Package ‘equateIRT’

October 2, 2018

Type Package
Title IRT Equating Methods
Imports statmod, stats, utils, mirt
Suggests knitr, ltm, rmarkdown, sna
Version 2.1.0
Author Michela Battauz
Maintainer Michela Battauz <michela.battauz@uniud.it>

Test scoring can be performed by true score equating and observed score equating methods.

DIF detection can be performed using a Wald-type test (Battauz (2018) <doi:10.1007/s10260-018-00442-w>).

License GPL-3
VignetteBuilder knitr
NeedsCompilation no
Repository CRAN
Date/Publication 2018-10-02 18:30:05 UTC

R topics documented:

  equateIRT-package .................................................. 2
  alldirec .............................................................. 4
  bisectorec ............................................................ 6
  chainec ............................................................... 8
  convert ............................................................... 10
  data2pl ............................................................... 12
  dataDIF .............................................................. 13
  dif.test ............................................................... 13
  direc ................................................................. 16
  eqc ................................................................. 19
Description

This package computes direct, chain and average (bisector) equating coefficients with standard errors using IRT methods for dichotomous items. The IRT models included are the three-parameter logistic model, the two-parameter logistic model, the one-parameter logistic model and the Rasch model. Test scoring can be performed by true score equating and observed score equating methods. DIF detection can be performed using a Wald-type test.

Details

Package: equateIRT
Type: Package
Version: 2.1.0
Date: 2018-09-28
License: GPL-3

Direct equating coefficients and their standard errors between forms presenting common items can be computed using function direc. The equating methods implemented are "mean-mean", "mean-geometric mean", "mean-sigma", "Haebara" and "Stocking-Lord". Estimates of item parameters and their covariance matrix can be imported from the R packages ltm and mirt or from the IRT programs IRTPRO and flexMIRT using functions import.ltm, import.mirt, import.irtpro and import.flexmirt. Item parameter estimates from other software can be imported as well by the user. Data should be previously organized using function modIRT. Function alldirec computes all direct equating coefficients (with standard errors) between all pairs of a list of forms. Function chainec computes chain equating coefficients (and standard errors) given direct equating coefficients between forms directly linked. Average equating coefficients with standard errors can be calculated using function bisectorec, that implements the (weighted) bisector method. Once the
equating coefficients are obtained, the computation of equated scores can be performed using function score, which implements true score equating and observed score equating. Standard errors of equated scores are also provided. Three simulated datasets are available for illustrative purposes. These datasets contain item parameter coefficients and their covariance matrix. In particular, est3pl concerns a three-parameter logistic model, est2pl regards a two-parameter logistic model, and estrasch refers to a Rasch model. The estimates included in est2pl are obtained from the dataset data2pl, also contained in the package. Function dif.test performs a Wald-type test for the detection of DIF (Battauz, 2018).

Author(s)
Michela Battauz
Maintainer: Michela Battauz <michela.battauz@uniud.it>

References
Description

Calculates direct equating coefficients and standard errors using IRT methods between all pairs of a list of forms.

Usage

```r
alldirec(mods, method = "mean-mean", all = FALSE, quadrature = TRUE, nq = 30,
        direction = "both")
```

Arguments

- **mods**: an object of the class `modIRT` containing item parameter coefficients and their covariance matrix of the forms to be equated.
- **method**: the equating method to be used. This should be one of "mean-mean", "mean-gmean", "mean-sigma", "Haebara" or "Stocking-Lord".
- **all**: logical; if FALSE forms that do not have common items will not appear in the output.
- **quadrature**: logical; if TRUE the Gauss-Hermite quadrature is used to approximate the integral in the function that is minimized in the Haebara and Stocking-Lord methods. If FALSE the integral is replaced with a sum over 40 equally spaced values ranging from -4 to 4 with an increment of 0.05 and weights equal to one for all values.
- **nq**: number of quadrature points used for the Gauss-Hermite quadrature if `quadrature` is TRUE.
- **direction**: a character string specifying the direction used to equate pair of forms. If "back" newer forms are equated to older forms, if "forward" older forms are equated to newer forms, if "both" (the default) equating is performed in both directions. Form A is considered to be newer than Form B if it is listed after Form B in `mods`.

Value

An object of class `eqclist` consisting in a list with length equal to the number of pairs of forms equated. Each component of the list is an object of class `eqc` returned by function `direc`.

Author(s)

Michela Battauz
alldirect

References


See Also

direc, eqc, itm, modIRT, summary.eqclist

Examples

```r
# three-parameter logistic model
# direct equating coefficients using the "Stocking-Lord" method
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
direclist3pl <- alldirec(mods = mod3pl, method = "Stocking-Lord")
summary(direclist3pl)
summary(direclist3pl$test1.test2)

# two-parameter logistic model
# direct equating coefficients using the "Haebra" method
data(est2pl)
test <- paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
direclist2pl <- alldirec(mods = mod2pl, method = "Haebra")
summary(direclist2pl)
summary(direclist2pl$test1.test5)

# Rasch model
# direct equating coefficients using the "mean-mean" method
data(estrasch)
test <- paste("test", 1:5, sep = "")
modrasch <- modIRT(coef = estrasch$coef, var = estrasch$var, names = test, display = FALSE)
direclistrasch <- alldirec(mods = modrasch, method = "mean-mean", all = TRUE)
summary(direclistrasch)
summary(direclistrasch$test5.test4)
```
Description

Calculates average equating coefficients using the bisector method and standard errors given a set of direct and chain equating coefficients.

Usage

`bisectorec(ecall, mods = NULL, weighted = TRUE, unweighted = TRUE)`

Arguments

- `ecall` list of objects of class `eqc` or `ceqc` returned by functions `direc` and `chainec`.
- `mods` an object of class `modirt` containing item parameter coefficients and their covariance matrix of the forms to be equated. From version 2.0 it can be `NULL`.
- `weighted` logical; if `TRUE` weighted bisector coefficients are computed.
- `unweighted` logical; if `TRUE` unweighted bisector coefficients are computed.

Value

An object of class `meqc` with components

- `coef` data frame containing `link`, `path`, `coefficient A`, `coefficient B`, standard error of coefficient A (`sea`), standard error of coefficient B (`seB`) and weights of direct, chain and bisector equating coefficients.
- `method` the equating method used.

