Package 'anchorpoint'

April 12, 2021

Title Anchor Point Selection Based on the Gini Inequality Criterion

Version 0.0.0.9000

Description This package implements a Anchor Point Selection method based on the paper 'Anchor Point Selection – Scale Alignment Based on an Inequality Criterion' by Strobl et al. (2020). It provides data generating processes and graphical decision support functions (criterion path, shifted item plots and a graphical test).

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports grDevices,
stats,
ineq,
graphics,
scales,
psychotools,
multcomp,
Rdpack

RdMacros Rdpack

Suggests psychotree,
qvcalc,
mirt,mvtnorm

R topics documented:

`anchorpoint` *Function to produce anchorpoint objects*

Description

Function to conduct the anchor point selection method of Strobl et al. (2021)

Usage

```
anchorpoint(
  rm1,
  rm2,
  select = c("CLF Criterion", "Gini Index"),
  grid = c("symmetric", "sparse")
)
```

Arguments

`rm1`	Fitted Rasch Model object for the first group of test takers
`rm2`	Fitted Rasch Model object for the second group of test takers
`select`	a string, specifying the criterion that is evaluated ("CLF Criterion" or "Gini Index", abbreviations are accepted)
`grid`	a string, specifying the method that is used to generate the grid of possible shifts to be evaluated

Value

an anchorpoint object containing:

- list with results for global optimum (single grid value and criterion value)
- list with all results (all grid values and criterion values)
- string with used criteria
- string with used grid methods
- list with Rasch Model objects for both groups of test takers

References

- Strobl, C., Kopf, J., Kohler, L., von Oertzen, T. & Zeileis, A. (2021). Anchor point selection: An approach for anchoring without anchoritems. Applied Psychological Measurement, to appear.

Examples

```
# Load the SPISA data set (general knowledge quiz - more information at ?SPISA)
library("psychotree")
data("SPISA")

# Fit the Rasch Models for the two groups females and males
fit <- anchorpoint::raschFit(SPISA, resp.mat.name='spisa', group.name='gender')

# Rasch Model fit for the first and second group
rm1 <- fit$rm1
rm2 <- fit$rm2

# Fit an Anchorpoint object
ap_object <- anchorpoint(rm1,rm2,select = "Gini Index", grid = "sparse")

# inspect the Anchorpoint object
# The print function summarizes the Global Optimum for the selected methods
print(ap_object)

# The summary function summarizes the Global Optimum for the selected methods
# and shows all the other results
summary(ap_object)

# The plot function shows the criterion plot (criterion value vs. shifts).
plot(ap_object)

# To extract the criterion value and shift for a specific position on the plot,
# set location_picker = TRUE and execute the command.
# Then, click on the desired positions and press ESCAPE.
plot(ap_object, location_picker = TRUE)
```

---

checkInput *Function to check user-specific Input for the right format*

Description

Function to check user-specific Input for the right format

Usage

```
checkInput(manuelInput, resp.var, group.var)
```

Arguments

manuelInput	manual Data input as list with response matrix and grouping vector
resp.var	name of the binary response matrix
group.var	name of the binary grouping vector

Value

Data ready for anchorpoint::raschFit function

`clfCriterion` *Calculates the CLF criterion employed by Asparouhov and Muthén*

Description

Calculates the CLF criterion employed by Asparouhov and Muthén

Usage

```
clfCriterion(dist, eps = 1e-04)
```

Arguments

`dist`	distance vector
`eps`	shift for numerical stability

Value

criterion value

References

- Asparouhov, T. & Muthén, B. (2014). Multiple-group factor analysis alignment. Structural Equation Modeling: A Multidisciplinary Journal, 21:4, 495-508

`dgp_multi` *Data generating process for multidimensional Rasch models*

Description

Data generating process for multidimensional Rasch models (only two dimensions are supported at the moment)

Usage

```
dgp_multi(
  nobs,
  tlength,
  DIFpercent,
  Nr.dim = 2,
  Theta = NULL,
  discriminations = NULL,
  difficulties = NULL,
  DIF_mode = "intersect",
  d_distr = list(mean = 0, sd = 0.2),
  MultiNorm = NULL,
  itemtype = "dich"
)
```

