

# Package ‘INEPsico’

December 28, 2024

**Title** Bora Facilitar a Vida

**Version** 1.2

**Description** Automatização de rotinas para análises psicométricas.

**License** `use\_mit\_license()`, `use\_gpl3\_license()` or friends to  
pick a license

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Imports** magrittr,  
data.table,  
shiny,  
mirtCAT,  
mirt,  
dplyr,  
stringr,  
stringi,  
gdata,  
rmarkdown,  
knitr,  
kableExtra,  
LaF,  
catIrt,  
tableHTML

**Depends** R (>= 2.10)

## Contents

INEPsico-package . . . . .	2
abre.banco . . . . .	2
abre.resp . . . . .	3
adeq.par . . . . .	4
aloca.dico . . . . .	5
aloca.pessoa . . . . .	7
aloca.poli . . . . .	8
arred . . . . .	9
brincar . . . . .	10

cci . . . . .	11
dif.bilog . . . . .	12
dif.mirt . . . . .	13
freq.nivel . . . . .	15
gera.caderno . . . . .	15
gera.form . . . . .	16
info.teste . . . . .	17
ler.exp . . . . .	19
ler.par . . . . .	19
ler.sco . . . . .	20
prop.exp . . . . .	21
relatorio.tct . . . . .	21
simular . . . . .	22
tct . . . . .	23

## **Index** 25

---

INEPsico-package	<i>INEPsico: Bora Facilitar a Vida</i>
------------------	--

---

### **Description**

Automatização de rotinas para análises psicométricas

### **Details**

Este pacote objetiva automatizar rotinas de análise de itens e testes por meio da Teoria Clássica dos Testes (TCT) e Teoria de Resposta ao Item (TRI). Além de realizar análises via TCT e produzir relatórios desses resultados, o pacote cria arquivos para serem lidos no BILOG-MG. A maioria das funções que envolvem análises via TRI são voltadas para trabalhar com resultados produzidos nesse programa.

### **Author(s)**

Alexandre Jaloto [alexandre.jaloto@inep.gov.br](mailto:alexandre.jaloto@inep.gov.br)

---

abre.banco	<i>Abrir o banco</i>
------------	----------------------

---

### **Description**

Construir um banco 'aberto' a partir de uma base com informações de respostas organizada por cadernos distintos.

### **Usage**

```
abre.banco(banco, itens, bib, disc, disc.cad = 2)
```

**Arguments**

banco	Objeto que tem somente a variável correspondente ao caderno e as variáveis correspondentes às espostas aos itens
itens	O(s) objeto(s) com os itens da(s) disciplina(s) que compõe(m) os cadernos no BIB (tem que ser todos, por enquanto; depois tem que melhorar a função)
bib	Objeto com o BIB
disc	Qual será a disciplina desse banco?
disc.cad	Quantidade de disciplinas em cada caderno

**Value**

A função retorna uma lista com dois elementos: 'respostas' e 'gabarito'

**Author(s)**

Alexandre Jaloto

**Examples**

```
set.seed(1000)
gab.lc = sample (LETTERS[1:4], 9, replace = TRUE)
gab.mt = sample (LETTERS[1:4], 9, replace = TRUE)
itens.lc = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Item = sample (12345:54321, 9), Origem = 'NOVO',
                      Gabarito = gab.lc, Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'LC')
itens.mt = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Gabarito = gab.mt, Item = sample (12345:54321, 9),
                      Origem = 'NOVO', Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'MT')

bib = data.frame (Caderno = 1:3, Disciplina1 = rep ('LC', 3), Disciplina2 = rep ('MT', 3),
                 Bloco1 = 1:3, Bloco2 = c(2, 3, 1), Bloco3 = 1:3, Bloco4 = c(2, 3, 1))
itens = rbind (itens.lc, itens.mt)
resp = matrix (sample (LETTERS[1:4], 180, replace = TRUE), ncol = 6)
banco = data.frame (CAD = seq(1:3), resp)
disc = 'LC'
aberto = abre.banco (banco = banco, itens = itens, bib = bib, disc = disc)
```

---

abre.resp

*Separar o vetor de resposta*


---

**Description**

Separa um único vetor de respostas (vários itens) em vetores compostos pela resposta a um único item

**Usage**

```
abre.resp(unico)
```

**Arguments**

unico                      Objeto com o vetor das respostas a todos os itens

**Value**

A função retorna uma matriz com número de colunas igual à quantidade de itens e número de linhas igual à quantidade de participantes.

**Author(s)**

Alexandre Jaloto

**Examples**

```
arq.par = 'Rodada6.PAR'
categorias = 5
p = .65
m = 0
dp = 1
int.dp = .25
int.nivel = c (-2, 2)
met = 3
nomes.cat = data.frame (a = paste0 ('a', 1:16), b = paste0 ('b', 1:16),
                        c = paste0 ('c', 1:16), d = paste0 ('d', 1:16),
                        e = paste0 ('e', 1:16))
aloca.item.multi (arq.par, categorias, p = .65, m = 0,
                  dp = 1, int.dp = .25, int.nivel = c (-2, 2),
                  met = 3, nomes.cat = nomes.cat)
```

---

adeq.par

*Adequação dos parâmetros estimados*

---

**Description**

Verificar a adequação dos parâmetros dos itens estimados pelo BILOG-MG.