Author(s)

Michela Battauz

References


See Also

`chainec`, `convert`, `direc`, `eqc`, `summary.meqc`
Examples

# three-parameter logistic model
# direct equating coefficients using the "Stocking-Lord" method
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
direclist3pl <- alldirec(mods = mod3pl, method = "Stocking-Lord")
# compute chain equating coefficients for path 1,2,3,4,5
pth3 <- paste("test", 1:5, sep = "")
chainec3 <- chainec(direclist = direclist3pl, pths = pth3)
# create a list of objects of class eqc or ceqc
ecall <- c(chainec3, direclist3pl["test1.test5"])
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)
summary(allec)

# two-parameter logistic model
# direct equating coefficients using the "Haebara" method
data(est2pl)
test <- paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclist2pl, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- c(paste("test", c(1,5,4), sep = ""))
chainec2 <- chainec(direclist = direclist2pl, pths = pth2)
# create a list of objects of class eqc or ceqc
ecall <- c(chainec1, chainec2, direclist2pl["test1.test5"])
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)
summary(allec)

# Rasch model
# direct equating coefficients using the "mean-mean" method
data(estrasch)
test <- paste("test", 1:5, sep = "")
modrasch <- modIRT(coef = estrasch$coef, var = estrasch$var, names = test, display = FALSE)
direclistrasch <- alldirec(mods = modrasch, method = "mean-mean", all = TRUE)
# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclistrasch, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- c(paste("test", c(1,5,4), sep = ""))
chainec2 <- chainec(direclist = direclistrasch, pths = pth2)
# create a list of objects of class eqc or ceqc
ecall <- c(chainec1, chainec2)
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)
summary(allec)

---

chainec

Chain Equating Coefficients

Description

Calculates chain (indirect) equating coefficients and standard errors using IRT methods.

Usage

chainec(r = NULL, direclist, f1 = NULL, f2 = NULL, pths = NULL)

Arguments

- `r`: length of the chain, that is the number of forms used for equating including extremes. It should be at least 3. It does not need to be specified if argument `pths` is not NULL.
- `direclist`: an object of the class `eqclist` return by function `alldirec` containing direct equating coefficients between pairs of forms.
- `f1`: the name of the first form of the chain.
- `f2`: the name of the last form of the chain.
- `pths`: vector, matrix or data frame containing the path used for equating. The number of columns is equal to `r` and the number of rows is equal to the equatings that have to be performed. If NULL all the chain equating coefficients of length `r` will be computed.

Details

Equating coefficients perform the conversion from the scale of the first form to the scale of the last form of the path.

Value

An object of class `ceqclist` consisting in a list with length equal to the number of chain equating coefficients computed. Each component of the list is an object of class `ceqc` with components

- `tab1`: item parameters of the first form.
- `tab2`: item parameters of the last form.
- `tab`: Data frame containing item names (Item), item parameters of the first form (e.g. `test1`), item parameters of the last form (e.g. `test3`), and item parameters of the first form converted in the scale of the last form (e.g. `test1.as.test3`).
- `varAll`: covariance matrix of item parameters of all forms used in the chain.
partial  partial derivatives of equating coefficients A and B with respect to item parameters.
A   equating coefficient A.
B   equating coefficient B.

\[ \text{covAB} \]  covariance matrix of the equating coefficients.
\[ \text{commonitem} \]  list of length r-1 containing the names of common item parameters between adjacent forms.
\[ \text{ni} \]  vector containing number of common items between pairs of adjacent forms.
\[ \text{forms} \]  names of equated forms.
\[ \text{method} \]  the equating method used.
\[ \text{itmp} \]  number of item parameters of the IRT model.

Author(s)
Michela Battauz

References

See Also
alldirec, eqc, itm, summary.ceqc, summary.ceqclist

Examples
# three-parameter logistic model
# direct equating coefficients using the "Stocking-Lord" method
data(est3pl)
test<-paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
direclist3pl <- alldirec(mods = mod3pl, method = "Stocking-Lord")
# compute all chain equating coefficients of length 4
chainec4 <- chainec(r = 4, direclist = direclist3pl)
summary(chainec4)
summary(chainec4$test1.test2.test3.test4)
# compute all chain equating coefficients of length 4
# where the first form is test1
chainec4.1 <- chainec(r = 4, direclist = direclist3pl, f1 = "test1")
summary(chainec4.1)
# compute all chain equating coefficients of length 4
# where the first form is test1 and the last form is test4
chainec4.14 <- chainec(r = 4, direclist = direclist3pl, f1 = "test1", f2 = "test4")
summary(chainec4.14)
# two-parameter logistic model
# direct equating coefficients using the "Haebara" method
data(est2pl)
test <- paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# compute chain equating coefficients of a given path
pth <- paste("test", c(1,5,4), sep = "")
chainec154 <- chainec(direclist = direclist2pl, pths = pth)
summary(chainec154)

# Rasch model
# direct equating coefficients using the "mean-mean" method
data(estrasch)
test <- paste("test", 1:5, sep = "")
modrasch <- modIRT(coef = estrasch$coef, var = estrasch$var, names = test, display = FALSE)
direclistrasch <- alldirec(mods = modrasch, method = "mean-mean", all = TRUE)
# compute chain equating coefficients of two given paths
pth1 <- paste("test", 1:3, sep = "")
pth2 <- paste("test", c(1,5,4), sep = "")
pths <- rbind(pth1, pth2)
chainec1 <- chainec(direclist = direclistrasch, pths = pths)
summary(chainec1)

---

**convert**

**Item Parameters Conversion**

**Description**

Converts item and person parameters using equating coefficients.

**Usage**

```r
convert(A, B, coef = NULL, person.par = NULL)
```

**Arguments**

- **A**: equating coefficient A.
- **B**: equating coefficient B.
- **coef**: vector of item parameters return by function `modIRT`.
- **person.par**: vector of person parameters estimates.
Details

Difficulty parameters $b$ are converted using transformation $b \cdot A + B$; discrimination parameters $a$ are converted using transformation $a/A$; guessing parameters $c$ are not transformed.

Person parameters $\theta$ are converted using transformation $\theta \cdot A + B$.

Value

A list with components

- coef: vector of item parameters transformed.
- person.par: vector of person parameters transformed.

Author(s)

Michela Battauz

References


See Also

itm, modirt

Examples

# conversion using direct coefficients
# three-parameter logistic model
# direct equating coefficients between forms 1 and 2 using the Stocking-Lord method
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modirt(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
l12 <- direc(mod1 = mod3pl[1], mod2 = mod3pl[2], method = "Stocking-Lord")
convert(A = l12$A, B = l12$B, coef = coef(mod3pl$test1))
# the conversion of item parameters is obtained also using itm(l12)

# conversion using bisector coefficients
# two-parameter logistic model
# direct equating coefficients using the "Haebara" method
data(est2pl)
test <- paste("test", 1:5, sep = "")
mod2pl <- modirt(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclist2pl, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- c(paste("test", c(1,5,4), sep = ""))
chainec2 <- chainec(direclist = direclist2pl, pths = pth2)
# create a list of objects of class ceqc
ecall <- c(chainec1, chainec2)
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)
summary(allec)
eqc14<-eqc(allec, link = "test1.test4", path = "bisector")
convert(A = eqc14$A, B = eqc14$B, coef = coef(mod2pl$test1), person.par = seq(-3, 3, 0.5))

---

### data2pl

#### Simulated Data Sets

**Description**

Five simulated data sets from a two-parameter logistic model.

