Package: bigIRT (via r-universe)

October 9, 2024

Title Fits item response theory models to big data

Version 0.1.8

Description Fits item response theory models to big data.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 3.5.0), data.table

Imports data.table, methods, mize, parallel, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.21.0), statmod, rstantools (>= 2.3.0)

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.21.1), StanHeaders (>= 2.18.0)

Suggests testthat

Biarch true

SystemRequirements GNU make

Repository https://cdriveraus.r-universe.dev

RemoteUrl https://github.com/cdriveraus/bigIRT

RemoteRef HEAD

RemoteSha d6072a8ff9ed897f8866de18aa8d3d1e63f3d72b

Contents

---

bigIRT-package *The 'bigIRT' package.*

Description

A DESCRIPTION OF THE PACKAGE

Author(s)

Maintainer: Charles Driver <charles.driver@ife.uzh.ch> (ORCID)

References

Stan Development Team (NA). RStan: the R interface to Stan. R package version 2.26.1. https://mc-stan.org

Examples

```
#Generate some data (here 2pl model
require(data.table)
dat <- simIRT(Nsubs = 5000,Nitems = 100,Nscales = 1,
  logitCMean = -10,logitCSD = 0,AMean = 1,ASD = .3,
  BMean=0,BSD = .5,
  AbilityMean = 0,AbilitySD = 1)

#convert to wide for TAM
wdat <- data.frame(dcast(data.table(dat$dat),formula = 'id ~ Item',value.var='score')[,-1])


#fit using TAM
require(TAM)
tfit <-tam.mml.2pl(resp = wdat,est.variance = TRUE)


#fit using bigIRT
fit <- fitIRT(dat$dat,cores=2,pl=2)

#some summary stuff:
plot(dat$Ability,(fit$pars$Ability-dat$Ability)^2) #ability error given ability
sqrt(mean((fit$pars$Ability-dat$Ability)^2)) #rms error stat

#correlations of estimated vs true
cor(data.frame(True=dat$Ability,Est=fit$pars$Ability))
cor(data.frame(True=dat$A,Est=fit$pars$A))
cor(data.frame(True=dat$B,Est=fit$pars$B))
```

`dropPerfectScores` *Drop subjects and items with all perfect scores*

Description

This function drops variables/items and subjects that have all perfect scores (either all 0's or all 1's) in a data table.

Usage

```
dropPerfectScores(
  dat,
  scoreref. = "score",
  itemref. = "Item",
  idref. = "id",
  tol. = 0.001
)
```

Arguments

`dat`	The input data table
`scoreref`	The column name of the score variable in `dat`
`itemref`	The column name of the item variable in `dat`
`idref`	The column name of the id variable in `dat`
`tol`	Tolerance level for checking perfect scores – .01 would drop subjects with less than 1% correct or incorrect

Value

The input data table (`dat`) without variables/items and subjects with all perfect scores.

Examples

```
dat <- data.table(id=c(1,1,1,2,2,2,3,3,3), Item=c('I1','I2','I3','I1','I2','I3','I1','I2','I3'),
   score=c(1,0,1,0,0,0,0,1,1))
print(dropPerfectScores(dat))
```

`fitIRT` *Fit a binary Item Response Theory (IRT) model*

Description

This function fits a binary Item Response Theory (IRT) model using various parameters and options.

Usage

```
fitIRT(
  dat,
  score = "score",
  id = "id",
  item = "Item",
  scale = "Scale",
  pl = 1,
  personDat = NA,
  personPreds = character(),
  itemDat = NA,
  AitemPreds = character(),
  BitemPreds = character(),
  CitemPreds = character(),
  DitemPreds = character(),
  itemSpecificBetas = FALSE,
  betaScale = 10,
  invspAMeandat = 0.542,
  invspASD = 2,
  BMeandat = 0,
  BSD = 10,
  logitCMeandat = -4,
  logitCSD = 2,
  logitDMeandat = 4,
  logitDSD = 2,
  AbilityMeandat = array(0, dim = c(length(unique(dat[[scale]])))),
  AbilitySD = array(10, dim = c(length(unique(dat[[scale]])))),
  AbilityCorr = diag(1, c(length(unique(dat[[scale]])))),
  AMeanSD = 1,
  BMeanSD = BSD,
  logitCMeanSD = logitCSD,
  logitDMeanSD = logitDSD,
  AbilityMeanSD = array(1, dim = c(length(unique(dat[[scale]])))),
  iter = 2000,
  cores = 6,
  carefulfit = FALSE,
  ebayes = TRUE,
  ebayesmultiplier = 2,
  ebayesFromFixed = FALSE,
```

```
  ebayesiter = 1,
  estMeans = c("ability", "B", "C", "D"),
  priors = TRUE,
  integrateEachAbility = FALSE,
  integrateEachAbilityFixedSE = FALSE,
  mml = FALSE,
  NintegratePoints = 5,
  normalise = FALSE,
  normaliseScale = 1,
  normaliseMean = 0,
  dropPerfectScores = TRUE,
  trainingRows = 1:nrow(dat),
  init = NA,
  tol = 1e-08 * 10^(log(nrow(dat), 10)),
  ...
)
```