Arguments

nobs	positive integer, number of total observations (default 1000) or positive integer vector vector of length 2, number of observations per group
tlength	interger > 0, test length (number of items)
DIFpercent	percentage of DIF items in the test
Nr.dim	positive integer, number of dimensions (default 2)
Theta	matrix of the underlying ability parameters (optional)
discriminations	binary matrix of size tlength x Nr.dim, item discrimination parameter matrix
difficulties	numeric vector of length tlength, item difficulty vector
DIF_mode	string, mode how DIF items are created, default: intersect.
d_distr	d_distr: parameters for normal distribution to generate item difficulties difficulties, default: mean = 0, sd = .2
MultiNorm	list with parameters for multivariate normal distribution to generate the abilities of the test takers in group 1 and group 2, respectively.
itemtype	type of items (default "dich" which corresponds to multidimensional Rasch model items)

Value

list consisting of:

- dat: binary response matrix
- groups: group vector (factor),
- discriminations: binary matrix containing the item discrimination parameter matrix,
- difficulties: item difficulty vector,
- DIFindex: indicating which items were generated with DIF,
- Theta,
- DIFside: which group is favored, default focal group is favored

References

Credit: Data is generated using the function simdata from **mirt** (Version: 1.32.1):

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

Examples

```
# For examples, see ?getData.
```

dgp_uni *Data generating process for unidimensional rasch model*

Description

Data generating process for unidimensional rasch model

Usage

```
dgp_uni(
  nobs,
  tlength,
  DIFpercent,
  DIFpattern = "balanced",
  DIFeffect = "constant",
  DIFamount = 0.6,
  ability = TRUE,
  sigmaable = c(1, 1),
  itemref = c(-2.522, -1.902, -1.351, -1.092, -0.234, -0.317, 0.037, 0.268, -0.571,
   0.317, 0.295, 0.778, 1.514, 1.744, 1.951, -1.152, -0.526, 1.104, 0.961, 1.314,
   -2.198, -1.621, -0.761, -1.179, -0.61, -0.291, 0.067, 0.706, -2.713, 0.213, 0.116,
    0.273, 0.84, 0.745, 1.485, -1.208, 0.189, 0.345, 0.962, 1.592)
)
```

Arguments

nobs	number of observations per group
tlength	interger > 0, test length (number of items)
DIFpercent	percentage of DIF items in the test
DIFpattern	"balanced": DIF balanced over groups "favorref","favorfoc": all DIF items favor one group
DIFeffect	data generating process for DIF effect: • normal: item parameter differences are drawn at random from a normal distribution with mean DIFamount and sd = 0.1, like in Wang et al. (2012) • uniform: item parameter differences are drawn at random from the vector `[DIFamount-0.4,DIFamount-0.2,DIFamount,DIFamount+0.2,DIFamount+0.4]` • constant: item parameter differences are defined as `DIFamount` for all items
DIFamount	magnitude of DIF
ability	should the groups differ in mean ability? (default is TRUE)
sigmaable	positive numeric vector of length two, standard deviations for person parameter distributions in the two groups (default is c(1,1))
itemref	numeric vector of length tlength (if shorter, then sampling with replacement is used), item difficulty parameter for reference group like in Wang et al. (2012)

Value

list containing:

- dat: binary response matrix
- DIFindex: indicating which items were generated with DIF
- DIFside: which group is favored per item (-1 focal, 1 reference) default: focal group is favored for all items
- itemref: item difficulty parameter for reference group
- itemfoc: item difficulty parameter for focal group
- groups: group vector (factor),

References

- Wang WC, Shih CL, Sun GW (2012). "The DIF-Free-Then-DIF Strategy for the Assessmentof Differential Item Functioning."Educational and Psychological Measurement,72(4), 687–708

Examples

```
# For examples, see ?getData.
```

`diftests` *Extend diftests function of psychotools to include offset*

Description

Extend diftests function of psychotools to include offset

Usage

```
diftests(obj1, obj2, anchor_items, adjust = "none", offset = 0)
```

Arguments

`obj1`	rasch model object 1
`obj2`	rasch model object 2
`anchor_items`	anchor items
`adjust`	p-value adjustment
`offset`	offset

Value

list containing test results, item paramters and covariance

generateGrid *This function generates the grid values.*

Description

This function generates the grid values.

Usage

```
generateGrid(
  beta1,
  beta2,
  grid_method = c("symmetric", "sparse"),
  j.length = 1000
)
```

Arguments

beta1	Coefficients from first Rasch model fit
beta2	Coefficients from second Rasch model fit
grid_method	a string, specifying the grid method that is used to generated the shifts for evaluation ("symmetric" and/or "sparse", abbreviations are accepted)
j.length	positive integer, the granularity of the symmetric grid (default: 1000)

Value

A list with the selected grid methods each having two components:

1. A vector 'c_grid' with the grid values
2. A list of matrices 'betas_grid' for each grid value (length = j.length) each matrix has three columns: coefficient from group 1 (beta1), shifted coefficient from group 2 (newbeta2) and distance between them (beta1-newbeta2). The rows correspond to the items.