**Usage**

```r
data(data2pl)
```

**Format**

A list of length 5, containing 5 data frames with 5000 dichotomous responses to 20 items.

**Author(s)**

Michela Battauz

**See Also**

`est2pl, import.ltm`

**Examples**

```r
data(data2pl)
data2pl[[1]][1:3,]
```
dataDIF

Simulated Data Set with DIF

Description

Item responses and group membership.

Usage

data(dataDIF)

Format

A dataframe with 3000 dichotomous responses to 20 items (I01 to I20) and group membership (group).

Author(s)

Michela Battauz

See Also

dif.test

Examples

data(dataDIF)

dif.test

DIF Test

Description

Performs a Wald-type test for Differential Item Functioning detection.

Usage

dif.test(coef, var, names = NULL, reference = NULL, method = "mean-mean", quadrature = TRUE, nq = 30, DIFtype = NULL, purification = FALSE, signif.level = 0.05, trace = FALSE, maxiter = 30, anchor = NULL)
Arguments

- **coef**: list of matrices (one for each group) containing the item parameter estimates. Guessing, difficulty and discrimination parameters should strictly be given in this order and they are contained in different columns of the matrix. The names of the rows of each matrix should be the names of the items.

- **var**: list of matrices (one for each group) containing the covariance matrix of item parameter estimates. They should be given in the same order of coefficients.

- **names**: character vector containing the names of the groups. This should have the same length of **coef** and **var**. If NULL, the names of the groups will be "T1", "T2", ...

- **reference**: reference group. Can be specified by name or number. The default is the first group.

- **method**: the equating method to be used in function *direc* to convert the item parameters to the scale of the reference group. This should be one of "mean-mean", "mean-gmean", "mean-sigma", "Haebara" or "Stocking-Lord".

- **quadrature**: logical; if TRUE the Gauss-Hermite quadrature is used in function *direc* to approximate the integral in the function that is minimized in the Haebara and Stocking-Lord methods. If FALSE the integral is replaced with a sum over 40 equally spaced values ranging from -4 to 4 with an increment of 0.05 and weights equal to one for all values.

- **nq**: number of quadrature points used for the Gauss-Hermite quadrature if quadrature is TRUE.

- **diftype**: character indicating which parameters to test for DIF. If NULL all parameters are tested for DIF. Use "beta1" for $\beta_1$, "beta2" for $\beta_2$, "beta3" for $\beta_3$, "beta12" for $\beta_1$ and $\beta_2$, "beta123" for $\beta_1$, $\beta_2$ and $\beta_3$. See details.

- **purification**: logical. if TRUE the procedure described in Candell and Drasgow (1988) is applied.

- **signif.level**: significance level to use in the purification process.

- **trace**: logical. If TRUE tracing information is produced.

- **maxiter**: The maximum number of iterations in the purification process.

- **anchor**: Optional character vector containing the names of the items to use for equating. These should be items free of DIF.

Details

The parameterization of the IRT model is that commonly used for estimation. Under this parameterization, the three-parameter logistic model is as follows

\[
\pi_i = c_i + (1 - c_i) \frac{\exp(\beta_{1i} + \beta_{2i} z)}{1 + \exp(\beta_{1i} + \beta_{2i} z)},
\]

where $\pi_i$ denotes the conditional probability of responding correctly to the \(i\)th item given \(z\), \(c_i\) denotes the guessing parameter, \(\beta_{1i}\) is the easiness parameter, \(\beta_{2i}\) is the discrimination parameter, and \(z\) denotes the latent ability. Furthermore, the guessing parameter is equal to

\[
c_i = \frac{\exp(\beta_{3i})}{1 + \exp(\beta_{3i})}
\]
The test verifies whether the item parameters $\beta_1, \beta_2, \beta_3$ are invariant across two or more groups as explained in Battauz (2018).

Value

An object of class difft with components

- **test** matrix containing the test statistic and the p-value for each item. "noGuess" is equal to 1 if the guessing parameter of a 3PL model was set to a fixed value.
- **eqmet** the equating method used.
- **DIFtype** character indicating which parameters were tested for DIF.
- **reference** the reference group.
- **focal** the focal groups.
- **names** names of the groups.
- **purification** logical. If TRUE the purification procedure was applied.
- **signif.level** significance level used in the purification process.
- **equatings** list containing the output of function direc.
- **coef_trasf** list containing the item parameters of each group transformed to the scale of the reference group.
- **var_trasf** list containing the covariance matrix of item parameters of each group transformed to the scale of the reference group.
- **items.dif** names of the items for which the null hypothesis of the test is rejected.
- **anchor** names of the items used as anchors.
- **niter** number of iterations.

Author(s)

Michela Battauz

References


Examples

```R
# load the data
data(dataDif)
head(dataDif)
# estimate a 2PL model for each group using the R package ltm
library(ltm)
data1 <- dataDif[dataDIF$group == 1, 1:20]
data2 <- dataDIF[dataDIF$group == 2, 1:20]
```
data3 <- dataDIF[dataDIF$group == 3, 1:20]
mod1 <- ltm(data1 ~ z1)
mod2 <- ltm(data2 ~ z1)
mod3 <- ltm(data3 ~ z1)

# extract the coefficients and the covariance matrix
est1 <- import.ltm(mod1, display = FALSE)
est2 <- import.ltm(mod2, display = FALSE)
est3 <- import.ltm(mod3, display = FALSE)

# perform the test for DIF on two groups
res_diftest2 <- dif.test(coef = list(est1$coef, est2$coef),
                         var = list(est1$var, est2$var))
res_diftest2

# perform the test for DIF on three groups
res_diftest3 <- dif.test(coef = list(est1$coef, est2$coef, est3$coef),
                         var = list(est1$var, est2$var, est3$var))
res_diftest3

# perform the test for DIF on three groups
# reference group: 2
# equating method: Haebara
# purification applied
res_diftest3 <- dif.test(coef = list(est1$coef, est2$coef, est3$coef),
                         var = list(est1$var, est2$var, est3$var),
                         method = "Haebara", purification = TRUE)
res_diftest3

---

**direc**

*Direct Equating Coefficients*

**Description**

Calculates direct equating coefficients and standard errors using IRT methods.

