>mirt::mirt</code> (Chalmers, 2012) to fit the model. The default is <code>method.mirt = "QMCEM"</code> .
<code>method.fscores</code>	Factor estimation algorithm for <code>mirt::fscores</code> (Chalmers, 2012) for extracting respondent trait scores. The default is <code>method.fscores = "EAP"</code> .
QMC	Integration method for <code>mirt::fscores</code> (Chalmers, 2012). The default is <code>QMC = TRUE</code> .

## 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 with item data, or a data frame with factor loadings or an S4 object of class 'SingleGroupClass' exported from `mirt::mirt` (Chalmers, 2012) function fitted in accordance with descriptive item response theory model specifications (see package vignette). 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 function also returns respondent trait scores that can be plotted with `plot` as spheres located in the three-dimensional latent space. In turn, this allows for studying respondent profiles using the `plot` function (for more on profiles, see function documentation on `plot`).

There are two types of models available for D3mirt analysis. The default model is the basic DMIRT model (Reckase, 2009, 1985, Reckase & McKinley, 1991), based on the MDISC, that relaxes the assumption of unidimensionality in the items while restricting the latent space to be orthogonal. To use the default option requires first selecting two items to identify the model. This can be done manually with the `modid` argument in the function call to `D3mirt`. However, it is advisable to use the dedicated function `modid` included in the package for this purpose (for more on model identification see function documentation for `modid`). In contrast, the optional orthogonal model constrains the items to be strictly parallel with the axes (see example section below). Consequently, this option allows the user to investigate the model under the assumption that the items are strictly unidimensional and orthogonally oriented in the latent space. In this context "orthogonal" refers to the perpendicular orientation of the item vectors the model specification creates. Note that using the optional model will result in reporting unidimensional discrimination parameters (DISC) and the shift in item orientation will affect respondent locations in the latent space. It is also possible to specify a unique model with the help of the `model` argument in the function call to `D3mirt` if written in `mirt` (Chalmers, 2012) syntax (for an example, see the appendix in the package vignette).

The user also has the option of including constructs in the estimation. Constructs, in this context, refer to the assumption that a subset of items or a particular angle in the latent space holds some higher-order latent variable of interest. 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 the items are unidimensional, measuring the same latent variable indicated by the angle of the construct vector.

To include constructs, the user can create one or more nested lists that indicate what items belong to what construct (from one item 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 can also be indicated using spherical coordinates stored in nested lists. This allows the user to freely add any number of constructs at any angle in the latent space to study the item discrimination.

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, vector coordinates, and respondent trait scores.

### 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.  
<https://doi.org/10.18637/jss.v048.i06>
- 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/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/10.1177/014662169101500407>

### Examples

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

# Call to D3mirt(), including optional nested lists for three constructs
# Item W7Q16 is not included in any construct because of model violations
# Constructs can also be defined using interval notation, i.e., c(1:10) and so on.

con <- list(c(1,2,3,4,5,6,7,8,9,10),
           c(11,12,13,14),
           c(15,17,18,19,20))
mod <- D3mirt(x, modid = c("W7Q3", "W7Q20"), con.items = con)
```

```

# Show summary of results
summary(mod)

# Call to D3mirt(), including optional constructs with the help of spherical coordinates
# Spherical coordinates are indicated using nested list structures with angles
# First angle indicates the rotation in the xz-plane
# The second angle is the angle away from the y-axis.
# The specification below indicates three constructs located at a 45-degree angle
# between the three axes in the positive orientation.
# It is possible to assign factor loadings and difficulty parameters from mod to a new data frame
# This skips fitting the compensatory model and makes fitting the model with D3mirt() instant
# Note that trait scores will not be included in the exported S3 object when using this option
y <- cbind(mod$loadings, mod$diff)
con <- list(c(0, 45),
           c(45, 90),
           c(90, 45))
mod <- D3mirt(y, con.sphe = con)