getCriterionRes *Criterion function*

Description

Criterion function

Usage

```
getCriterionRes(
  rm1,
  rm2,
  select = c("Gini Index", "CLF Criterion"),
  grid = c("symmetric", "sparse"),
  shift = NULL
)
```

Arguments

rm1	Fitted Rasch Model object corresponding to the first group. Object is of class "raschmodel", produced by function raschmodel of the package **psychotools**.
rm2	Fitted Rasch Model object corresponding to the second group. Object is of class "raschmodel", produced by function raschmodel of the package **psychotools**.
select	criterion: Gini Index or CLF Criterion
grid	grid method: symmetric or sparse
shift	desired shift. if NULL, then the criterion maximizing is used. Can also be numeric to get desired shift. Caution: must be within grid!

Value

a list which contains:

- a list with the results (grid values, criterion values, information about the optima)
- a rm object,

---

getData *Simulate data from unidimensional or multidimensional DGP*

Description

Simulate data from unidimensional or multidimensional Rasch model and two groups of test takers.

Usage

```
getData(nobs, tlength, DIFpercent, type = c("uni", "multi"), ...)
```

Arguments

nobs	positive integer, number of observations
tlength	interger > 0, test length (number of items)
DIFpercent	percentage of DIF items in the test
type	string specifying which DGP to use ("uni" or "multi")
...	additional arguments: • For unidimensional DGP, check ?anchorpoint:::dgp_uni • For multidimensional DGP, check ?anchorpoint:::dgp_multi

Value

a list containing:

- DGP: simulated data
- RM: Rasch Model objects for the two groups of test takers

Examples

```
# The number of observations
nobs = 20

# The number of items
tlength = 10

# The percentage of items to experience differential item functioning
DIFpercent = 0

# Create data from a [uni, multi]-dimensional DGP using [type = "uni",type = "multi"]:
getData(nobs, tlength, DIFpercent, type = "uni")
```

`getItemDiscrimination` *Function to create a item discrimination parameter matrix*

Description

Function to create a item discrimination parameter matrix

Usage

```
getItemDiscrimination(
  dimensions,
  DIFpercent,
  tlength,
  DIF_mode = c("intersect", "disjoint")
)
```

Arguments

`dimensions`	integer specifying the number of dimensions used in dgp (currently only 2 are allowed)
`DIFpercent`	percentage of DIF items in the test
`tlength`	interger > 0, test length (number of items)
`DIF_mode`	character vector specifying the mode how to create the matrix: • "intersect": all items load on the first, length*DIFpercent items also on the second • "disjoint": ceiling(tlength*DIFpercent) items load on the first, the rest on the second, where ceiling rounds the number up to the next integer

Value

A binary item discrimination parameter matrix of dimension: tlength x dimensions

getWald — *Function which executes Wald test for given rm object and shift (with "min_dist" setting)*

Description

Function which executes Wald test for given rm object and shift (with "min_dist" setting)

Usage

```
getWald(rm, shift)
```

Arguments

rm	A list containing the two Rasch Model objects of group 0 and group 1
shift	Shift in item parameters for the second group

Value

A list containing the output of the function Wald_test():

- p: results from the test (p-values)
- vcov: the covariance matrices of the fit (from diftests function)

get_covmat — *Function that gives back a covariance matrix for n dimesnions*

Description

Function that gives back a covariance matrix for n dimesnions

Usage

```
get_covmat(Nr.dim, variances = 0.25, covariances = 0.125)
```

Arguments

Nr.dim	integer - the number of dimensions
variances	numeric, positive, <= 1, (same for all dimensions) or Nr.dim-dimensional vector - variance of each dimension
covariances	numeric, positive, <= 1, (same for all dimensions) or choose(Nr.dim,2)-dimensional vector - covariances between dimensions

Value

covariance matrix of dimension Nr.dim x Nr.dim

get_results — *Function to evaluate criterion values and obtain test results for a given grid and method*

Description

Function to evaluate criterion values and obtain test results for a given grid and method

Usage

```
get_results(grid, shift, getTestResults, rm, metric)
```

Arguments

grid	The grid values: output of the "generateGrid.R" function
shift	the desired shift
getTestResults	logic, whether test should be applied
rm	list containing the two Rasch Model corresponding two group 0 and 1
metric	criterion to evaluate as a function