Arguments

dat	A data frame containing the data to be analyzed.
score	Character. The name of the column in dat representing the response. Default is 'score'.
id	Character. The name of the column in dat representing the individual. Default is 'id'.
item	Character. The name of the column in dat representing the item. Default is 'Item'.
scale	Character. The name of the column in dat representing the scale of each item. Default is 'Scale'.
pl	Integer. The number of parameters for the logistic model (1PL, 2PL, 3PL, or 4PL). Default is 1.
personDat	Data frame. Any fixed ability data for persons. Default is NA.
personPreds	Character vector. Names of predictors for person parameters found in data. Default is an empty character vector.
itemDat	Data frame. Any fixed item data. Default is NA.
AitemPreds	Character vector. Names of predictors for item discrimination parameters. Default is an empty character vector.
BitemPreds	Character vector. Names of predictors for item difficulty parameters. Default is an empty character vector.
CitemPreds	Character vector. Names of predictors for item guessing parameters. Default is an empty character vector.
DitemPreds	Character vector. Names of predictors for item upper asymptote (slipping) parameters. Default is an empty character vector.
itemSpecificBetas	Logical. Whether to allow item-specific betas for covariate effects, or simply estimate one effect per covariate. Default is FALSE.

betaScale	Numeric. Scale of the prior for beta parameters. Default is 10.
invspAMeandat	Numeric. Mean for the prior distribution of the raw discrimination parameters, which subsequently have a 'softplus' log(1+exp(x)) applied. Default is 0.542, giving a mean for A pars of ~ 1.
invspASD	Numeric. Standard deviation for the prior distribution of the raw discrimination parameters. Default is 2.
BMeandat	Numeric. Mean for the prior distribution of the item difficulty parameters. Default is 0.
BSD	Numeric. Standard deviation for the prior distribution of the item difficulty parameters. Default is 10.
logitCMeandat	Numeric. Mean for the prior distribution of the item guessing parameters (on logit scale). Default is -4.
logitCSD	Numeric. Standard deviation for the prior distribution of the item guessing parameters (on logit scale). Default is 2.
logitDMeandat	Numeric. Mean for the prior distribution of the item upper asymptote parameters (on logit scale). Default is 4.
logitDSD	Numeric. Standard deviation for the prior distribution of the item upper asymptote parameters (on logit scale). Default is 2.
AbilityMeandat	Numeric array. Mean for the prior distribution of the ability parameters. Default is 0 for each scale.
AbilitySD	Numeric array. Standard deviation for the prior distribution of the ability parameters. Default is 10 for each scale.
AbilityCorr	Matrix. Correlation matrix for the ability parameters. Default is an identity matrix.
AMeanSD	Numeric. Standard deviation for the prior distribution of the discrimination parameters. Default is 1.
BMeanSD	Numeric. Standard deviation for the prior distribution of the difficulty parameters. Default is BSD.
logitCMeanSD	Numeric. Standard deviation for the prior distribution of the guessing parameters (on logit scale). Default is logitCSD.
logitDMeanSD	Numeric. Standard deviation for the prior distribution of the upper asymptote parameters (on logit scale). Default is logitDSD.
AbilityMeanSD	Numeric array. Standard deviation for the prior distribution of the ability parameters. Default is 1 for each scale.
iter	Integer. Maximum number of iterations for the fitting algorithm. Default is 2000.
cores	Integer. Number of cores to use for parallel computation. Default is 6.
carefulfit	Logical. Whether to use a slower, careful fitting procedure. Default is FALSE. Experimental.
ebayes	Logical. Whether to use empirical Bayes estimation. Default is TRUE. With ebayes, the priors are adapted based on a first pass estimate.

ebayesmultiplier
: Numeric. Multiplier for the widths of the empirical Bayes priors. Default is 2, as this appears to work better in practice.

ebayesFromFixed
: Logical. Whether to initialize empirical Bayes from any specifed fixed values Default is FALSE.