# Show summary of results
summary(mod)

# Call D3mirt() using the orthogonal optional model
# often requires removing items with poor fit
# In this example item W7Q16 is removed from the data frame
y <- data.frame(x[,-16])

# Items are constrained to the x, y, and z-axes using
# nested lists with positive integers as item indicators
# Note that integers indicate where the items are located in the data frame
mod <- D3mirt(y, modid = list(c(1:10),
                             c(15:19),
                             c(11:14)))

# Show summary of results
summary(mod)

```

---

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 <code>efa = TRUE</code> .
factors	The number of factors for the exploratory factor analysis. The default is <code>factors = 3</code> .
lower	The lower bound for the item pool calculated using the standard deviation of scaled item factor loadings. The default is <code>lower = 0.5</code> .
upper	The upper bound for filtering absolute sum scores less than or equal to the indicated value. The default is <code>upper = .10</code> .
fac.order	Optional. Users can override the automatic sorting of factors by manually indicating factor order with integer values, e.g., <code>c(2, 1, 3)</code> to start with the second factor (or column) in data frame <code>x</code> , followed by the first factor (or column) in <code>x</code> , and then lastly the third factor (or column). The default is <code>fac.order = NULL</code> .
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 <code>itemtype = "graded"</code> . See <a href="#">mirt::mirt</a> (Chalmers, 2012) for more on item models.
method	A string indicating what integration algorithm to use for the EFA. The default is <code>method = 'EM'</code> . See <a href="#">mirt::mirt</a> (Chalmers, 2012) for more on methods.
rotate	A string indicating what rotation method to use for the EFA. The default is <code>rotate = "oblimin"</code> . See <a href="#">mirt::mirt</a> (Chalmers, 2012) for more on rotations.
...	Any additional arguments passed to <code>mirt()</code> .

### Details

Before performing DMIRT analysis, it is necessary to identify the compensatory model (Reckase, 2009). For a three-dimensional model, this implies 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 achieved, it is possible to create a three-dimensional DMIRT model that reflects the data correctly.

The `modid()` function can help by suggesting what items to use for the latter purpose. The function does this by first performing an EFA on the data and then selecting the strongest loading items,

following the order of strength of the factors and following 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 that 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 multi-dimensional 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 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 data frame with factor loadings 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 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.  
<https://doi.org/10.18637/jss.v048.i06>
- 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
id <- modid(x)

# Call to summary
summary(id)

# Call to modid with increased lower and upper bound
# Assign loadings to a data frame and set efa to false
x <- id$loadings
id <- modid(x, efa = FALSE, lower = 1, upper = 1 )
summary(id)

# Override factor order by reversing columns in the original data frame
id <- modid(x, efa = FALSE, fac.order = c(3,2,1))
summary(id)

```

---

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,
  console = TRUE,
  scale = TRUE,
  hide = FALSE,
  ind.scores = FALSE,
  diff.level = NULL,
  items = NULL,
  item.names = TRUE,
  item.lab = NULL,
  constructs = FALSE,
  construct.lab = NULL,
  adjust.lab = c(0.5, -0.8),
  dist = 0,

```

```

c.dist = 0,
x.lab = "X",
y.lab = "Y",
z.lab = "Z",
title = "",
line = -5,
font = 1,
cex = 0.8,
font.col = "black",
axis.scalar = 1.1,
axis.length = NULL,
axis.points = "black",
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 <code>D3mirt</code> .
console	Logical, if RGL object should be printed to console (TRUE) or to external pop-