Value

a list containing the criterion evaluated at grid points and the result of the Wald test

graphicalTest — *Function to produce graphical test plot*

Description

Function to produce graphical test plot

Usage

```
graphicalTest(
  object,
  shift = NULL,
  highlight = NULL,
  alpha = 0.05,
  testColors = list(`not significant` = "darkgreen", significant = "red3", anchoritem =
    "black"),
  TestResults = NULL,
  ask = TRUE,
  ...
)
```

Arguments

object	anchorpoint object as produced by the function anchorpoint
shift	shift in item parameters for the second group, default NULL (for global optimum), else numeric (for user-defined shift)
highlight	positive integer(s), numbers of the items to be highlighted
alpha	significance level for DIF test
testColors	list with colors for the items: • "not significant" = "darkgreen" • "significant" = "red3" • "anchoritem" = "black"
TestResults	Waldtest object from anchorpoint::getWald. If NULL, then they are computed within the function. Default: NULL.
ask	logical, ask for next plot. Default = TRUE
...	further arguments for plot() like lty, cex.axis, cex.main, cex.lab etc.

References

Credit: Part of the code is adapted from the function plotGOF of the package **eRm** (Version: 1.32.1):

- Mair P, Hatzinger R. Extended Rasch modeling: The eRm package for the application of IRT models in R. Journal of Statistical Software. 2007;20 (9) :1-20.

Examples

```
# Load the SPISA data set (general knowledge quiz - more information at ?SPISA)
library("psychotree")
data("SPISA")

# Fit the Rasch Models for the two groups females and males
fit <- raschFit(SPISA, resp.mat.name='spisa', group.name='gender')

# Rasch Model fit for the first and second group
rm1 <- fit$rm1
rm2 <- fit$rm2

# Fit an Anchorpoint object
ap_object <- anchorpoint(rm1,rm2,select = "Gini Index", grid = "sparse")

# Use the Anchorpoint object to get the graphical test
graphicalTest(ap_object)
```

---

plot.anchorpoint *Plot function hand over location_picker = TRUE, to identify specific points in the plot to terminate the function, press any mouse button other than the first (X11 device) or press ESC key (quartz) see ?identify for help*

Description

Plot function hand over location_picker = TRUE, to identify specific points in the plot to terminate the function, press any mouse button other than the first (X11 device) or press ESC key (quartz) see ?identify for help

Usage

```
## S3 method for class 'anchorpoint'
plot(x, ask = T, location_picker = FALSE, ...)
```

Arguments

x	anchorpoint object as produced by the function anchorpoint
ask	logical, ask for next plot. Default = TRUE
location_picker	logical, use location picker. Default FALSE.
...	additional parameters for plot function as for standard plot function (e.g. col)

---

plotCriterion *Function to produce criterion plot*

Description

Function to produce criterion plot

Usage

```
plotCriterion(
  object,
  names,
  location_picker = FALSE,
  lty = 1,
  col = 1,
  cex.axis = 1,
  cex.lab = 1,
  cex.main = 1,
  cex = 1
)
```

Arguments

object	anchorpoint object as produced by the function anchorpoint
names	list, with criterion and grid: names of the methods used.
location_picker	use location picker
lty	line type
col	color
cex.axis	cex.axis
cex.lab	cex.lab
cex.main	cex.main
cex	cex

Value

selected points with additional information

`print.anchorpoint` *Print function summarizes the*

Description

Print function summarizes the

Usage

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

Arguments

`x`	anchorpoint object as produced by the function `anchorpoint`
`...`	further arguments passed to or from other methods (e.g. digits for rounding).

`print.plot.anchorpoint`
Print function for plot.anchorpoint

Description

Print function for plot.anchorpoint

Usage

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

Arguments

`x`	plot.anchorpoint object
`...`	further arguments passed to or from other methods.

print.summary.anchorpoint *Print function for summary.anchorpoint*

Description

Print function for summary.anchorpoint

Usage

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

Arguments

x	summary.anchorpoint object
...	further arguments passed to or from other methods.

print.WaldtestpV *Print function for WaldtestpV object*

Description

Print function for WaldtestpV object

Usage

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

Arguments

x	Wald test object
...	further arguments passed to or from other methods.

raschFit *Fits Rasch models for the reference group 0 and the focal group 1*

Description

Fits Rasch models for the reference group 0 and the focal group 1

Usage

```
raschFit(data, resp.mat.name = "i", group.name = "groups")
```

Arguments

data	• data.frame - simulated or a real data. Must contain: • response item matrix (matrix), binary (0/1) input. • group (vector), the group of the test takers.
resp.mat.name	string vector, the name of the response matrix in 'data' input with 'i' as a default (as dgp).
group.name	string vector, the group name in the data frame 'data' (as dgp).

