Package ‘Rwinsteps’

January 31, 2019

Version 1.0-1.1

Date 2012-1-30

Title Running Winsteps in R

Author Anthony Albano <`tony.d.albano@gmail.com`>, Ben Babcock <`ben.babcock@arrt.org`>

Maintainer Anthony Albano <`tony.d.albano@gmail.com`>

Depends R (>= 2.12.0)

Description Facilitates communication between R
and the Rasch modeling software Winsteps. Currently
includes functions for reading and writing command files,
sending them to Winsteps, reading and writing data according to
command file specifications, reading output into R, and
plotting various results.

LazyLoad yes

License GPL (>= 2)

URL `https://www.r-project.org`

Repository CRAN

Date/Publication 2019-01-31 17:27:47 UTC

NeedsCompilation no

R topics documented:

Rwinsteps-package *Running Winsteps in R*

Description

The Rwinsteps package facilitates communication between R and the Rasch modeling software Winsteps. The package currently includes functions for reading and writing command files, sending them to Winsteps, reading and writing data according to command file specifications, reading output into R, and plotting various results.

Details

Package:	Rwinsteps
Version:	1.0-1
Date:	2012-1-30
Depends:	R (>= 2.12.0)
LazyLoad:	yes
License:	GPL (>= 2)
URL:	http://www.r-project.org
Built:	R 2.13.0; ; 2011-08-30 21:02:07 UTC; windows

Index:

```
ifile                   Item and Person Summary Files
plot.ifile              Plotting Item Staistics
wdat                    Winsteps Data File
rirf                    Item Functions
wcmd                    Winsteps Command File
winsteps                Running Winsteps
```

Rasch modeling in Winsteps begins with a command file, which specifies the structure and contents of the data file and which includes options for customizing how the model is run and fit and for requesting output tables and files. A command file may be construced in R using the `wcmd` function, read in to R using `read.wcmd`, or simply referenced in the `winsteps` function, in which case it is still read into R but invisibly.

The `winsteps` function returns two of the more relevant output files, the ifile and pfile, for further manipulation and analysis in R. These can also be saved to file and they are read into R using the functions `read.ifile` and `read.pfile`, which are simple wrappers to the function `read.csv`. In this case, files should be saved as csv, and the default file headers should always be included.

Rwinsteps includes additional functions for obtaining the response, information, and error functions at both the test and item levels for the dichotomous Rasch model, and for plotting and comparing results. Finally, the `read.wdat` and `write.wdat` functions read and write fixed width formatted data by extracting required information from a command file.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>, Ben Babcock <ben.babcock@arrt.org>

Maintainer: Anthony Albano <tony.d.albano@gmail.com>

References

Winsteps is commercial software, available at `www.winsteps.com`

`ifile` *Item and Person Summary Files*

Description

An "ifile" is a data frame containing statistical output for a set of items. A "pfile" is a data frame containing statistical output for a group of people, or examinees. These functions create, read, and convert to "ifile" and "pfile" objects.

Usage

```
ifile(measure, entry = 1:length(measure), ...)

pfile(measure, entry = 1:length(measure), ...)

as.ifile(x)

as.pfile(x)

read.ifile(filename, skip = 1, col.names, sep = ",", ...)

read.pfile(filename, skip = 1, col.names, sep = ",", ...)
```

Arguments

`measure`	numeric vector of item locations
`entry`	item position numbers, normally a sequence from 1 to the number of items
`x`	a data frame with items as rows, containing, at a minimum, columns titled "measure" and "entry"
`filename`	path to the file, which is assumed to be a table in 'csv' format
`skip`	number of header lines to skip, defaulting to 1
`col.names`	vector of column names to be added to the file after it is read in
`sep`	the field separator character, sent to `read.table`, defaulting to comma separated values
`...`	For `ifile` and `pfile`, additional variables, as vectors of the same length as `measure`, to be included in the ifile or pfile; for `read.ifile` and `read.pfile`, further arguments passed to `read.table`

Details

The ifile contains item-level information, and the pfile person-level information, based on a fit of the Rasch model to a set of item-response data.

Using `read.ifile` and `read.pfile`, the ifile and pfile are read in as 'csv' files, i.e., tables with `sep = ","`. These functions are simple wrappers for the function `read.csv`.