	up window. The default is <code>console = TRUE</code> .
<code>scale</code>	Logical, if item vector arrow length should visualize the MDISC. If set to <code>FALSE</code> , the vector arrow length will be of one unit length. The default is <code>scale = TRUE</code> .
<code>hide</code>	Logical, if items should be plotted. The default is <code>hide = FALSE</code> .
<code>ind.scores</code>	Logical, should output plot respondents trait scores. The default is <code>ind.scores = FALSE</code> .
<code>diff.level</code>	Optional. Plotting of a single level of difficulty indicated by an integer.
<code>items</code>	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.
<code>item.names</code>	Logical, if item labels should be plotted. The default is <code>item.names = TRUE</code> .
<code>item.lab</code>	Optional. String vector of item names that will override row names extracted from the data frame. Note that row names are not overwritten. Instead, the string vector in <code>item.lab</code> 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 <code>items</code> , e.g., <code>items = c(24, 34, 25)</code> , then the item labels will be displayed following the order of the vector in <code>items</code> 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.
<code>constructs</code>	Logical, if construct vector arrows should be plotted. The default is <code>constructs = FALSE</code> .
<code>construct.lab</code>	Optional. String vector of names for constructs, similar to <code>item.lab</code> .
<code>adjust.lab</code>	Vector of parameters for the position of the item and construct labels for the <code>text3d</code> function. The first value is for horizontal adjustment, and the second is for vertical adjustment. The default is <code>adjust.lab = c(0.5, -0.8)</code> .
<code>dist</code>	Distance (+/-) of item vector label from item vector. The default is <code>dist = 0</code> .
<code>c.dist</code>	Distance (+/-) of construct vector label from construct vector. The default is <code>c.dist = 0</code> .
<code>x.lab</code>	Labels for x-axis, the default is <code>x.lab = "X"</code> .
<code>y.lab</code>	Labels for y-axis, the default is <code>y.lab = "Y"</code> .
<code>z.lab</code>	Labels for z-axis, the default is <code>z.lab = "Z"</code> .
<code>title</code>	The main title for the graphical device, plotted with the <code>title3d()</code> function. The default is no title.
<code>line</code>	Title placement for <code>title3d()</code> . The default is <code>line = -5</code> .
<code>font</code>	A numeric font number from 1 to 5, the default is <code>font = 1</code> . See <a href="#">rgl::text3d</a> for more on font options.
<code>cex</code>	A numeric character expansion value to adjust font size, the default is <code>cex = 0.8</code> .
<code>font.col</code>	Color of axes, numbers, and fonts. The default is <code>font.col = "black"</code> .
<code>axis.scalar</code>	Scalar multiple for adjusting the length of all axes (x, y, z) in the 3D model proportionally. The default is <code>axis.scalar = 1.1</code> .

axis.length	Optional. For adjusting the length of the axis manually by entering a numeric or a numeric vector. For instance, <code>c(3,2,4,3,3,2)</code> indicate axis coordinates $x = 3$ , $-x = 3$ , $y = 4$ , $-y = 3$ , $z = 3$ , $-z = 2$ . Note that a symmetric model can be created easily by adding a single numeric in the <code>axis.length</code> argument (e.g., <code>axis.length = 4</code> ) because the function repeats the last value in the vector to cover all axes points. The default is <code>axis.length = NULL</code> .
axis.points	Color of axis points for the <code>points3d()</code> function. The default is <code>axis.points = "black"</code> .
axis.ticks	Logical, if axis ticks from the <code>axis3d()</code> function should be plotted. The default is <code>axis.ticks = TRUE</code> .
nticks	Number of ticks for <code>axis3d()</code> . 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., <code>nticks = 6</code> ) or up to three (e.g., <code>nticks = c(6,4,8)</code> ) corresponding to the for the x, y, and z axes respectively. The default is <code>nticks = 4</code> .
width.rgl.x	Width in the x direction for <code>par3d()</code> . The default is <code>width.rgl.x = 1040</code> .
width.rgl.y	Width in the y direction for <code>par3d()</code> . The default is <code>width.rgl.y = 1040</code> .
view	Vector with polar coordinates and zoom factor for the <code>view3d</code> function. The default is <code>view = c(15,20, 1)</code> .
show.plane	Logical, if xz-plane should be visible in the graphical device. The default is <code>show.plane = TRUE</code> .
plane.col	Color of the plane, the default is <code>plane.col = "grey80"</code> .
background	Set background color for the graphical device, the default is <code>background = "white"</code> .
type	Type of vector arrow for items, the default is <code>type = "rotation"</code> . See <a href="#">rgl::arrow3d</a> for more options regarding arrow types.
col	Vector of colors representing difficulty levels for item response functions used in <code>arrow3d()</code> . The default is <code>col = c("black", "grey20", "grey40", "grey60", "grey80")</code> .
arrow.width	Width of vector arrows for <code>arrow3d()</code> . The default is <code>arrow.width = 0.6</code> .
n	Number of barbs for the vector arrows from <code>arrow3d()</code> . The default is <code>n = 20</code> .
theta	Opening angle of barbs for vector arrows from <code>arrow3d()</code> . The default is <code>theta = 0.2</code> .
barblen	The length of the barbs for vector arrows from <code>arrow3d()</code> . The default is <code>barblen = 0.03</code> .
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 <code>c.scalars = c(1,1)</code> .
c.type	Type of vector arrow for constructs. See <a href="#">rgl::arrow3d</a> for more options regarding arrow types. The default is <code>c.type = "rotation"</code> .
c.col	Color of construct vector arrows from <code>arrow3d()</code> , the default is <code>c.col = "black"</code> .
c.arrow.width	Width of construct vector arrows for <code>arrow3d()</code> . The default is <code>c.arrow.width = 0.6</code> .