**Usage**

```
adeq.par(
  arq.par,
  a = list(0.45, 4),
  b = list(-4, 4),
  c = list(0, 0.45),
  erro.a = NULL,
  erro.b = NULL,
  erro.c = NULL,
  salvar = FALSE
)
```

**Arguments**

arq.par	Arquivo PAR geado pelo BILOG-MG
a	Vetor com o menor valor aceitável e o maior valor aceitável do parâmetro 'a'
b	Vetor com o menor valor aceitável e o maior valor aceitável do parâmetro 'b'
c	Vetor com o menor valor aceitável e o maior valor aceitável do parâmetro 'c'
erro.a	valor do maior valor aceitável para o erro do parâmetro 'a'
erro.b	valor do maior valor aceitável para o erro do parâmetro 'b'
erro.c	valor do maior valor aceitável para o erro do parâmetro 'c'
salvar	Valor lógico. Indica se o resultado será salvo em um arquivo; caso seja TRUE, o arquivo será salvo como "ITENS_FORA_INTERVALO.txt"

**Details**

Utilize este campo para escrever detalhes mais técnicos da sua função (se necessário), ou para detalhar melhor como utilizar determinados argumentos.

**Value**

A função retorna uma data.frame com o número do item no BILOG-MG e o motivo de sua exclusão

**Author(s)**

Alexandre Jaloto

**Examples**

```
adeq.par (arq.par = 'PTLCV3.PAR', a = c (1, 3.5), b = c (-3.5, 3.8), c = c (0, .25))
```

---

aloca.dico

*Alocação de itens dicotômicos nos níveis da escala*


---

**Description**

Alocar itens dicotômicos em pontos ou níveis da escala

**Usage**

```
aloca.dico(
  par,
  met,
  dp = 1,
  int.dp = 0.5,
  int.nivel = c(-2, 2),
  centro = "centro",
  banco,
  min.casos = 50,
  log = TRUE
)
```

## Arguments

<code>par</code>	Objeto com os parâmetros dos itens estimados. Deve possuir três colunas, uma para cada parâmetro (nesta ordem: a, b, c). Só é necessário quando o método for um dos que seguem: b, b65, n65 ou $(2+c)/3$ .
<code>met</code>	O método de alocação. As opções são: b (o parâmetro 'b' é utilizado para alocar o item no nível); b65 (o valor de theta em que a probabilidade de acertar o item vale 0.65); b65emp (o valor de theta em que 65 por cento das pessoas acertaram o item; é a abcissa da intercessão da reta 0.65, com a interpolação linear entre a proporção empírica do nível e a do nível anterior); n65 (o nível para b65); n65emp (o nível em que 65 por cento das pessoas acertaram o item); $(2+c)/3$ (o valor de theta em que a probabilidade de acertar o item vale $(2+c)/3$ )
<code>dp</code>	Desvio padrão da escala transformada
<code>int.dp</code>	Os níveis variam de quanto em quanto, em relação ao dp?
<code>int.nivel</code>	O nível mais baixo e o nível mais alto
<code>centro</code>	'centro' se o nível for centrado no ponto; 'esquerda' se o nível inicia no ponto (fechado na esquerda)
<code>banco</code>	Objeto do tipo <code>data.frame</code> com as respostas (0 = erro; 1 = acerto) e o escore de cada indivíduo. A última variável do banco é o score. Esse objeto é necessário somente se <code>met = 'b65emp'</code> ou <code>met = 'n65emp'</code> . ATENÇÃO: os nomes das colunas dos itens devem começar com 'I' e o da nota deve ser 'NOTA'; para detalhes, veja o exemplo.
<code>min.casos</code>	Número mínimo de respostas em um determinado nível para que o item possa ser alocado nele. Usado somente quando o método for 'b65emp' ou 'n65emp'
<code>log</code>	Argumento lógico. A métrica é logística?

## Details

Para esta função, ainda é necessário pensar sobre as transformações dos parâmetros para alocação dos itens na escala. Por exemplo, se de fato vai para escala (0,1) adequadamente e depois para a escala da avaliação. É possível que haja algum erro nessas transformações.

## Author(s)

Alexandre Jaloto

## Examples

```
set.seed(1000)
par = as.matrix(data.frame(a = rlnorm(30, 0, .5), b = rnorm(30, 0, 1), c = rbeta(30, 5, 17)))
pop = rnorm(5000, 0, 1)

# precisa ser do tipo data.frame (não pode ser data.table nem matrix)
banco = data.frame(simular(pop, par))

calib = mirt::mirt(banco, 1, itemtype = '3PL', TOL = 0.01)
nota = data.frame(mirt::fscores(calib))
names(nota) = 'NOTA'
par = mirt::coef(calib, simplify = TRUE, IRTpars = TRUE)$items[,1:3]
banco$NOTA = nota

# atenção para a forma de nomear
```

```
names (banco) = c(paste0('I', 1:30), 'NOTA')
head (banco)

int.nivel = c(-3, 3)

centro = 'esquerda'
met = 'n65emp'

aloca.dico(par = par, met = met, int.nivel = int.nivel, centro = centro, banco = banco, min.casos = 30)
```

---

aloca.pessoa*Alocação dos indivíduos nos níveis da escala*

---

## Description

Alocar os indivíduos, de acordo com seu escore, nos níveis da escala

## Usage

```
aloca.pessoa(escore, niv = seq(-2, 2, 0.25), direita = TRUE)
```

## Arguments

escore	Vetor com os escores dos indivíduos
niv	Os níveis da escala
direita	argumento lógico. Se TRUE (padrão), é fechado na direita e aberto na esquerda. Se FALSE, é aberto na direita e fechado na esquerda.