Value

A data frame of class "ifile" or "pfile"

Author(s)

Anthony Albano `<tony.d.albano@gmail.com>`

See Also

`read.ifile`, `plot.ifile`

Examples

```
imeasure <- rnorm(5)
ifile(imeasure, name = paste("item", 1:5, sep = ""))
as.ifile(data.frame(measure = imeasure, entry = 1:5))

pmeasure <- rnorm(3)
pfile(pmeasure, name = c("Skeeter", "Shira", "Soto"))
as.pfile(data.frame(measure = pmeasure, entry = 1:3))
```

`plot.ifile` *Plotting Item Staistics*

Description

These are Rasch model plot methods for item response functions, item inforamtion functions, item error functions, the test response function, the test information function, and the test error function, where each applies to an object of the corresponding class. `plot.ifile` summarizes the items within an ifile by plotting each of these functions in succession.

Usage

```
## S3 method for class 'ifile'
plot(x, theta = seq(-4, 4, length = 100),
  subset = 1:nrow(x), ...)

## S3 method for class 'rirf'
plot(x, theta = x$theta, add = FALSE,
  xlab = expression(theta), ylab = "P(X)", main = "IRF",
    lwd = 2, col = "r", ...)

## S3 method for class 'riif'
plot(x, theta = x$theta, add = FALSE,
  xlab = expression(theta), ylab = "Information", main = "IIF",
  lwd = 2, col = "r", ...)

## S3 method for class 'rief'
plot(x, theta = x$theta, add = FALSE,
  xlab = expression(theta), ylab = "Standard Error",
  main = "IEF", lwd = 2, col = "r", ...)

## S3 method for class 'rtrf'
plot(x, theta = x$theta, xlab = expression(theta),
  ylab = "Total", main = "TRF", lwd = 2, ...)

## S3 method for class 'rtif'
plot(x, theta = x$theta, xlab = expression(theta),
  ylab = "Information", main = "TIF", lwd = 2, ...)

## S3 method for class 'rtef'
plot(x, theta = x$theta, xlab = expression(theta),
  ylab = "Standard Error", main = "TEF", lwd = 2, ...)
```

Arguments

`x`	for `plot.ifile`, an object of class “ifile”, otherwise, either an ifile or a vector of item locations
`theta`	optional vector of theta values over which the functions will be evaluated
`subset`	an index vector indicating the items to plot, as row numbers or item names if present in the ifile
`xlab`	the x-axis label, as a string, defaulting to `expression(theta)`
`ylab`	the y-axis label, as a string, with default depending on the method
`main`	the plot title, with default depending on the method
`lwd`	line width, as an integer, defaulting to 2
`col`	vector of line colors, where the default `"r"` indicates a selection from the rainbow. Recycled if necessary
`add`	boolean, with default `FALSE`, indicating whether or not to add to the current plot
`...`	Further arguments passed to `plot.default`

Author(s)

Anthony Albano <`tony.d.albano@gmail.com`>

See Also

`ifile`

Examples

```
plot(ifile(-2:2))
```

`rirf` *Item Functions*

Description

These functions return the item response, item inforamtion, item error, test response, the test information, and the test error functions for a set of Rasch item location, i.e., difficulty parameters.

Usage

```
rirf(x, theta = seq(-4, 4, length = 100))

riif(x, theta = seq(-4, 4, length = 100))

rief(x, theta = seq(-4, 4, length = 100))

rtrf(x, theta = seq(-4, 4, length = 100))

rtif(x, theta = seq(-4, 4, length = 100))

rtef(x, theta = seq(-4, 4, length = 100))
```

Arguments

`x`	numeric item location or a vector of item locations, or an ifile
`theta`	optional vector of theta values over which the functions will be evaluated

Details

`rirf` evaluates the Rasch model for each item location over theta, returning the probabilities correct. The remaining functions all incorporate these probablities to return the information (`riif` and standard errors (`rief`) for each item across theta, or aggregates of these values across a set of items (`rtrf` for the test response function, `rtif`for the test information function, and `rtef` for the test error function).

Value

A list is returned containing theta and the corresponding item function.

Author(s)

Anthony Albano `<tony.d.albano@gmail.com>`

See Also

plot.ifile

Examples

```
rirf(-2)
rtrf(-2:2)
```

---

wcmd *Winsteps Command File*

Description

wcmd creates a Winsteps command file object. as.wcmd converts a list to an object of class “wcmd”. read.wcmd and write.wcmd read and write command files to and from text files.