<code>c.n</code>	Number of barbs for the construct vector arrows from the <code>arrow3d()</code> function. The default is <code>c.n = 20</code> .
<code>c.theta</code>	Opening angle of barbs for construct vector arrows from <code>arrow3d()</code> . The default is <code>c.theta = 0.2</code> .
<code>c.barblen</code>	The length of the barbs for construct vector arrows from <code>arrow3d()</code> . The default is <code>c.barblen = 0.03</code> .
<code>profiles</code>	Data frame with coordinates for spheres representing a subset of respondent scores. The default is <code>profiles = NULL</code> .
<code>levels</code>	Optional. A column with values indicating levels for sphere colors from the <code>sphere.col</code> vector. The default is <code>levels = NULL</code> .
<code>sphere.col</code>	Color vector for <code>spheres3d()</code> . The default is <code>sphere.col = c("black", "grey20", "grey40", "grey60", "grey80")</code> .
<code>spheres.r</code>	Radius of the spheres for <code>spheres3d()</code> . The default is <code>spheres.r = 0.05</code> .
<code>ci</code>	Logical, if spheres should include an ellipsoid outlining a confidence region returned from the <code>ellipse3d()</code> function. The default is <code>ci = FALSE</code> .
<code>ci.level</code>	Level of confidence for <code>ellipse3d()</code> , the default is <code>ci.level = 0.95</code> .
<code>ellipse.col</code>	Color of the ellipse from <code>ellipse3d()</code> . The default is <code>ellipse.col = "grey80"</code> .
<code>ellipse.alpha</code>	Opacity for the confidence region from <code>ellipse3d()</code> . The default is <code>ellipse.alpha = 0.20</code> .
<code>...</code>	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 = FALSE`. This removes the visualization of the MDISC parameter. Note that when items are not scaled, 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 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 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 degrees seen from the x-axis, and 90 degrees 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 degrees 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, higher amounts of statistical violations. Moreover, if a polytomous item struggles or even fails 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::fcores` (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 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 saved as an interactive HTML file or as a still shoot (see examples below). In the the latter case, 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.  
<https://doi.org/10.18637/jss.v048.i06>
- 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/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/10.1177/014662169101500407>

### Examples

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

# Call D3mirt() with constructs assigned to con
con <- list(c(1,2,3,4,5,6,7,8,9,10),
```

```

        c(11,12,13,14),
        c(15,17,18,19,20))
mod <- D3mirt(x, modid = c("W7Q3", "W7Q20"), con.items = con)

# Plot RGL device without constructs
plot(mod)

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

plot(mod, constructs = TRUE,
      items = c(15, 17, 18, 19, 20),
      construct.lab = c("Compassion", "Fairness", "Conformity"))