## Details

O nível é selecionado de acordo com o escore. O participante é posicionado no nível em que o escore é maior do que o valor mínimo do intervalo e menor ou igual ao maior valor do intervalo, se `direita = TRUE`. Se `direita = FALSE`, o participante é posicionado no nível em que o escore é maior ou igual ao valor mínimo do intervalo e menor do que o maior valor do intervalo.

## Value

A função retorna um objeto do tipo lista com dois elementos:

\$theta: O escore dos indivíduos

\$niveis: O nível de cada indivíduo na escala

## Author(s)

Alexandre Jaloto

## Examples

```
set.seed (12345)
escore = rnorm (100)
aloca.pessoa (escore)
```

aloca.poli

*Alocação de itens politômicos nos níveis da escala***Description**

Alocar os itens com parâmetros estimados pelo MULTILOG nos níveis da escala

**Usage**

```
aloca.poli(
  arq.par,
  categorias,
  p = 0.65,
  m = 0,
  dp = 1,
  int.dp = 0.25,
  int.nivel = c(-2, 2),
  met = 3,
  nomes.cat = c()
)
```

**Arguments**

arq.par	Arquivo PAR gerado, pelo MULTILOG, com os parâmetros dos itens estimados
categorias	O maior quantitativo de categorias de um mesmo item
p	A probabilidade de obter determinado código (categoria) utilizada para alocar o item em um nível
m	Média da escala transformada (o padrão da função é não transformar)
dp	Desvio padrão da escala transformada (o padrão da função é não transformar)
int.dp	Os níveis variam de quanto em quanto, em relação ao dp?
int.nivel	O nível mais baixo e o nível mais alto
met	O método de alocação, podendo ser 1 ou 3
nomes.cat	objeto do tipo <code>data.frame</code> ou <code>matrix</code> indicando o nome de cada categoria de cada item (nesta <code>data.frame</code> , se a categoria não existir em determinado item, indicar NA)

**Details**

Para esta função, ainda é necessário pensar sobre as transformações dos parâmetros para alocação dos itens na escala. Por exemplo, se de fato vai para a escala (0,1) adequadamente e depois para a escala da avaliação. É possível que haja algum erro nessas transformações.

Se `met = 1`, o posicionamento do item é feito da seguinte maneira: calcula-se a proficiência que o participante deveria ter, dada a probabilidade de 0,65 de ele marcar aquela categoria ou mais alta

$$\theta = (-1/a) * \ln((1 - p)/1) + b$$

onde  $a$  é a discriminação,  $p$  é a probabilidade de marcar a categoria ou superior (no caso, 0,65) e  $b$  é o parâmetro de dificuldade.



Se  $met = 3$ , o posicionamento do item é feito da seguinte maneira: calcula-se a probabilidade de uma categoria ser selecionada dado que o  $\theta$  é igual ao nível. Seleciona-se o nível em que há 65% de probabilidade ou mais de um estudante com proficiência naquele nível marcar uma categoria ou mais alta.

### Value

A função retorna um objeto do tipo lista com três elementos:

\$prof: O escore que retorna a probabilidade 'p' do indivíduo selecionar determinada categoria

\$níveis: O nível de cada categoria do item

\$MaiorProb: A categoria mais provável de ser selecionada em cada nível

### Author(s)

Alexandre Jaloto

### Examples

```
arq.par = 'Rodada6.PAR'
categorias = 5
p = .65
m = 0
dp = 1
int.dp = .25
int.nivel = c (-2, 2)
met = 3
nomes.cat = data.frame (a = paste0 ('a', 1:16), b = paste0 ('b', 1:16),
                        c = paste0 ('c', 1:16), d = paste0 ('d', 1:16),
                        e = paste0 ('e', 1:16))
aloca.poli (arq.par, categorias, p = .65, m = 0,
            dp = 1, int.dp = .25, int.nivel = c (-2, 2),
            met = 3, nomes.cat = nomes.cat)
```

---

arred

*Arredondamento de números*

---

### Description

Arredondar valores numéricos de acordo com a norma 5891:2014 da ABNT. O mesmo critério é adotado por programas como Excel e SPSS.

### Usage

```
arred(x, n)
```

### Arguments

x	Valor a ser arredondado
n	Quantidade de casas decimais

**Details**

O programa multiplica  $x$  por  $10^n$ , trunca esse número e diminui de  $x * 10^n$ . Se essa diferença for maior ou igual a 0.5, então arredonda para cima; se essa diferença for menor do que 0.5, então arredonda para baixo.

ATENÇÃO: devido à forma como o número binário é convertido em número real, é aconselhável utilizar o comando `options(digits = 16)`. O padrão para essa opção é 7.