ebayesiter
: Integer. Number of iterations for empirical Bayes estimation. Default is 1.

estMeans
: Character vector. Which means to estimate from 'ability', 'A', 'B', 'C', 'D'. Default is c('ability', 'B', 'C', 'D'), with discrimination means fixed.

priors
: Logical. Whether to use prior distributions. Default is TRUE.

integrateEachAbility
: Logical. Whether to integrate across each ability. Default is FALSE.

integrateEachAbilityFixedSE
: Logical. Whether to integrate each ability with fixed standard error. Default is FALSE.

mml
: Logical. Experimental and not working well. Whether to use marginal maximum likelihood estimation. Default is FALSE.

NintegratePoints
: Integer. Number of integration points for numerical integration. Default is 5.

normalise
: Logical. Whether to normalize the output estimates. Default is FALSE.

normaliseScale
: Numeric. Scale for normalization. Default is 1.

normaliseMean
: Numeric. Mean for normalization. Default is 0.

dropPerfectScores
: Logical. Whether to drop perfect scores from each subject and item before estimation. Default is TRUE.

trainingRows
: Integer vector. Rows of data to use for estimation of parameters. Default is all rows in dat.

init
: Initial values for the fitting algorithm. Default is NA.

tol
: Numeric. Tolerance for convergence. Default attempts to sensibly adjust for amount of data.

...
: Additional arguments passed to the fitting function.

Value

A list containing the fitted IRT model parameters and additional information about the fit.

Examples

```
#Generate some data (here 2pl model
require(data.table)
dat <- simIRT(Nsubs = 50,Nitems = 100,Nscales = 1,
  logitCMean = -10,logitCSD = 0,AMean = 1,ASD = .3,
  BMean=0,BSD = .5,
  AbilityMean = 0,AbilitySD = 1)

#fit using bigIRT
```

```
fit <- fitIRT(dat$dat,cores=2,score = 'score',id = 'id',
  scale = 'Scale',item = 'Item', pl=2)

  print(fit$personPars)
  print(fit$itemPars)
```

`normaliseIRT` *normaliseIRT*

Description

Normalise item response theory (IRT) parameters.

Usage

```
normaliseIRT(
  B,
  Ability,
  A,
  normbase = "Ability",
  normaliseScale = 1,
  normaliseMean = ifelse(normbase == "A", 1, 0),
  robust = TRUE
)
```

Arguments

`B`	Vector of item difficulty parameters.
`Ability`	Vector of persons' ability parameters.
`A`	Vector of item discrimination parameters.
`normbase`	The base from which the normalisation should be calculated. Can be 'Ability' or 'B'.
`normaliseScale`	The scale to normalise to.
`normaliseMean`	The mean to normalise to. The default is 0 when normbase is 'Ability' or 'B', and 1 when normbase is 'A'.
`robust`	if TRUE, outliers (greater than 1.5x the interquartile range from the interquartile region of 25-75%) are dropped before computing the mean and sd for normalisation.

Value

A list containing the normalised A, B, and Ability parameters.

Examples

```
B <- rnorm(100,2,1)
Ability <- rnorm(500,3,.5)
A <- rnorm(100,1.4,.05)
normaliseIRT(B, Ability, A, normbase='B')
```

---

simIRT *Simulate IRT data*

Description

Simulate IRT data

Usage

```
simIRT(
  Nsubs = 100,
  Nitems = 200,
  Nscales = 1,
  NitemsAnswered = Nitems,
  ASD = 0,
  AMean = 1,
  BSD = 1,
  BMean = 0,
  logitCSD = 1,
  logitCMean = -2,
  AbilitySD = 1,
  AbilityMean = 0,
  itemPreds = NA,
  AitemPredEffects = NA,
  BitemPredEffects = NA,
  logitCitemPredEffects = NA,
  personPreds = NA,
  AbilityPredEffects = NA,
  normalise = FALSE
)
```

Arguments

AbilityMean

wleIRT *Compute Weighted Likelihood Estimate (WLE) and Standard Error (SE)*

Description

Computes the WLE and SE for each subject and scale in a bigIRT model.

Usage

```
wleIRT(fit)
```

Arguments

fit	A bigIRT model fit object.

Value

A list containing two matrices. The first matrix contains the WLEs for each subject and scale. The second matrix contains the SEs for each subject and scale.

Examples

```
# Fit a bigIRT model
#fit <- bigIRT(data, itempars)

# Compute WLE and SE
#wleIRT(fit)
```

Index