Usage

```
wcmd(title = "R2Winsteps Run", data, item1, ni, name1,
  namelen = item1 - name1, codes = 0:1, csv = "y", hlines = "y",
  tfile = NULL, arglist = NULL, anchor = NULL, labels = NULL,
  extra = NULL)

as.wcmd(x)

read.wcmd(filename)

write.wcmd(cmd, filename)
```

Arguments

title	a title for the Winsteps run, as a string
data	data set filename
item1	integer, fixed width column number of the first item
ni	integer, number of items
name1	integer, fixed width column number of the first character of the person name
namelen	integer, number of characters in person name, defaulting to the difference between `item1` and `name1`
codes	vector of response codes with default `c(0, 1)`
csv	character indicating how output files should be written, as csv (`"y"`, default) or as tab delimited (`"n"`). In order to read in and manipulate files using R2Winsteps, this must be `"y"`, and the Winsteps default which includes headers must not change (`hlines = "y"`)
hlines	character indicating whether or not to include header lines in output files, with default `"y"`, as required for using functions such as `read.ifile` and `read.pfile`

`tfile`	numeric vector of table numbers to be included in output
`arglist`	optional list of additional arguments to be appended to the command file, where the name of each list element will be pasted into a string with the value of the element, as `name = "value"`
`anchor`	optional matrix of item anchor values, with item sequences in column 1 and anchor values in column 2
`labels`	optional string vector of item labels, to be pasted after the `&END` statement in the command file
`extra`	optional vector of additional commands, similar to `arglist`, but where element names are ignored and each element is simply pasted into the command file separated by new lines
`x`	list of arguments to be converted into an object of class "wcmd"
`filename`	path to which the command file will be written, or from which it will be read
`cmd`	command file object to be written

Details

The Winsteps command file specifies the structure of a persons by items matrix and includes all arguments necessary to run the model. Arguments without defaults, which are currently required in the "wcmd" class, are `data`, `item1`, `ni`, and `name1`. Winsteps accepts a variety of arguments and data specification options within the command file and only a selection of these are currently supported by the "wcmd" class, not including those necessary for polytomous models. Thus, for the greatest flexibility, the command file should be written by hand. For details, see the Winsteps manual.

Using `write.wcmd`, the command file is written to `filename` by separating all arguments and item labels with new lines. As all arguments are read and written as text strings, any argument with integer values such as `codes` may also be specified as a string, e.g., `codes = "01"`.

The command file can also be used to facilitate reading and writing of data files in R. See `read.wdat` and `write.wdat` for details.

Value

Returns a command file as a list of arguments

Author(s)

Anthony Albano <`tony.d.albano@gmail.com`>

See Also

`winsteps`

Examples

```
# Create
tempcmd <- tempfile()
cmd <- wcmd(title = "R2Winsteps Example", data = "example.dat",
  item1 = 1, ni = 15, name1 = 16, namelen = 5,
  labels = paste("i", 1:15, sep = ""), hlines = "Y")

# Write and read
write.wcmd(cmd, tempcmd)
cmd2 <- read.wcmd(tempcmd)
```

---

wdat *Winsteps Data File*

Description

These functions read and write Winsteps data files, where formatting is specified in a command file object.

Usage

```
write.wdat(x, cmd, datfile = cmd$data, na = " ")

read.wdat(cmd, datfile = cmd$data, na = " ", ilabels = NULL)
```

Arguments

`x`	data frame of item responses
`cmd`	command file object
`datfile`	path to which the data file will be written, or from which it will be read, defaulting to the path supplied in the command file
`na`	character representing missing data, defaulting to a blank space
`ilabels`	vector of new item labels to be added to the item columns, overriding cmd$labels

Details

When reading data, the person names should occupy 1 column and each item should occupy 1 column in the file. A warning is returned if any rows have fewer than the maximum number of columns and the person names are not in the last column, in which case the missing information pertains to items. `ilabels` will override the item labels in `cmd$labels`, if they exist. If both are `NULL`, generic item names are supplied.

When writing data, all unnamed columns are assumed to be item responses. If the command file contains item labels they will be used to extract and reorder the item responses from `x` and all other columns in `x` will be ignored, except for the person IDs which must have a column name of `"name"`. The data are written to the filename `datfile` in fixed width format according to the `item1`, `ni`, `name1`, and `namelen` command file components.