**Value**

A função retorna o valor arredondado de acordo com o critério determinado.

**Author(s)**

Alexandre Jaloto

**Examples**

```
x = 100.123456
n = 5
arred(x, n)
options(digits=7)
arred(x, n)
options(digits=16)
arred(x, n)
options(digits=22)
arred(x, n)
```

---

brincar

*Brincando com a TRI*

---

**Description**

Gera um aplicativo shiny com o objetivo de conhecer melhor a TRI

**Usage**

```
brincar()
```

**Author(s)**

Alexandre Jaloto

**Examples**

```
brincar()
```

---

`cci`*Curva característica do item*

---

**Description**

Produz a curva característica de um item (CCI)

**Usage**

```
cci(  
  a = 1.2,  
  b = 0,  
  c = 0.2,  
  theta = seq(-4, 4, 0.01),  
  info = FALSE,  
  xlab = "Proficiência (habilidade)",  
  ylab = "Probabilidade de acerto",  
  ...  
)
```

**Arguments**

<code>a</code>	valor do parâmetro a
<code>b</code>	valor do parâmetro b
<code>c</code>	valor do parâmetro c
<code>theta</code>	vetor com os valores de traço latente para a construção do gráfico
<code>info</code>	Valor lógico. Se TRUE, plota também a curva de informação do item
<code>...</code>	Outros argumentos das funções plot e lines

**Value**

A função retorna um gráfico com a curva característica do item

**Author(s)**

Alexandre Jaloto

**Examples**

```
# em uma escala (0,1)  
a = 1.5  
b = 0.3  
c = .15  
  
cci (a, b, c)  
cci (a, b, c, info = TRUE)  
  
# agora em uma escala (500, 100)  
a = .01  
b = 505  
c = .15
```

```
theta = seq (100, 900, 10)
cci (a, b, c, theta)

# alterando parâmtros do gráfico
cci (a, b, c, theta, col = 'red', lty = 2, main = 'CCI')
```

dif.bilog

*Análise de DIF para o BILOG-MG*

## Description

Verifica existência de DIF e a qualidade do ajuste de um item calibrado no BILOG-MG

## Usage

```
dif.bilog(
  arq.exp,
  arq.sco,
  perc = list(5, 95),
  dif.dif = 0.15,
  grupo = 7,
  int.comum = TRUE,
  salvar = FALSE
)
```

## Arguments

arq.exp	arquivo .EXP gerado pelo BILOG-MG
arq.sco	arquivo .SCO gerado pelo BILOG-MG
perc	os percentis que definirão o intervalo de análise para detecção de DIF
dif.dif	a diferença máxima tolerada entre a proporção de acerto observada e a esperada segundo o modelo
grupo	o grupo, no arquivo .BLM, que está em análise; ou seja, o grupo focal corresponde a qual grupo no arquivo .BLM?
int.comum	valor lógico: se TRUE, o intervalo a ser analisado é o compreendido entre o maior percentil inferior e o menor percentil superior.
salvar	argumento lógico que indica se a saída será salva em dois arquivos (um para DIF e outro para ajuste do modelo)

## Details

Para analisar DIF, a função diminui a proporção de acerto do grupo focal da proporção de acerto do grupo de referência. Para verificar a qualidade do ajuste do item, a função diminui a probabilidade de acerto segundo o modelo da proporção de acerto observada.

Sobre o argumento int.comum: supondo que o intervalo escolhido foi perc = c (.05, .95). A análise de DIF se dará para o intervalo compreendido entre o maior P5 e o menor P95 dos dois grupos.

O que a função faz? Ela lê o arquivo expect; seleciona os itens comuns; seleciona o grupo de interesse e os outros grupos; compara as duas proporções esperadas de resposta correta; seleciona o grupo de interesse; compara o teórico com a proporção esperada do grupo (para análise de ajuste)

ATENÇÃO: é necessário que o início do arquivo SCO tenha duas linhas que não são referentes a dados

### Value

Uma lista com três elementos:

\$DIF Contém os itens que apresentaram DIF entre o grupo focal e algum outro grupo do BILOG-MG.

\$INTERVALOS Os intervalos em que ocorreram as análises. P5 e P95 é o intervalo compreendido entre o maior P5 e o menor P95 de cada análise entre dois grupos. LIMITE\_INF e LIMITE\_SUP compreende o limite em que a análise de fato ocorreu, ou seja, os pontos de quadratura no interior do intervalo do P% e P95.

\$AJUSTE contém os itens que apresentaram problemas de ajuste.