**Usage**

```r
direc(mods, which, mod1, mod2, method = "mean-mean", suff1 = ".1", suff2 = ".2",
      D = 1, quadrature = TRUE, nq = 30, items.select = NULL)
```

**Arguments**

- **mods**: an object of the class `modIRT` containing item parameter coefficients and their covariance matrix of the forms to be equated.
- **which**: which forms to equate. Can be specified by name or number.
- **mod1**: deprecated; please use mods instead. An object of the class `modIRT` containing item parameter coefficients and their covariance matrix of the first form.
- **mod2**: deprecated; please use mods instead. An object of the class `modIRT` containing item parameter coefficients and their covariance matrix of the second form.
method the equating method to be used. This should be one of "mean-mean", "mean-gmean", "mean-sigma", "Haebara" or "Stocking-Lord".

suff1 suffix to identify the first form to be equated.

suff2 suffix to identify the second form to be equated.

D constant D of the IRT model used to estimate item parameters. See below for more details.

quadrature logical; if TRUE the Gauss-Hermite quadrature is used to approximate the integral in the function that is minimized in the Haebara and Stocking-Lord methods. If FALSE the integral is replaced with a sum over 40 equally spaced values ranging from -4 to 4 with an increment of 0.05 and weights equal to one for all values.

nq number of quadrature points used for the Gauss-Hermite quadrature if quadrature is TRUE.

items.select optional character vector including the names of the items to use for equating.

Details

Equating coefficients perform the conversion from the scale of the first form to the scale of the second form.

In the three-parameter logistic model the probability of a positive response on item $i$ is

$$\pi_i = c_i + (1 - c_i) \frac{\exp[D a_i (\theta - b_i)]}{1 + \exp[D a_i (\theta - b_i)]},$$

where $a_i$ is the item discrimination parameter, $b_i$ is the item difficulty parameter, $c_i$ is the item guessing parameter and $\theta$ is the latent ability. The constant $D$ can be specified using argument D of the direc function. The two-parameter logistic model can be obtained by setting $c_i$ equal to zero, the one-parameter logistic model can be obtained by setting $c_i$ equal to zero and $a_i$ constant across items, while the Rasch model can be obtained by setting $c_i$ equal to zero and $a_i$ equal to 1.

The type of IRT model does not need to be specified as it is obtained from arguments mod1 and mod2.

Value

An object of class eqc with components

- **tab1** item parameters of the first form.
- **tab2** item parameters of the second form.
- **tab** Data frame containing item names (item), item parameters of the first form (e.g. test1), item parameters of the second form (e.g. test2), and item parameters of the first form converted in the scale of the second form (e.g. test1.as.test2).
- **var12** covariance matrix of item parameters of the first and the second form (only items used for equating).
- **varFull** list of covariance matrices of the item parameters (one matrix for each form)
- **partial** partial derivatives of equating coefficients A and B with respect to item parameters.
A equating coefficient A.
B equating coefficient B.
varAB covariance matrix of the equating coefficients.
commonitem list of length 2 containing the names of common item parameters and the names of the common items selected for equating.
suffixes suffixes used to identify the forms.
ni number of common items.
nis number of common items selected for equating.
forms names of equated forms.
method the equating method used.
itmp number of item parameters of the IRT model.

Author(s)
Michela Battauz

References


See Also
eqc, itm, modIRT, summary.eqc

Examples

```r
# three-parameter logistic model
# direct equating coefficients between forms 1 and 2 using the Stocking-Lord method
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
l12 <- direc(mods = mod3pl, which = c("test1", "test2"), method = "Stocking-Lord")
summary(l12)

# two-parameter logistic model
# direct equating coefficients between forms 1 and 5 using the Haabara method
data(est2pl)
test <- paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
l15 <- direc(mods = mod2pl, which = c(1,5), method = "Haabara")
```
eqc

Extract Equating Coefficients

Description

eqc is a generic function which extracts the equating coefficients.

Usage

eqc(x, ...)

## S3 method for class 'eqc'
eqc(x, ...)

## S3 method for class 'eqclist'
eqc(x, link = NULL, ...)

## S3 method for class 'ceqc'
eqc(x, ...)

## S3 method for class 'ceqclist'
eqc(x, link = NULL, path = NULL, ...)

## S3 method for class 'meqc'
eqc(x, link = NULL, path = NULL, ...)

Arguments

x object of the class eqc returned by function direc or of the class eqclist returned by function alldirec or of the class ceqc and ceqclist returned by function chainec or of the class meqc returned by function bisectorec.

link a character string with the names of the two forms being linked separated by a dot (e.g. "test1.test2").

path a character string with the names of the forms that constitute the path separated by a dot (e.g. "test1.test2.test3").

... further arguments passed to or from other methods.
Value

A data frame containing the equating coefficients for every link and path.

Author(s)

Michela Battauz

See Also

alldirec, bisectorec, chainec, direc

Examples

# two-parameter logistic model
data(est2pl)
test<-paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
# direct equating coefficients between forms 1 and 2 using the Haebara method
l12 <- direc(mods = mod2pl, which = c(1,2), method = "Haebara")
# all direct equating coefficients using the Haebara method
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# compute all chain equating coefficients of length 3
chainec3 <- chainec(r = 3, direclist = direclist2pl)
# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclist2pl, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- c(paste("test", c(1,5,4), sep = ""))
chainec2 <- chainec(direclist = direclist2pl, pths = pth2)
# create a list of objects of class ceqc
ecall <- c(chainec1, chainec2)
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)

# extract equating coefficients
eqc(l12)
eqc(direclist2pl)
eqc(direclist2pl, link = "test1.test2")
eqc(chainec3)
eqc(chainec3, link = "test1.test3")
eqc(allec)
eqc(allec, path = "bisector")
Description

This dataset includes item parameter estimates and covariance matrices of a two-parameter logistic model applied to 5 simulated datasets with common items. The dichotomous item responses can be found in the dataset data2pl. See details for more information on the linkage plan.

Usage

data(est2pl)

Format

A list of length 2 with components:

- **coef**: a list of length 5 containing the matrices of item parameter estimates. Each matrix presents 2 columns; the first column contains difficulty parameters and the second column contains discrimination parameters. See details for information on the parameterization used. Names of rows correspond to the names of the items.
- **var**: a list of length 5 containing the covariance matrices of item parameter estimates.

Details

Every form is composed by 20 items and presents 10 items in common with adjacent forms. Furthermore, forms 1 and 5 present 10 common items. Use `linkp` to obtain a matrix with elements equal to the number of common items between different forms.

Item parameters are given under the parameterization used in the `ltm` package. Under this parameterization, the two-parameter logistic model is as follows

\[
\pi_i = \frac{\exp(\beta_{1i} + \beta_{2i}z)}{1 + \exp(\beta_{1i} + \beta_{2i}z)},
\]

where \(\pi_i\) denotes the conditional probability of responding correctly to the \(i\)th item given \(z\), \(\beta_{1i}\) is the easiness parameter, \(\beta_{2i}\) is the discrimination parameter, and \(z\) denotes the latent ability.

