Package 'MultiDiscreteRNG'

June 20, 2025

Type Package

Title Generation of Multivariate Correlated Discrete Data

Version 1.0.0

Author Chak Kwong (Tommy) Cheng [aut, cre],
Hakan Demirtas [aut]

Maintainer Chak Kwong (Tommy) Cheng <ccheng46@uic.edu>

Description Generation of multivariate correlated discrete data with generalized Poisson, negative binomial and binomial marginal distributions.

License GPL (>= 2)

Encoding UTF-8

LazyData true

Imports GenOrd, Matrix, MultiOrd, matrixcalc, mvtnorm

URL https://github.com/ckchengtommy/MultiDiscreteRNG

BugReports https://github.com/ckchengtommy/MultiDiscreteRNG/issues

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Contents

`BinToB` *This function implements step 5 of the algorithm in the paper. It converts the multivariate binary data back to the original binomial scale*

Description

This function implements step 5 of the algorithm in the paper. It converts the multivariate binary data back to the original binomial scale

Usage

```
BinToB(prop.vec.bin, BProp, Mlocation, bin.data)
```

Arguments

`prop.vec.bin`	vector of binary probabilities
`BProp`	Binary proportion
`Mlocation`	locations of median in the vector
`bin.data`	generated multivariate binary data

Value

multivariate Binomial data and its correlation matrix

`BinToGPD` *This function implements Step 5 of the algorithm. It converts the multivariate binary data back to the original GPD scale*

Description

This function implements Step 5 of the algorithm. It converts the multivariate binary data back to the original GPD scale

Usage

```
BinToGPD(prop.vec.bin, GPDprop, Mlocation, bin.data)
```

Arguments

`prop.vec.bin`	vector of binary probabilities
`GPDprop`	GPD proportion
`Mlocation`	locations of median in the vector
`bin.data`	generated multivariate binary data

Value

multivariate GPD data and its correlation matrix

`BinToNB` *This function implements Step 5 of the algorithm. It converts the multivariate binary data back to the original GPD scale*

Description

This function implements Step 5 of the algorithm. It converts the multivariate binary data back to the original GPD scale

Usage

```
BinToNB(prop.vec.bin, NBprop, Mlocation, bin.data)
```

Arguments

`prop.vec.bin`	vector of binary probabilities
`NBprop`	NB proportion
`Mlocation`	locations of median in the vector
`bin.data`	generated multivariate binary data

Value

multivariate NB data and its correlation matrix

`calc.bin.prob.B` *This function implements Step 1 of the algorithm. It collapses the discrete outcome to binary ones for each variable.*

Description

This function implements Step 1 of the algorithm. It collapses the discrete outcome to binary ones for each variable.

Usage

```
calc.bin.prob.B(n.vec, p.vec)
```

Arguments

`n.vec`	vector of number of trials
`p.vec`	vector of probabilities

Value

vector of binary probability, dichotomous threshold

`calc.bin.prob.GPD` *This function implements Step 1 of the algorithm. It collapses the discrete outcome to binary ones for each variable.*

Description

This function implements Step 1 of the algorithm. It collapses the discrete outcome to binary ones for each variable.

Usage

```
calc.bin.prob.GPD(theta.vec, lambda.vec)
```

Arguments

`theta.vec`	vector of theta values
`lambda.vec`	vector of lambda values

Value

vector of binary probability, dichotomous threshold

`calc.bin.prob.NB` *This function implements Step 1 of the algorithm. It collapses the discrete outcome to binary ones for each variable.*

Description

This function implements Step 1 of the algorithm. It collapses the discrete outcome to binary ones for each variable.

Usage

```
calc.bin.prob.NB(r.vec, prob.vec)
```

Arguments

`r.vec`	vector of number of trials
`prob.vec`	vector of probabilities

Value

vector of binary probability, dichotomous threshold

discrete_cont *Check positive definiteness of the intermediate matrix*

Description

Check positive definiteness of the intermediate matrix

Usage

```
discrete_cont(
  marginal,
  Sigma,
  support = list(),
  Spearman = FALSE,
  epsilon = 1e-06,
  maxit = 100
)
```

Arguments

marginal	a list of k elements, where k is the number of variables. The i-th element of marginal is the vector of the cumulative probabilities defining the marginal distribution of the i-th component of the multivariate variable. If the i-th component can take k_i values, the i-th element of marginal will contain $k_i - 1$ probabilities (the k_i-th is obviously 1 and shall not be included).
Sigma	the target correlation matrix of the discrete variables
support	a list of k elements, where k is the number of variables. The i-th element of support is the vector containing the ordered values of the support of the i-th variable. By default, the support of the i-th variable is $1, 2, ..., k_i$
Spearman	A logical flag indicating whether Spearman correlation should be used
epsilon	tolerance of the algorithm convergence
maxit	maximum iterations of the algorithm to correct PD matrix

Value

No return values; called it to check parameter inputs

genB *This function generates multivariate Binomial data*

Description

This function generates multivariate Binomial data

Usage

```
genB(no.rows, binObj)
```

Arguments

no.rows	number of data
binObj	intermediate correlation matrix object

Value

generated Binomial data with user's specifications

---

generate.binaryVar *Generate multivariate Binary data using the Emrich and Piedmonte (1991) approach Approach*