### Author(s)

Alexandre Jaloto

---

dif.mirt

*Análise de DIF para o mirt*

---

### Description

Verifica existência de DIF e a qualidade do ajuste de um item calibrado no mirt

### Usage

```
dif.mirt(
  fit.antigo,
  fit.atual,
  comuns.antigo = NULL,
  comuns.atual = NULL,
  int.teta = c(-6, 6),
  n.qdpt = 61
)
```

### Arguments

fit.antigo	objeto mirt do tipo SingleGroupClass referente ao grupo em que os itens foram calibrados (grupo de referência).
fit.atual	objeto mirt do tipo SingleGroupClass referente ao grupo da aplicação em voga (grupo focal).
comuns.antigo	vetor com nomes dos itens comuns no banco antigo. Os nomes dos itens podem ser obtidos com <code>mirt::extract.mirt(fit.antigo, 'itemnames')</code> . Só é necessário se os nomes dos itens forem diferentes nos dois bancos.
comuns.atual	vetor com nomes dos itens comuns no banco atual. Os nomes dos itens podem ser obtidos com <code>mirt::extract.mirt(fit.atual, 'itemnames')</code> . Só é necessário se os nomes dos itens forem diferentes nos dois bancos.
n.qdpt	quantidade de pontos de quadratura.
int.qdpt	intervalo dos pontos de quadratura. Esse intervalo será usado para a análise de MaxADif. O padrão é <code>c(-6, 6)</code> .

**Value**

A função retorna uma lista com quatro elementos

- info lista com os percentis dos grupos antigo e atual, e os pontos de quadratura utilizados
- itens vetor com os nomes dos itens que apresentaram DIF. Se os nomes nos bancos forem diferentes, os nomes do banco atual são apresentados
- maxadif data.frame com os itens que apresentaram DIF pelo método MaxADIF, com o nome do item, o ponto de quadratura em que a proporção superou 0.15 e as diferenças em cada categoria de resposta
- regressao lista com os itens que apresentaram DIF uniforme, não uniforme e misto com o método de regressão logística
- rmsd data.frame com os itens que apresentaram DIF pelo método RMSD, com o nome do item e os valores de RMSD para cada categoria de resposta

**Author(s)**

Alexandre Jaloto

**Examples**

```
set.seed(1234)

a <- rlnorm(60)
d <- rnorm(60)
data.atual <- data.frame(mirt::simdata(a, d, 1000, '2PL'))
data.antigo <- data.frame(mirt::simdata(a, d, 1000, '2PL'))

names(data.antigo) <- c(paste0('IME_', 1:50), paste0('IRC_', 51:60))
names(data.atual) <- c(paste0('IME_', 61:85), paste0('IME_', 1:25), paste0('IRC_', 51:60))

for(i in 51:60)
{
  data.antigo[,i] <- sample(3, 1000, TRUE)
  data.atual[,i] <- sample(3, 1000, TRUE)
}
fit.antigo <- mirt::mirt(data.antigo, 1, TOL = .01)
fit.atual <- mirt::mirt(data.atual, 1, TOL = .01)
dif <- dif.mirt(fit.antigo, fit.atual)

# para itens com nomes diferentes
names(data.antigo) <- paste0('I', 1:ncol(data.antigo))

comuns.antigo <- c(paste0('I', 1:25), paste0('I', 51:60))
comuns.atual <- c(paste0('IME_', 1:25), paste0('IRC_', 51:60))

fit.antigo <- mirt::mirt(data.antigo, 1, TOL = .01)
fit.atual <- mirt::mirt(data.atual, 1, TOL = .01)

dif <- dif.mirt(fit.antigo, fit.atual, comuns.antigo = comuns.antigo, comuns.atual = comuns.atual)
```

---

freq.nivel	<i>Frequência dos indivíduos nos níveis da escala</i>
------------	---

---

**Description**

Verificar a frequência dos indivíduos em cada nível da escala

**Usage**

```
freq.nivel(escore, peso = 1, niv = seq(-2, 2, 0.25), met = 3)
```

**Arguments**

escore	Vetor com os escores
peso	Vetor com o peso de cada escore
niv	Os níveis da escala
met	O método de alocação, segundo o relatório de 2014 da ANA

**Details**

Detalhes

**Value**

A função retorna um objeto do tipo data.frame com duas variáveis: o nível da escala e a frequência relativa dos indivíduos.

**Author(s)**

Alexandre Jaloto

**Examples**

```
set.seed(1000)
escore = rnorm(100)
freq.nivel(escore)
```

---

gera.caderno	<i>Gerar os cadernos de prova</i>
--------------	-----------------------------------

---

**Description**

Gerar objeto com a estrutura correspondente à composição de cada caderno de cada disciplina / área

**Usage**

```
gera.caderno(itens, bib, disc.cad = 2)
```

**Arguments**

itens	O(s) objeto(s) com os itens da(s) disciplina(s) que compõe(m) os cadernos no BIB (tem que ser todos, por enquanto; depois tem que melhorar a função)
bib	Objeto com o BIB
disc.cad	Quantidade de disciplinas em cada caderno (padrão: 2)

**Details**

Utilize este campo para escrever detalhes mais técnicos da sua função (se necessário), ou para detalhar melhor como utilizar determinados argumentos.

**Value**

O que a função retorna?