Author(s)

Michela Battauz

See Also

data2pl, linkp, modIRT

Examples

data(est2pl)
est2pl$coef
est2pl$varlinkp(coef = est2pl$coef)
Item Parameter Estimates and Covariance Matrices of a Three-Parameter Logistic Model

Description

This dataset includes item parameter estimates and covariance matrices of a three-parameter logistic model applied to 5 simulated datasets with common items. See details for more information on the linkage plan.

Usage

data(est3pl)

Format

A list of length 2 with components:

- coef: a list of length 5 containing the matrices of item parameter estimates. Each matrix presents 3 columns; the first column contains guessing parameters, the second column contains difficulty parameters and the third column contains discrimination parameters. See details for information on the parameterization used. Names of rows correspond to the names of the items.

- var: a list of length 5 containing the covariance matrices of item parameter estimates.

Details

Every form is composed by 20 items and presents 10 items in common with adjacent forms. Furthermore, forms 1 and 5 present 10 common items. Use linkp to obtain a matrix with elements equal to the number of common items between different forms.

Item parameters are given under the parameterization used in the ltm package. Under this parameterization, the three-parameter logistic model is as follows

\[
\pi_i = c_i + (1 - c_i) \frac{\exp(\beta_{1i} + \beta_{2i} z)}{1 + \exp(\beta_{1i} + \beta_{2i} z)},
\]

where \( \pi_i \) denotes the conditional probability of responding correctly to the \( i \)th item given \( z \), \( c_i \) denotes the guessing parameter, \( \beta_{1i} \) is the easiness parameter, \( \beta_{2i} \) is the discrimination parameter, and \( z \) denotes the latent ability. Furthermore, the guessing parameters are given under this parameterization

\[
c_i = \frac{\exp(c_i)}{1 + \exp(c_i^*)}.
\]

Author(s)

Michela Battauz
See Also

`linkp, modIRT`

Examples

```r
data(est3pl)
est3pl$coef
est3pl$var
linkp(coef = est3pl$coef)
```

Description

This dataset includes item parameter estimates and covariance matrices of a Rasch model applied to 5 simulated datasets with common items. See details for more information on the linkage plan.

Usage

```r
data(estrasch)
```

Format

A list of length 2 with components:

- `coef` a list of length 5 containing the matrices of item parameter estimates. Each matrix presents 2 columns; the first column contains difficulty parameters and the second column is equal to 1. See details for information on the parameterization used. Names of rows correspond to the names of the items.
- `var` a list of length 5 containing the covariance matrices of item parameter estimates.

Details

Every form is composed by 20 items and presents 10 items in common with adjacent forms. Furthermore, forms 1 and 5 present 10 common items. Use `linkp` to obtain a matrix with elements equal to the number of common items between different forms.

Item parameters are given under the parameterization used in the `ltm` package. Under this parameterization, the Rasch model is as follows

\[
\pi_i = \frac{\exp(\beta_{1i} + z)}{1 + \exp(\beta_{1i} + z)},
\]

where \( \pi_i \) denotes the conditional probability of responding correctly to the \( i \)th item given \( z \), \( \beta_{1i} \) is the easiness parameter, and \( z \) denotes the latent ability.

Author(s)

Michela Battauz
See Also

`linkp`, `modIRT`

Examples

```r
data(estrasch)
estrasch$coef
estrasch$var
linkp(coef = estrasch$coef)
```

---

**import.ltm**

**Import Item Parameters Estimates and Covariance Matrices from IRT Software**

**Description**

Import estimated item parameters and covariance matrix from the R packages ltm and mirt, and from external software IRTPRO and flexMIRT.

**Usage**

```r
import.ltm(mod, display = TRUE, digits = 4)
import.mirt(mod, display = TRUE, digits = 3)
import.irtpro(fnamep, fnamev = NULL, fnameirt = NULL, display = TRUE, digits = 2)
import.flexmirt(fnamep, fnamev = NULL, fnameirt = NULL, display = TRUE, digits = 2)
```

**Arguments**

- `mod`: output object from functions `rasch`, `ltm`, or `tpm` of the `ltm` package or from function `mirt` of the `mirt` package.
- `display`: logical; if `TRUE` coefficients and standard errors are printed.
- `digits`: integer indicating the number of decimal places to be used if `display` is `TRUE`.
- `fnamep`: name of the file containing the estimated item parameters. Typically, `-prm.txt`.
- `fnamev`: name of the file containing the covariance matrix of the estimated item parameters. Typically, `-cov.txt`.
- `fnameirt`: name of the file containing additional information to link item parameters with the covariance matrix. Typically, `-irt.txt`.

**Details**

Item parameters are imported with the parameterization used by the software to estimate the IRT model. The usual IRT parameterization can be obtained later by using function `modIRT`. 
Value

- A list with components
  - `coef`: item parameter estimates.
  - `var`: covariance matrix of item parameter estimates.

Author(s)

Michela Battauz

References


See Also

- `modIRT`

Examples

```r
#============================
# from package ltm
library(ltm)

# one-parameter logistic model
mod1pl <- rasch(LSAT)
est.mod1pl <- import.ltm(mod1pl)
est.mod1pl

# two-parameter logistic model
mod2pl <- ltm(LSAT ~ z1)
est.mod2pl <- import.ltm(mod2pl)
est.mod2pl

#============================
# from package mirt
library(mirt)

# one-parameter logistic model
```
data(LSAT, package = "ltm")
val <- mirt(LSAT, 1, SE = TRUE, pars = "values")
constr <- val$val$name == "a1"?$parnum
mod1pl.m <- mirt(LSAT, 1, SE = TRUE, SE.type = 'Richardson', constrain = list(constr))
est.mod1pl.m <- import.mirt(mod1pl.m, digits = 4)
est.mod1pl.m

# two-parameter logistic model
data(LSAT, package = "ltm")
mod2pl.m <- mirt(LSAT, 1, SE = TRUE, SE.type = 'Richardson')
est.mod2pl.m <- import.mirt(mod2pl.m, display = FALSE)
est.mod2pl.m

---

**itm**

*Extract Item Parameters*

**Description**

`itm` is a generic function which extracts a data frame containing the item parameters of two forms being equated in the original scale and item parameters of the first form converted in the scale of the second form.

**Usage**

`itm(x, ...)`

- `x` object of the class `eqc` returned by function `direct` or of the class `eqclist` returned by function `alldirect` or of the class `ceqc` and `ceqclist` returned by function `chainec` or of the class `meqc` return by function `bisectorec`.

**Arguments**

- `x` object of the class `eqc` returned by function `direct` or of the class `eqclist` returned by function `alldirect` or of the class `ceqc` and `ceqclist` returned by function `chainec` or of the class `meqc` return by function `bisectorec`.