Value

`read.wdat` returns a data frame

Author(s)

Anthony Albano <`tony.d.albano@gmail.com`>

Examples

```
# Simulate data, with missings
set.seed(82911)
b <- seq(-3, 3, length = 15)
theta <- rnorm(100, 1)
rmat <- ifelse(rirf(b, theta)$p > runif(1500), 1, 0)
rmat[sample(1500, 30)] <- NA
rmat <- data.frame(rmat)

# Item and person labels
colnames(rmat) <- paste("i", 1:15, sep = "")
rmat$name <- paste("p", 1:100, sep = "")

# Create command file
tempdat <- tempfile()
cmd <- wcmd(title = "R2Winsteps Example", data = tempdat,
  item1 = 1, ni = 15, name1 = 16, namelen = 5,
  labels = paste("i", 1:15, sep = ""), hlines = "Y")

# Write to temp file and read back in
write.wdat(rmat, cmd, na = "N")
rmat2 <- read.wdat(cmd, na = "N")
```

`winsteps` *Running Winsteps*

Description

The `winsteps` function sends a command file to the program Winsteps and returns the resulting item and person information, optionally saving them as output files. `as.winsteps` converts its arguments into an object of class “winsteps”.

Usage

```
winsteps(cmd, cmdfile = "cmdfile", outfile = "outfile",
  ifile = "ifile", pfile = "pfile", newdir = getwd(),
  run = TRUE, windir = "winsteps")

as.winsteps(cmd, ifile, pfile, daterun, comptime)
```

Arguments

`cmd`	a command file object, created using `wcmd`
`cmdfile`	a path to the command file, when running `winsteps`, or a list to be converted into a command file object, when running `as.winsteps`
`outfile`	a path to which the standard Winsteps output will be written
`ifile`	a path to the item file, when running `winsteps`, or a list to be converted into an ifile object, when running `as.winsteps`
`pfile`	a path to the person file, when running `winsteps`, or a list to be converted into a pfile object, when running `as.winsteps`
`newdir`	an optional, temporary, directory for reading and writing files, defaulting to the current working directory
`run`	boolean, with default `TRUE`, indicating whether or not Winsteps should be run. If `FALSE`, the Winsteps files cmdfile, ifile, and pfile are read in, assuming they are each specified as strings of length 1.
`windir`	the directory for the Winsteps program, which is sent to the Windows command prompt. If the directory is already included in the Windows PATH variable, the default `"winsteps"` will suffice
`daterun`	date and time the model was run
`comptime`	time required to run the model

Details

Winsteps is commercial software for analyzing item response data, primarily through applications of the Rasch model, a 1 parameter item response theory model. The `winsteps` function interacts with Winsteps in batch mode and retrieves the two main output files produced by the program, referred to as an ifile, which contains item output, and a pfile, which contains person output.

The only argument required by `winsteps` is a command file, which can be supplied directly as `cmd`, a command file object of class “wcmd”, or which can be referenced with the filename `cmdfile`. If both are included, the object is written to the filename.

`cmdfile`, `outfile`, `ifile`, and `pfile` are all used to write the corresponding Winsteps files when `run = TRUE`. Regardless of whether Winsteps is run or not, the `cmdfile`, `ifile`, and `pfile` will be read into R and returned. If the defaults are retained for these arguments the files will be deleted once the function has completed without errors.

Winsteps is called by invoking the operating system using the `system` command. `windir` can be used to specify the Winsteps directory folder (e.g., "c:/program files/winsteps"), should the system have a hard time finding it.

Value

A list object of class “winsteps” containing:

`cmd`	the command file
`ifile`	the item file
`pfile`	the person file
`daterun`	date and time the model was run
`comptime`	time required to run the model

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

References

Winsteps is commercial software, available at www.winsteps.com

See Also

wcmd, ifile, pfile

Examples

```
# Simulate scores for 15 items and 100 people
set.seed(82911)
b <- seq(-3, 3, length = 15)
theta <- rnorm(100, 1)
rmat <- data.frame(ifelse(rirf(b, theta)$p > runif(1500), 1, 0))

# Item and person labels
colnames(rmat) <- paste("i", 1:15, sep = "")
rmat$name <- paste("p", 1:100, sep = "")

# Create a command file object
cmd <- wcmd(title = "R2Winsteps Example", data = "example.dat",
  item1 = 1, ni = 15, name1 = 16, namelen = 5,
  labels = paste("i", 1:15, sep = ""), hlines = "Y")

# The last two steps require access to a local directory and
# are not run

# Write the data to file
# write.wdat(rmat, cmd)

# Run Winsteps, with default filenames, not saving the
# command file or other output to file
# out <- winsteps(cmd)
```

Index