# Item W7Q16 has location 16 in the data set (gender and age excluded)
# Below, the item is plotted together with construct to aid the visual interpretation
plot(mod, constructs = TRUE,
      items = 16,
      construct.lab = c("Compassion", "Fairness", "Conformity"))

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

# Plot RGL device with item vectors at unit length and constructs visible and named
plot(mod, scale = TRUE,
      constructs = TRUE,
      construct.lab = c("Compassion", "Fairness", "Conformity"))

# Profile Analysis
# Plot respondents trait scores with item vectors hidden and no constructs
plot(mod, hide = TRUE, ind.scores = TRUE)

# Plot respondents separated on gender
# Start by assigning the gender variable to a data frame
# In this example, this is done by sub-setting the gender column
x <- anes0809offwaves

# Call plot() and use the gender variable column in the levels argument
# Respondent data on gender is in column two, x[, 2]
# In the function call below, both items and constructs are hidden
# Score levels: 1 = Blue ("male") and 2 = Red ("female")
plot(mod, hide = TRUE, ind.scores = TRUE,
      levels = x[, 2],
      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 trait scores with the age variable "W3Xage"
z <- data.frame(cbind(mod$fscores, x[, 1]))

```

```
# Subset data frame z conditioned on age <= 30
z1 <- subset(z, z[, 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
# Use the profiles argument for the data frame containing the subset to be plotted
plot(mod, 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:
# Plot as interactive RGL device in external window
plot(mod, console = FALSE)

# Export an open RGL device to the console to be saved as HTML or image file
plot(mod)
s <- rgl::scene3d()
rgl::rglwidget(s,
               width = 1040,
               height = 1040)

# Export a snapshot of an open RGL device directly to file
plot(mod)
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 names of the respective items contained in each construct.

**Author(s)**

Erik Forsberg

**Examples**

```
## Not run:
# Load data
data("anes0809offwaves")
x <- anes0809offwaves
x <- x[, 3:22] # Remove columns for age and gender

# Call D3mirt()
mod <- D3mirt(x, modid = c("W7Q3", "W7Q20"))

# Print model summary
print(mod)

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

# Identify the DMIRT model
id <- modid(x)

# Print model identification summary
print(id)

## 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

<code>object</code>	A S3 object of class <code>D3mirt</code> .
<code>...</code>	Additional arguments.
<code>digits</code>	The number of digits shown per estimate. The default is <code>digits = 4</code> .

## 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, degrees for construct vectors, and directional discrimination (DDISC) parameters.

## Author(s)

Erik Forsberg

**Examples**

```
## Not run:
# Load data
data("anes0809offwaves")
x <- anes0809offwaves
x <- x[, 3:22] # Remove columns for age and gender

# Call D3mirt() with constructs
con <- list(c(1,2,3,4,5,6,7,8,9,10),
            c(11,12,13,14),
            c(15,17,18,19,20))
mod <- D3mirt(x, modid = c("W7Q3", "W7Q20"), con.items = con)

# Call to summary
summary(mod)

#' # Call to summary rounded off to 2 digits
summary(mod, 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**

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

**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:  
# Load data  
data("anes0809offwaves")  
x <- anes0809offwaves  
x <- x[,3:22] # Remove columns for age and gender  
  
# Identify the DMIRT model  
id <- modid(x)  
  
# Call to summary  
summary(id)  
  
# Call to summary rounded off to 2 digits  
summary(id, digits = 2)  
  
## End(Not run)
```

# Index

## \* datasets

anes0809offwaves, 2  
angles, 3

anes0809offwaves, 2  
angles, 3

D3mirt, 4, 14  
D3mirt(), 10, 17, 19

mirt::fscores, 5, 15  
mirt::mirt, 4, 5, 8, 9  
modid, 4, 5, 7  
modid(), 18, 20

plot, 5  
plot.D3mirt, 10  
print.D3mirt, 17  
print.modid, 18

rgl::arrow3d, 13  
rgl::text3d, 12

summary.D3mirt, 19  
summary.modid, 20