---
link  a character string with the names of the two forms being linked separated by a dot (e.g. "test1.test3").

path  a character string with the names of the forms that constitute the path separated by a dot (e.g. "test1.test2.test3").

bistype  bisector type. It should be weighted or unweighted. It could be NULL if only one type was computed by function bisectorec.

Value

A data frame containing item names (Item), item parameters of the first form (e.g. test1), item parameters of the last form (e.g. test3), and item parameters of the first form converted in the scale of the last form (e.g. test1.as.test3).

Author(s)

Michela Battauz

See Also

convert, alldirec, bisectorec, chainec, direc

Examples

# two-parameter logistic model
data(est2pl)
test<-paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
# direct equating coefficients between forms 1 and 2 using the Haebara method
l12 <- direc(mods = mod2pl, which = c(1,2), method = "Haebara")
# all direct equating coefficients using the Haebara method
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# compute all chain equating coefficients of length 3
chainec3 <- chainec(r = 3, direclist = direclist2pl)
# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclist2pl, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- c(paste("test", c(1,5,4), sep = ""))
chainec2 <- chainec(direclist = direclist2pl, pths = pth2)
# create a list of objects of class ceqc
ecall <- c(chainec1, chainec2)
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)

# extract item parameters
itm(l12)
itm(direclist2pl, link = "test1.test2")
itm(chainec3, path = "test1.test2.test3")
itm(allec, bistype = "weighted")
Description

Calculates the number of common items between a list of forms.

Usage

`linkp(coef)`

Arguments

coeff:

- list of matrices (one for each form) containing item parameter estimates. The names of the rows of each matrix should be the names of the items.

Value

A matrix whose elements indicate the number of common items between the forms. On the diagonal of the matrix there are the number of items of each form.

Author(s)

Michela Battauz

See Also

`est2pl, est3pl, estrasch`

Examples

```r
data(est3pl)
linkp(coef = est3pl$coef)
```
Arguments

coef list of matrices (one for each form) containing item parameter estimates. Guessing, difficulty and discrimination parameters should strictly be given in this order and they are contained in different columns of the matrix. The names of the rows of each matrix should be the names of the items.

var list of matrices (one for each form) containing the covariance matrix of item parameter estimates. They should be given in the same order of coefficients.

names character vector containing the names of the forms. This should have the same length of coef and var. If NULL, the names of the forms are assigned by function modIRT.

ltparam logical; if TRUE the latent trait parameterization is used for difficulty parameters and the modIRT function performs a transformation of item parameters to return them in the usual IRT parameterization. Set to FALSE to avoid transformations. See below for more details.

lparam logical; if TRUE the logistic parameterization is used for guessing parameters and the modIRT function performs a transformation of item parameters to return them in the usual IRT parameterization. Set to FALSE to avoid transformations. See below for more details.

display logical; if TRUE coefficients and standard errors are printed.

digits integer indicating the number of decimal places to be used if display is TRUE.

Details

ltparam and lparam refers the the parameterization used by the software used to estimate item parameters. The R package ltm, and the programs IRTPRO and flexMIRT use these parameterizations. If ltparam is TRUE the latent trait parameterization is used. Under this parameterization, the three-parameter logistic model is as follows

\[ \pi_i = c_i + (1 - c_i) \frac{\exp(\beta_{1i} + \beta_{2i}z)}{1 + \exp(\beta_{1i} + \beta_{2i}z)}, \]

where \( \pi_i \) denotes the conditional probability of responding correctly to the \( i \)th item given \( z \), \( c_i \) denotes the guessing parameter, \( \beta_{1i} \) is the easiness parameter, \( \beta_{2i} \) is the discrimination parameter, and \( z \) denotes the latent ability. The two-parameter logistic model, the one-parameter logistic model and the Rasch model present the same formulation. The two-parameter logistic model can be obtained by setting \( c_i \) equal to zero, the one-parameter logistic model can be obtained by setting \( c_i \) equal to zero and \( \beta_{2i} \) constant across items, while the Rasch model can be obtained by setting \( c_i \) equal to zero and \( \beta_{2i} \) equal to 1.

If lparam is TRUE the guessing parameters are given under this parameterization

\[ c_i = \frac{\exp(c_i^*)}{1 + \exp(c_i^*)}, \]

The modIRT function returns parameter estimates under the usual IRT parameterization, that is,

\[ \pi_i = c_i + (1 - c_i) \frac{\exp[D_{ai}(\theta - b_i)]}{1 + \exp[D_{ai}(\theta - b_i)]}, \]
where \( D_{\alpha_i} = \beta_{2i}, b_i = -\beta_{1i}/\beta_{2i} \) and \( \theta = z \).

If `ltparam` or `lparam` are TRUE, the covariance matrix is calculated using the delta method.

If item parameters are already given under the usual IRT parameterization, arguments `ltparam` and `lparam` should be set to FALSE.

**Value**

An object of class `modIRT` consisting in a list with length equal to the number of forms containing lists with components

- `coefficients`: item parameter estimates.
- `var`: covariance matrix of item parameter estimates.
- `itmp`: number of item parameters of the IRT model. This is 1 for the Rasch model, 2 for the one-parameter logistic model with constant discriminations, 2 for the two-parameter logistic model and 3 for the three-parameter logistic model.

**Author(s)**

Michela Battauz

**References**


**See Also**

direc, import.ltm

**Examples**

```r
# three-parameter logistic model
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)

# two-parameter logistic model
data(est2pl)
test <- paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)

# Rasch model
data(estrasch)
test <- paste("test", 1:5, sep = "")
modrasch <- modIRT(coef = estrasch$coef, var = estrasch$var, names = test,
```
score = FALSE)

# one-parameter logistic model imported from the R package ltm
library(ltm)
mod1pl <- rasch(LSAT)
summary(mod1pl)
est.mod1pl <- import.ltm(mod1pl)
mod1pl.ltm <- modirt(coef = list(est.mod1pl$coef), var = list(est.mod1pl$var), digits = 4)

---

### Scoring

**Description**

Relates number-correct scores on two forms.

**Usage**

```r
score(obj, ...)
```

```
## S3 method for class 'eqc'
score(obj, method = "TSE", D = 1, scores = NULL, se = TRUE, nq = 30,
      w = 0.5, theta = NULL, weights = NULL, ...)

## S3 method for class 'eqclist'
score(obj, link = NULL, method = "TSE", D = 1, scores = NULL, se = TRUE, nq = 30,
      w = 0.5, theta = NULL, weights = NULL, ...)

## S3 method for class 'ceqc'
score(obj, method = "TSE", D = 1, scores = NULL, se = TRUE, nq = 30,
      w = 0.5, theta = NULL, weights = NULL, ...)

## S3 method for class 'ceqclist'
score(obj, path = NULL, method = "TSE", D = 1, scores = NULL, se = TRUE, nq = 30,
      w = 0.5, theta = NULL, weights = NULL, ...)

## S3 method for class 'meqc'
score(obj, link = NULL, method = "TSE", D = 1, scores = NULL, se = TRUE,
      bistype = NULL, nq = 30, w = 0.5, theta = NULL, weights = NULL, ...)
```

**Arguments**

- `obj` output object from functions `direc`, `alldirec`, `chainec` or `bisectorec`.
- `link` a character string with the names of the two forms being linked separated by a dot (e.g. "test1.test3"). Necessary if `obj` is output of functions `alldirec` or `bisectorec`. 
path is a character string with the names of the forms that constitute the path separated by a dot (e.g. "test1.test2.test3"). Necessary if obj is output of function chainec.

method is the scoring method to be used. This should be one of "TSE" (the default) for true score equating or "OSE" for observed score equating.