Value

two objects of class "raschmodel", produced by function `RaschModel.fit` of the package **psychotools**.

Examples

```
# Load the SPISA data set (general knowledge quiz - more information at ?SPISA)
library("psychotree")
data("SPISA")

# Fit the Rasch Models for the two groups females and males
fit <- raschFit(SPISA, resp.mat.name='spisa', group.name='gender')
```

shiftPlot *Function to produce a shift plot*

Description

Function to produce a shift plot

Usage

```
shiftPlot(
  object,
  shift = NULL,
  testColors = list(`not significant` = "darkgreen", significant = "red3", anchoritem =
    "black"),
  testPCH = list(`not significant` = 21, significant = 22, anchoritem = 23),
  addLegend = TRUE,
  highlight = NULL,
  digits = 3,
  cex.legend = 0.75,
  TestResults = NULL,
  ask = TRUE,
  ...
)
```

Arguments

object	anchorpoint object as produced by the function anchorpoint
shift	shift in item parameters for the second group, default NULL (for global optimum), else numeric (for user-defined shift)
testColors	list with colors for the items: • "not significant" = "darkgreen" • "significant" = "red3" • "anchoritem" = "black"
testPCH	list with pch for the items (for color blind people): • "not significant" = 21 • "significant" = 22 • "anchoritem" = 23
addLegend	logic, add a legend to the plot, default: False
highlight	positive integer(s), numbers of the items to be highlighted
digits	positive integer, controls rounding of the shift in title
cex.legend	numeric, controls size of legend
TestResults	Waldtest object from anchorpoint::getWald. If NULL, then they are computed within the function. Default: NULL.
ask	logical, ask for next plot. Default = TRUE
...	additional graphics arguments

Examples

```
# Load the SPISA data set (general knowledge quiz - more information at ?SPISA)
library("psychotree")
data("SPISA")

# Fit the Rasch Models for the two groups females and males
fit <- raschFit(SPISA, resp.mat.name='spisa', group.name='gender')

# Rasch Model fit for the first and second group
rm1 <- fit$rm1
```

```
rm2 <- fit$rm2

# Fit an Anchorpoint object
ap_object <- anchorpoint(rm1,rm2,select = "Gini Index", grid = "sparse")

# Use the Anchorpoint object to get the shift plot
shiftPlot(ap_object)
```

`summary.anchorpoint` *Summary function*

Description

Summary function

Usage

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

Arguments

`object`	anchorpoint object as produced by the function `anchorpoint`
`...`	additional arguments affecting the summary produced.

`WaldtestpV` *Function to get Wald test p-value results*

Description

Function to get Wald test p-value results

Usage

```
WaldtestpV(object, shift = NULL, ...)
```

Arguments

`object`	anchorpoint object as produced by the function `anchorpoint`
`shift`	shift in item parameters for the second group, default NULL (for global optimum), else numeric (for user-defined shift)
`...`	further arguments for signif(x,...) (digits)

Examples

```
# Load the SPISA data set (general knowledge quiz - more information at ?SPISA)
library("psychotree")
data("SPISA")

# Fit the Rasch Models for the two groups females and males
fit <- raschFit(SPISA, resp.mat.name='spisa', group.name='gender')

# Rasch Model fit for the first and second group
rm1 <- fit$rm1
rm2 <- fit$rm2

# Fit an Anchorpoint object
ap_object <- anchorpoint(rm1,rm2,select = "Gini Index", grid = "sparse")

# Obtain the Wald test p-values
WaldtestpV(ap_object)
```

`Wald_test` *Function to create a item discrimination parameter matrix*

Description

Function to create a item discrimination parameter matrix

Usage

```
Wald_test(
  rm1,
  rm2,
  c_shift,
  alias_method = c("constant4_MPT", "quasi_var", "min_dist"),
  alias_anchor_items = NULL,
  adjust = "none"
)
```

Arguments

`rm1`	Fitted Rasch Model object corresponding to the first group. Object is of class "raschmodel", produced by function `RaschModel.fit` of the package **psychotools**.
`rm2`	Fitted Rasch Model object corresponding to the first group. Object is of class "raschmodel", produced by function `RaschModel.fit` of the package **psychotools**.
`c_shift`	The shift of the second group
`alias_method`	character vector specifying the aliasing method. One of "constant4_MPT", "quasi_var", "min_dist".
`alias_anchor_items`	integer in 1,...,#items. Default: NULL, will be chosen according to alias_method
`adjust`	p-value adjustment (multiple testing correction), Default: "none"

Value

list containing

- p: results from the test (p-values)
- vcov: the covariance matrices of the fit (from diftests function)

Index