**Author(s)**

Alexandre Jaloto

**Examples**

```
set.seed(1000)
gab.lc = sample (LETTERS[1:4], 9, replace = TRUE)
gab.mt = sample (LETTERS[1:4], 9, replace = TRUE)
itens.lc = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Item = sample (12345:54321, 9), Origem = 'NOVO',
                      Gabarito = gab.lc, Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'LC')
itens.mt = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Gabarito = gab.mt, Item = sample (12345:54321, 9),
                      Origem = 'NOVO', Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'MT')

bib = data.frame (Caderno = 1:3, Disciplina1 = rep ('LC', 3), Disciplina2 = rep ('MT', 3),
                 Bloco1 = 1:3, Bloco2 = c(2, 3, 1), Bloco3 = 1:3, Bloco4 = c(2, 3, 1))
itens = rbind (itens.lc, itens.mt)
cadernos = gera.caderno (itens, bib)
```

---

gera.form

*Gera os forms e os gabaritos para o BILOG-MG*

---

**Description**

A partir das informações do BIB e dos itens, esta função gera dois arquivos 'txt' para serem incorporados aos arquivos do BILOG-MG. São os forms, que devem ser adicionados no BLM, e os gabaritos.

**Usage**

```
gera.form(itens, bib, disc.cad = 2, cad.1 = c(1, 1), carac = 34)
```



**Arguments**

itens	O(s) objeto(s) com os itens da(s) disciplina(s) que compõe(m) os cadernos no BIB (tem que ser todos, por enquanto; depois tem que melhorar a função). ATENÇÃO: precisa ser data.frame
bib	Objeto com o BIB. ATENÇÃO: precisa ser data.frame
disc.cad	Quantidade de disciplinas em cada caderno
cad.1	Número do primeiro form, no BILOG-MG, referente aos cadernos de cada disciplina
carac	Quantidade de caracteres antes da leitura do vetor de resposta no BILOG-MG

**Value**

Para cada disciplina, dois arquivos 'txt':

1. arquivo com os forms para o BILOG-MG
2. arquivo com os gabaritos dos forms para o BILOG-MG

**Author(s)**

Alexandre Jaloto

**Examples**

```
set.seed(1000)
gab.lc = sample (LETTERS[1:4], 9, replace = TRUE)
gab.mt = sample (LETTERS[1:4], 9, replace = TRUE)
itens.lc = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Item = sample (12345:54321, 9), Origem = 'NOVO',
                      Gabarito = gab.lc, Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'LC')
itens.mt = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Gabarito = gab.mt, Item = sample (12345:54321, 9),
                      Origem = 'NOVO', Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'MT')

bib = data.frame (Caderno = 1:3, Disciplina1 = rep ('LC', 3), Disciplina2 = rep ('MT', 3),
                 Bloco1 = 1:3, Bloco2 = c(2, 3, 1), Bloco3 = 1:3, Bloco4 = c(2, 3, 1))
itens = rbind (itens.lc, itens.mt)

gera.form (itens, bib, cad.1 = c (12, 14))
```

---

info.teste

---

*Curva de informação do teste*


---

**Description**

Produz a(s) curva(s) de informação do(s) teste(s)

**Usage**

```
info.teste(
  pars = list(),
  testes = list(),
  theta = seq(-4, 4, 0.01),
  erro = FALSE,
  ajustar = TRUE
)
```

**Arguments**

<code>pars</code>	Objeto do tipo <code>data.frame</code> , <code>matrix</code> ou uma lista com <code>data.frame</code> e/ou <code>matrix</code> . Este objeto contém os parâmetros dos itens de cada instrumento. Se o objetivo é plotar a curva de apenas um instrumento, o objeto deve ser uma <code>data.frame</code> ou <code>matrix</code> ; se a curva de mais de um instrumento for plotada, o objeto deve ser uma lista em que cada elemento é uma <code>data.frame</code> ou <code>matrix</code> com os parâmetros. A primeira coluna do objeto (ou elemento, em caso de ser uma lista) contém os valores do parâmetro 'a'; a segunda, os do parâmetro 'b'; a terceira, os do parâmetro 'c'
<code>testes</code>	Vetor com os nomes dos testes, para inserir a legenda no gráfico
<code>theta</code>	Vetor com os valores de traço latente para a construção do gráfico
<code>erro</code>	Valor lógico. Se <code>TRUE</code> , plota também a curva de erro da medida
<code>ajustar</code>	Valor lógico. Se <code>TRUE</code> , plota todas as curvas de informação na mesma escala. Se <code>FALSE</code> , as curvas serão plotadas em escalas diferentes no que diz respeito ao eixo 'y'.

**Value**

A função retorna um gráfico com a curva de informação de cada teste.

**Author(s)**

Alexandre Jaloto

**Examples**

```
# criar objeto com os parâmetros
set.seed(1000)
pars1 = data.frame (a = runif (50, .7, 1.3), b = runif (50, -3, 3), c = runif (50, 0, 1))
pars2 = data.frame (a = runif (50, .7, 1.3), b = runif (50, -3, 3), c = runif (50, 0, 1))
pars3 = data.frame (a = runif (50, .7, 1.3), b = runif (50, -3, 3), c = runif (50, 0, 1))

pars = list (pars1, pars2, pars3)
testes = c ('T1', 'T2', 'T3')
theta = seq (-3, 3, .5)

info.teste (pars = pars, theta = theta, testes = testes, erro = FALSE, ajustar = TRUE)

info.teste (pars = pars[[1]], theta = theta, testes = testes[1], erro = TRUE)
```

---

ler.exp	<i>Importação do arquivo EXPECT</i>
---------	-------------------------------------

---

**Description**

Importar o arquivo .EXP produzido pelo BILOG-MG

**Usage**

```
ler.exp(arq.exp)
```

**Arguments**

arq.exp	Arquivo .EXP
---------	--------------

**Details**

Utilize este campo para escrever detalhes mais técnicos da sua função (se necessário), ou para detalhar melhor como utilizar determinados argumentos.