Description

Generate multivariate Binary data using the Emrich and Piedmonte (1991) approach Approach

Usage

```
generate.binaryVar(nObs, prop.vec.bin, corr.mat)
```

Arguments

nObs	number of observations
prop.vec.bin	probability of binary variables in a vector
corr.mat	Correlation matrix

Value

multivariate Binary Data

---

genGPD *This function generates multivariate GPD data*

Description

This function generates multivariate GPD data

Usage

```
genGPD(no.rows, binObj)
```

Arguments

no.rows	number of data
binObj	intermediate correlation matrix object

Value

generated GPD data with user's specifications

genNB *This function generates multivariate NB data*

Description

This function generates multivariate NB data

Usage

```
genNB(no.rows, binObj)
```

Arguments

no.rows	number of data
binObj	intermediate correlation matrix object

Value

generated NB data with user's specifications

GetGpoisPMF *Get probability mass function of Generalized Poisson distribution*

Description

This function returns a table of the probability mass function of GPD

Usage

```
GetGpoisPMF(p, theta, lambda, details = FALSE)
```

Arguments

p	probability of generalized Poisson distribution
theta	GPD theta value
lambda	GPD lambda value
details	A logical flag indicating computation information should be returned

Value

a PMF table

`QuantileGpois` *This function computes the quantile of Generalized Poisson*

Description

This function computes the quantile of Generalized Poisson

Usage

```
QuantileGpois(p, theta, lambda, details = FALSE)
```

Arguments

`p`	vector of probabilities
`theta`	vector of theta
`lambda`	vector of lambda
`details`	A logical flag to return the computational details

Value

the quantile of GPD

`simBinaryCorr.B` *This function implements Step 2 of the algorithm It calculates the intermediate binary correlations.*

Description

This function implements Step 2 of the algorithm It calculates the intermediate binary correlations.

Usage

```
simBinaryCorr.B(n.vec, p.vec, CorrMat, no.rows, steps = 0.025)
```

Arguments

`n.vec`	vector of number of trials
`p.vec`	vector of probabilities
`CorrMat`	specified Correlation matrix
`no.rows`	number of observations for generating Multivariate Binary data
`steps`	Fraction of difference between the current and target matrix to be added in each iteration.

Value

intermediate multivariate binary Correlation matrix

`simBinaryCorr.GPD` *This function implements Step 2 of the algorithm It calculates the intermediate binary correlations.*

Description

This function implements Step 2 of the algorithm It calculates the intermediate binary correlations.

Usage

```
simBinaryCorr.GPD(theta.vec, lambda.vec, CorrMat, no.rows, steps = 0.025)
```

Arguments

`theta.vec`	vector of theta values
`lambda.vec`	vector of lambda values
`CorrMat`	specified Correlation matrix
`no.rows`	number of observations for generating Multivariate Binary data
`steps`	Fraction of difference between the current and target matrix to be added in each iteration.

Value

intermediate multivariate binary Correlation matrix

`simBinaryCorr.NB` *This function implements Step 2 of the algorithm It calculates the intermediate binary correlations.*

Description

This function implements Step 2 of the algorithm It calculates the intermediate binary correlations.

Usage

```
simBinaryCorr.NB(r.vec, prob.vec, CorrMat, no.rows, steps = 0.025)
```

Arguments

`r.vec`	vector of number of trials
`prob.vec`	vector of probabilities
`CorrMat`	specified Correlation matrix
`no.rows`	number of observations for generating Multivariate Binary data
`steps`	fraction of difference between the current and target matrix to be added in each iteration.

Value

intermediate multivariate binary Correlation matrix

validation.Bparameters

Validate if the input Binomial parameters are within feasible range

Description

This function returns the sum of two numbers.

Usage

```
validation.Bparameters(n.vec, p.vec)
```

Arguments

n.vec	Vector of number of trials
p.vec	Vector of probability

Value

No return values; called it to check parameter inputs

Examples

```
validation.Bparameters(n.vec = c(10, 15), p.vec = c(0.4, 0.2))
```

validation.GPDparameters

Validate if the input GPD parameters are within feasible range

Description

This function returns the sum of two numbers.

Usage

```
validation.GPDparameters(theta.vec, lambda.vec)
```

Arguments

theta.vec	Vector of theta values
lambda.vec	Vector of lambda values

Value

No return values; called it to check parameter inputs

Examples

```
validation.GPDparameters(theta.vec = c(3, 2), lambda.vec = c(0.4, 0.2))
```

validation.NBparameters *Validate if the input NB parameters are within feasible range*

Description

Validate if the input NB parameters are within feasible range

Usage

```
validation.NBparameters(r.vec, prob.vec)
```

Arguments

r.vec	Vector of number of trials parameters
prob.vec	Vector of probabilities

Value

No return values; called it to check parameter inputs

Examples

```
validation.NBparameters(r.vec = c(10, 15), prob.vec = c(0.7, 0.5))
```

Index