D is a constant D of the IRT model used to estimate item parameters.

scores are integer values to be converted.

se is logical; is TRUE standard errors of equated scores are computed.

bistype is bisector type. It should be specified when obj is an output object from function bisectorec. It should be weighted or unweighted. It can be NULL if only one type was computed by function bisectorec.

nq is number of quadrature points used to approximate integrals with observed score equating. Used only if arguments theta and weights are NULL.

w is synthetic weight for population 1. It should be a number between 0 and 1.

theta is vector of ability values used to approximate integrals with observed score equating.

weights is vector of weights used to approximate integrals with observed score equating.

... further arguments passed to or from other methods.

Details

In this function common items are internal, i.e. they are used for scoring the test.

This function computes standard error of equated scores with the observed score equating method using \((\alpha'_{1X}, \alpha'_{1V}, A, B)'\) for the description of \(\beta_{X2,x}(At_m+B)\), and \((\alpha'_{1Y}, \alpha'_{1V2}, A, B)'\) for \(\beta_{Y1,y}(tm)\) (see Ogasawara, 2003).

Value

A data frame containing theta values (only for true score equating), scores in the form chosen as base, equated scores, and standard errors of equated scores.

Author(s)

Michela Battauz

References


See Also

`alldirec`, `bisectorec`, `chainec`, `direc`

Examples

```r
# two-parameter logistic model
data(est2pl)
test<-paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
# direct equating coefficients between forms 1 and 2 using the Haebara method
l12 <- direc(mods = mod2pl, which = c(1,2), method = "Haebara")
# scoring using direct equating coefficients and the true score equating method
score(l12)
# scoring using direct equating coefficients and the observed score equating method
score(l12, method = "OSE")
# specify only scores from 10 to 15
score(l12, method = "OSE", scores = 10:15)

# all direct equating coefficients using the Haebara method
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# scoring using direct equating coefficients and the true score equating method
score(direclist2pl, link = "test1.test2")

# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclist2pl, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- paste("test", c(1,5,4), sep = "")
chainec2 <- chainec(direclist = direclist2pl, pths = pth2)
# scoring using chain equating coefficients and the true score equating method for score 12
score(chainec1, path = "test1.test2.test3.test4", scores = 12)
score(chainec2, path = "test1.test5.test4", scores = 12)

# create a list of objects of class ceqc
ecall <- c(chainec1, chainec2)
# compute bisector and weighted bisector coefficients
allec <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)
# scoring using weighted bisector equating coefficients and the true score
# equating method for score 12
score(allec, bistype = "weighted", link = "test1.test4", scores = 12)
```

summary.ceqc

**Summarizing Estimated Chain Equating Coefficients**

**Description**

`summary` method for class `ceqc`.  

---

### Summary of estimated chain equating coefficients

#### Example

```r
# two-parameter logistic model
```data(est2pl)
test<-paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)

### Direct equating coefficients

```r
# direct equating coefficients between forms 1 and 2 using the Haebara method
```l12 <- direc(mods = mod2pl, which = c(1,2), method = "Haebara")

### Scoring

```r
# scoring using direct equating coefficients and the true score equating method
```score(l12)

### Chain equating coefficients

```r
# compute chain equating coefficients for path 1,2,3,4
```
```r
pth1 <- paste("test", 1:4, sep = "")
chainec1 <- chainec(direclist = direclist2pl, pths = pth1)
```

### Chain equating scores

```r
# scoring using chain equating coefficients and the true score equating method for score 12
```score(chainec1, path = "test1.test2.test3.test4", scores = 12)
```r
```
Usage

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

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

Arguments

- `object` an object of the class `ceqc` returned by function `chainec`.
- `x` an object of class `summary.ceqc`, a result of a call to `summary.ceqc`.
- `...` further arguments passed to or from other methods.

Value

The function `summary.ceqc` returns a list with components

- `forms` names of equated forms.
- `method` the equating method used.
- `coefficients` a 2 × 2 matrix with columns for the estimated coefficients A and B and standard errors.

Author(s)

Michela Battauz

See Also

`chainec`

Examples

```r
# two-parameter logistic model
# direct equating coefficients using the "Haabara" method
data(est2pl)
test<-paste("test", 1:5, sep = "")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
direclist2pl <- alldirec(mods = mod2pl, method = "Haabara")
# compute all chain equating coefficients of length 3
chainec3 <- chainec(r = 3, direclist = direclist2pl)
summary(chainec3$test1.test5.test4)
```
Summary

### Summary

**Summary of a List of Estimated Chain Equating Coefficients**

**Description**

Summary method for class `ceqclist`.

**Usage**

```r
## S3 method for class 'ceqclist'
summary(object, path, ...)
```

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

**Arguments**

- `object`: an object of the class `ceqclist` returned by function `chainec`
- `path`: a vector of character strings with the names of the forms that constitute the path separated by a dot (e.g. "test1.test2.test3")
- `x`: an object of class `summary.ceqclist`, a result of a call to `summary.ceqclist`
- `...`: further arguments passed to or from other methods.

**Value**

The function `summary.ceqclist` returns a list containing the output of function `summary.ceqc` for each path contained in `object`.

**Author(s)**

Michela Battauz

**See Also**

`chainec`, `summary.ceqc`

**Examples**

```r
# two-parameter logistic model
# direct equating coefficients using the "Haebara" method
data(est2pl)
test<-paste("test", 1:5, sep = ")
mod2pl <- modIRT(coef = est2pl$coef, var = est2pl$var, names = test, display = FALSE)
direclist2pl <- alldirec(mods = mod2pl, method = "Haebara")
# compute all chain equating coefficients of length 3
chainec3 <- chainec(r = 3, direclist = direclist2pl)
summary(chainec3)
summary(chainec3, path = "test1.test2.test3")
summary(chainec3, path = c("test1.test2.test3","test1.test5.test4"))
```
## Summary

### Description

summary method for class `eqc`.