**Value**

A função retorna um objeto do tipo data.frame com os dados do arquivo .EXP

**Author(s)**

Alexandre Jaloto

**Examples**

```
ler.exp (arq.exp = 'PTLCV3.EXP')
```

---

ler.par	<i>Importação do arquivo PAR</i>
---------	----------------------------------

---

**Description**

Importar o arquivo .PAR produzido pelo BILOG-MG ou MULTILOG

**Usage**

```
ler.par(arq.par, prog = "BLM", categorias)
```

**Arguments**

arq.par	Arquivo .PAR
prog	O programa que produziu o arquivo .PAR. Use 'BLM' (padrão) para BILOG-MG e 'MLM' para MULTILOG.
categorias	Somente se prog = 'MLM'. O número máximo de categorias de um item.

**Details**

Utilize este campo para escrever detalhes mais técnicos da sua função (se necessário), ou para detalhar melhor como utilizar determinados argumentos.

**Value**

A função retorna um objeto do tipo `data.frame` com os dados do arquivo `.PAR`

**Author(s)**

Alexandre Jaloto

**Examples**

```
ler.par ('PTLCV3.PAR')
ler.par ('ANA16.PAR', prog = 'MLM', categorias = 4)
```

---

ler.sco	<i>Importação do arquivo de escore</i>
---------	--

---

**Description**

Importar o arquivo de escore do BILOG-MG ou do MULTILOG

**Usage**

```
ler.sco(arq.sco, prog = "BLM")
```

**Arguments**

arq.sco	Arquivo SCO gerado pelo BILOG-MG ou pelo MULTILOG
prog	Programa que gerou a base de dados ('BLM' para BILOG-MG e 'MLM' para MULTILOG)

**Details**

Utilize este campo para escrever detalhes mais técnicos da sua função (se necessário), ou para detalhar melhor como utilizar determinados argumentos.

**Value**

A função retorna uma `data.frame` com os escores dos indivíduos

**Author(s)**

Alexandre Jaloto

**Examples**

```
ler.sco (arq.sco = 'PTLCV3.SCO', prog = 'BLM')
```

---

`prop.exp`*Proporções esperadas*

---

**Description**

"Abre" o objeto gerado pela função `ler.exp`. Ou seja, verifica as proporções empíricas e do modelo de acerto do item em cada ponto de quadratura.

**Usage**

```
prop.exp(dados)
```

**Arguments**

`dados`                      objeto gerado pela função `ler.exp`

**Value**

data.frame com o nome do item no BILOG-MG, o grupo em que foi aplicado, a variável analisada (PROPORÇÃO ou PROPORÇÃO DO MODELO), e a proporção em cada ponto de quadratura.

**Author(s)**

Alexandre Jaloto

---

`relatorio.tct`*Relatório TCT*

---

**Description**

Elabora um relatório com informações das análises psicométricas segundo a Teoria Clássica dos Testes (TCT)

**Usage**

```
relatorio.tct(  
  disc,  
  disc.extenso,  
  teste,  
  n.itens.comuns,  
  n.itens.novos,  
  n.alt,  
  tct  
)
```

**Arguments**

<code>disc</code>	Sigla da disciplina / área
<code>disc.extenso</code>	Nome da disciplina / área por extenso
<code>teste</code>	Nome do teste
<code>n.itens.comuns</code>	Quantidade de itens comuns
<code>n.itens.novos</code>	Quantidade de itens novos
<code>n.alt</code>	Número máximo de alternativas dos itens
<code>tct</code>	Objeto com os resultados da análise clássica (no mesmo padrão do objeto que a função <code>tct</code> retorna). Importante que o nome das variáveis deste objeto seja no mesmo padrão do objeto que a função <code>tct</code> retorna

**Value**

A função retorna um arquivo HTML com o relatório das análises psicométricas segundo a Teoria Clássica dos Testes (TCT).

**Author(s)**

Alexandre Jaloto

---

<code>simular</code>	<i>Simular padrões de resposta</i>
----------------------	------------------------------------

---

**Description**

Simular padrões de resposta para itens dicotômicos segundo a Teoria de Resposta ao Item (modelos logístico e normal). Verifique se a função `simdata` do pacote `mirt` contempla suas necessidades, pois ela é mais rápida.

**Usage**

```
simular(theta = seq(-3, 3, by = 0.1), pars, mod = "log")
```

**Arguments**

<code>theta</code>	Vetor com as medidas de traço latente dos indivíduos
<code>pars</code>	Objeto do tipo <code>data.frame</code> ou <code>matrix</code> com os parâmetros dos itens. A primeira coluna deve apresentar os valores do parâmetro de discriminação; a segunda, os valores de dificuldade (posição); a terceira, os valores de acerto casual ( <i>pseudochute</i> ). Para modelos de dois parâmetros ou um, veja a seção <i>Details</i> .
<code>mod</code>	O modelo da distribuição de probabilidade. Use <code>"log"</code> para logístico e <code>"norm"</code> para normal

**Details**

A simulação requer itens de três parâmetros. Para realizar simulação de respostas a itens de dois parâmetros, arbitre o valor do acerto casual para 0. Para realizar a simulação de respostas a itens de um parâmetro, considere o acerto casual como 0 e a discriminação como 1.

**Value**

A função retorna um objeto do tipo matrix que contém a probabilidade de cada indivíduo acertar cada item. As linhas correspondem aos indivíduos e as colunas, aos itens.

**Author(s)**

Alexandre Jaloto

**Examples**

```
set.seed(1000)
theta = rnorm (50, 0, 1)
pars = matrix ( c (runif (20, .5, 2), rnorm (20, 0, 1),
                  runif (20, .05, .4)), nrow = 20, ncol = 3)
simular (theta = theta, pars = pars)
```

tct

*Análise via Teoria Clássica dos Testes (TCT)***Description**

Análise psicométrica de itens por meio da TCT

**Usage**

```
tct(
  banco.aberto,
  gab.aberto,
  alt = c("A", "B", "C", "D", ".", "*"),
  usa.normit = TRUE,
  met.perc = 6,
  pop = FALSE
)
```

**Arguments**

banco.aberto	Objeto do tipo data.frame ou matrix cuja primeira variável é o número do caderno e as demais variáveis são as respostas a cada item; é necessário que o banco esteja aberto e os cadernos estejam ordenados a partir do 1; importante: o banco já tem que ser somente da disciplina que será analisada
gab.aberto	Objeto do tipo data.frame com duas variáveis: código do item e gabarito; é necessário que a ordem dos itens seja a mesma da ordem do objeto banco.aberto
alt	As alternativas possíveis em cada item
usa.normit	Valor lógico que indica se o escore utilizado para a análise é o normit (TRUE) ou a soma de acertos (FALSE)
met.perc	O método utilizado para o cálculo do percentil. Varia de 1 a 9. Para mais informações, verifique ajuda da função quantile.
pop	TRUE se for população, FALSE (padrão) se for uma amostra. Essa escolha interfere nas contas que envolvem o cálculo da variância ou do desvio padrão.

## Details

A análise utiliza análise via normit. Para os cálculos que envolvem o desvio padrão, considera-se a raiz da variância da população; a função var considera  $n-1$ , assim como a função sd.

## Value

A função retorna um objeto do tipo list com os seguintes elementos:

\$tct Dados dos itens e da análise, quais sejam: Número sequencial do item; Código do item; Gabarito do item; Índice de dificuldade; Índice de discriminação; Porcentagem de acerto no grupo inferior; Porcentagem de acerto no grupo superior; Correlação bisserial; Proporção de escolha de cada alternativa; Correlação bisserial de cada alternativa.

\$normit Dados dos indivíduos, quais sejam: Caderno apresentado; Resposta a cada item (banco aberto); Soma de acertos; Normit.

## Author(s)

Alexandre Jaloto

## Examples

```
# criar um banco aberto
set.seed(1000)
gab.lc = sample (LETTERS[1:4], 9, replace = TRUE)
gab.mt = sample (LETTERS[1:4], 9, replace = TRUE)
itens.lc = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Item = sample (12345:54321, 9), Origem = 'NOVO',
                      Gabarito = gab.lc, Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'LC')
itens.mt = data.frame (Bloco = rep (1:3, c (3,3,3)), Posicao = rep (1:3, 3),
                      Gabarito = gab.mt, Item = sample (12345:54321, 9),
                      Origem = 'NOVO', Num_bilog = 201:209,
                      Nome_bilog = paste ('P', 0001:0009, sep = ''), Disciplina = 'MT')

bib = data.frame (Caderno = 1:3, Disciplina1 = rep ('LC', 3), Disciplina2 = rep ('MT', 3),
                 Bloco1 = 1:3, Bloco2 = c(2, 3, 1), Bloco3 = 1:3, Bloco4 = c(2, 3, 1))
itens = rbind (itens.lc, itens.mt)
resp = matrix (sample (LETTERS[1:4], 180, replace = TRUE), ncol = 6)
banco = data.frame (CAD = seq(1:3), resp)
disc = 'LC'
aberto = abre.banco (banco = banco, itens = itens, bib = bib, disc = disc)

tct = tct (banco.aberto = aberto$respostas, gab.aberto = aberto$gabarito)
```



# Index

abre.banco, [2](#)  
abre.resp, [3](#)  
adeq.par, [4](#)  
aloca.dico, [5](#)  
aloca.pessoa, [7](#)  
aloca.poli, [8](#)  
arred, [9](#)  
  
brincar, [10](#)  
  
cci, [11](#)  
  
dif.bilog, [12](#)  
dif.mirt, [13](#)  
  
freq.nivel, [15](#)  
  
gera.caderno, [15](#)  
gera.form, [16](#)  
  
INEPsico (INEPsico-package), [2](#)  
INEPsico-package, [2](#)  
info.teste, [17](#)  
  
ler.exp, [19](#)  
ler.par, [19](#)  
ler.sco, [20](#)  
  
prop.exp, [21](#)  
  
relatorio.tct, [21](#)  
  
simular, [22](#)  
  
tct, [23](#)