### Usage

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

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

### Arguments

- `object`: an object of the class `eqc` returned by function `direc`.
- `x`: an object of class `summary.eqc`, a result of a call to `summary.eqc`.
- `...`: further arguments passed to or from other methods.

### Value

The function `summary.eqc` returns a list with components:

- `forms`: names of equated forms.
- `method`: the equating method used.
- `coefficients`: a $2 \times 2$ matrix with columns for the estimated coefficients A and B and standard errors.

### Author(s)

Michela Battauz

### See Also

direc

### Examples

```r
# three-parameter logistic model
# direct equating coefficients between forms 1 and 2 using the Stocking-Lord method
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
l12 <- direc(mod1 = mod3pl[1], mod2 = mod3pl[2], method = "Stocking-Lord")
summary(l12)
```
Summary method for class eqclist.

Usage

```r
## S3 method for class 'eqclist'
summary(object, link, ...)

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

Arguments

- **object**: an object of the class eqclist returned by function `alldirec`.
- **link**: a vector of character strings with the names of the two forms being linked separated by a dot (e.g., "test1.test2").
- **x**: an object of class `summary.eqclist`, a result of a call to `summary.eqclist`.
- **...**: further arguments passed to or from other methods.

Value

The function `summary.eqclist` returns a list containing the output of function `summary.eqc` for each link contained in `object`.

Author(s)

Michela Battauz

See Also

`alldirec, summary.eqc`

Examples

```r
# Rasch model
# direct equating coefficients using the "mean-mean" method
data(estrasch)
test<-paste("test", 1:5, sep = "")
modrasch <- modIRT(coef = estrasch$coef, var = estrasch$var, names = test, 
                   display = FALSE)
direclistrasch <- alldirec(mods = modrasch, method = "mean-mean", all = TRUE)
summary(direclistrasch)
summary(direclistrasch, link = "test1.test2")
summary(direclistrasch, link = c("test1.test2", "test1.test5"))
```
summary.meqc

Summarizing Bisector Equating Coefficients

Description

summary method for class meqc.

Usage

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

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

Arguments

object 
an object of the class meqc returned by function bisectorec.

x 
an object of class summary.meqc, a result of a call to summary.meqc.

... further arguments passed to or from other methods.

Value

The function summary.meqc returns a list with components

link character vector with names of equated forms.

method the equating method used.

coefficients list of data frames containing Path, Estimate and StdErr of direct, chain and bisector equating coefficients.

Author(s)

Michela Battauz

See Also

bisectorec

Examples

# three-parameter logistic model
# direct equating coefficients using the "Stocking-Lord" method
data(est3pl)
test <- paste("test", 1:5, sep = "")
mod3pl <- modIRT(coef = est3pl$coef, var = est3pl$var, names = test, display = FALSE)
direclist3pl <- alldirec(mods = mod3pl, method = "Stocking-Lord")
# compute chain equating coefficients for path 1,2,3,4
pth1 <- paste("test", 1:4, sep = "")
pth1 <- data.frame(t(pth1), stringsAsFactors = FALSE)
chainec1 <- chainec(direclist = direclist3pl, pths = pth1)
# compute chain equating coefficients for path 1,5,4
pth2 <- c(paste("test", c(1,5,4), sep = ""))
pth2 <- data.frame(t(pth2), stringsAsFactors = FALSE)
chainec2 <- chainec(direclist = direclist3pl, pths = pth2)
# compute chain equating coefficients for path 1,2,3,4,5
pth3 <- paste("test", 1:5, sep = "")
pth3 <- data.frame(t(pth3), stringsAsFactors = FALSE)
chainec3 <- chainec(direclist = direclist3pl, pths = pth3)
# create a list of objects of class eqc or ceqc
ecall <- c(chainec1, chainec2, chainec3, direclist3pl["test1.test5"])
# compute bisector and weighted bisector coefficients
allee <- bisectorec(ecall = ecall, weighted = TRUE, unweighted = TRUE)
summary(allee)
### Index

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>datasets</td>
<td></td>
</tr>
<tr>
<td>data2pl</td>
<td>12</td>
</tr>
<tr>
<td>dataDIF</td>
<td>13</td>
</tr>
<tr>
<td>est2pl</td>
<td>20</td>
</tr>
<tr>
<td>est3pl</td>
<td>22</td>
</tr>
<tr>
<td>estrasch</td>
<td>23</td>
</tr>
<tr>
<td>alldirec</td>
<td>4, 8, 9, 19, 20, 26, 27, 31, 33, 37</td>
</tr>
<tr>
<td>bisectorec</td>
<td>6, 19, 20, 26, 27, 31–33, 38</td>
</tr>
<tr>
<td>ceqc (chainec)</td>
<td>8</td>
</tr>
<tr>
<td>chainec</td>
<td>6, 8, 19, 20, 26, 27, 31, 33–35</td>
</tr>
<tr>
<td>convert</td>
<td>6, 10, 27</td>
</tr>
<tr>
<td>data2pl</td>
<td>12, 21</td>
</tr>
<tr>
<td>dataDIF</td>
<td>13</td>
</tr>
<tr>
<td>dif.test</td>
<td>13, 13</td>
</tr>
<tr>
<td>direc</td>
<td>4–6, 14, 15, 16, 19, 20, 26, 27, 30, 31, 33, 36</td>
</tr>
<tr>
<td>eqc</td>
<td>5, 6, 9, 18, 19</td>
</tr>
<tr>
<td>equateIRT</td>
<td>2</td>
</tr>
<tr>
<td>equateIRT-package</td>
<td>2</td>
</tr>
<tr>
<td>est2pl</td>
<td>12, 20, 28</td>
</tr>
<tr>
<td>est3pl</td>
<td>22, 28</td>
</tr>
<tr>
<td>estrasch</td>
<td>23, 28</td>
</tr>
<tr>
<td>import.flexmirt (import.ltm)</td>
<td>24</td>
</tr>
<tr>
<td>import.irtpro (import.ltm)</td>
<td>24</td>
</tr>
<tr>
<td>import.ltm</td>
<td>12, 24, 30</td>
</tr>
<tr>
<td>import.mirt (import.ltm)</td>
<td>24</td>
</tr>
<tr>
<td>itm</td>
<td>5, 9, 11, 18, 26</td>
</tr>
<tr>
<td>linkp</td>
<td>21–24, 28</td>
</tr>
<tr>
<td>modIRT</td>
<td>5, 10, 11, 18, 21, 23–25, 28</td>
</tr>
<tr>
<td>print.dift (dif.test)</td>
<td>13</td>
</tr>
<tr>
<td>print.eqc (direc)</td>
<td>16</td>
</tr>
<tr>
<td>print.eqclist (alldirec)</td>
<td>4</td>
</tr>
</tbody>
</table>

*Note: The page numbers correspond to the positions where the terms or functions are defined or used within the text.*