**Package ‘rTableICC’**

January 18, 2018

**Type** Package

**Title** Random Generation of Contingency Tables

**Version** 1.0.6

**Date** 2018-01-18

**Author** Haydar Demirhan

**Maintainer** Haydar Demirhan <haydar.demirhan@rmit.edu.au>

**Description** Contains functions for random generation of R x C and 2 x 2 x K contingency tables. In addition to the generation of contingency tables over predetermined intraclass-correlated clusters, it is possible to generate contingency tables without intraclass correlations under product multinomial, multinomial, and Poisson sampling plans. It also consists of a function for generation of random data from a given discrete probability distribution function. See Demirhan (2016) <https://journal.r-project.org/archive/2016-1/demirhan.pdf> for more information.

**Depends** partitions, aster, stats

**License** GPL-3

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-01-18 10:32:11 UTC

**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>rTableICC-package</td>
<td>2</td>
</tr>
<tr>
<td>print.rTableICC</td>
<td>4</td>
</tr>
<tr>
<td>rDiscrete</td>
<td>5</td>
</tr>
<tr>
<td>rTable.2x2xK</td>
<td>6</td>
</tr>
<tr>
<td>rTable.RxC</td>
<td>8</td>
</tr>
<tr>
<td>rTableICC-internal</td>
<td>11</td>
</tr>
<tr>
<td>rTableICC.2x2xK</td>
<td>12</td>
</tr>
<tr>
<td>rTableICC.RxC</td>
<td>16</td>
</tr>
</tbody>
</table>

**Index** 21
rTableICC-package

Random Generation of R x C and 2 x 2 x K Contingency Tables

Description

Contains functions for random generation of R x C and 2 x 2 x K contingency tables. In addition to
the generation of contingency tables over predetermined intraclass-correlated clusters, it is possible
to generate contingency tables without intraclass correlations under product multinomial, multinom-
ial, and Poisson sampling plans. It also consists of a function for generation of random data from
a given discrete probability distribution function (Demirhan, 2016).

Details

Package: rTableICC
Type: Package
Version: 1.0.6
Date: 2018-01-18
License: GPL-3

To generate 2 x 2 x K and R x C contingency tables with intraclass-correlated observations under
product multinomial, multinomial or Poisson sampling plans, respectively use rTableICC.2x2xK
and rTableICC.RxC functions.

To generate 2 x 2 x K and R x C contingency tables without intraclass-correlated observations
product multinomial, multinomial or Poisson sampling plans, respectively use rTable.2x2xK and
rTable.RxC functions.

To generate random data from an empirical probability function, use rDiscrete function.

Author(s)

Haydar Demirhan
Maintainer: Haydar Demirhan <haydarde@hacettepe.edu.tr>

References

Altham, P.M. (1976) Discrete variable analysis for individuals grouped into families, Biometrika
63, 263–269.
Demirhan, H. (2016) rTableICC: An R package for random generation of 2x2xK and RxC contingen-
cy tables, The R Journal 8, 1, 48–63.
Demirhan, H. (2013) Bayesian estimation of log odds ratios over two-way contingency tables with


See Also

rTableICC.2x2xK, rTableICC.RxC, rTable.2x2xK, rTable.RxC, rDiscrete

Examples

```r
# --- For more examples, please refer to specific functions ---

# --- Generate a random value from given probability function ---
p = c(0.23,0.11,0.05,0.03,0.31,0.03,0.22,0.02)
rDiscrete(n=2, pf=p)

# --- Generate a 2x2x4 contingency table under multinomial sampling plan with ICCs ---
num.centers=4                              # Number of centers
max.cluster.size=9                        # Maximum allowed cluster size
num.cluster=95                            # Total number of clusters under each
                                        # center is equal across the centers
ICCs=array(0.1,dim=max.cluster.size)      # Assign equal ICCs for this example
ICCs[1]=0                                # Assign zero ICC to clusters with
                                        # one individual
sampl="Multinomial"                      # Generate table under multinomial
                                        # sampling plan
num.obs=900                               # Number of observations to be generated
cell.prob=array(0.0625,dim=c(num.centers,4))  # Cell probabilities sum up to one

x=rTableICC.2x2xK(p=cell.prob,theta=ICCs,M=num.cluster,sampling=sampl,
                  N=num.obs,print.regular=TRUE,print.raw=FALSE)
p=x

# --- Generate a 2x3 contingency table under product multinomial sampling plan ---
# --- with fixed row margins with ICCs ---
max.cluster.size=9                        # Maximum allowed cluster size
num.cluster=12                            # Total number of clusters
ICCs=array(0.1,dim=max.cluster.size)      # Assign equal ICCs for this example
ICCs[1]=0                                # Assign zero ICC to clusters with
                                        # one individual
sampl="Product"                          # Generate table under product
                                        # multinomial sampling plan
row=c(12,12)                              # Fixed row margins
cell.prob=array(0, dim=c(2,3))            # Cell probabilities sum up to one
cell.prob[1,1:2]=0.2
cell.prob[1,3]=0.1
cell.prob[2,1:2]=0.1
 cell.prob[2,3]=0.3

y=rTableICC.RxC(p=cell.prob,theta=ICCs,row.margins=row,M=num.cluster,
                 M=max.cluster,sampling=sampl,regular=TRUE,raw=FALSE)
```
print(rTableICC)

print(y)

# Generate a 2x2x8 contingency table under Poisson sampling plan without ICC
num.centers=8  # Number of centers
sampl="Poisson" # Generate table under Poisson
               # sampling plan
cell.mean=array(3,dim=c(2,2,num.centers)) # Enter mean number of individuals
                   # in each cell

z=rTable.2x2xK(sampling=sampl,lambda=cell.mean)
print(z)

# Generate a 5x7 contingency table under multinomial sampling plan without ICC
num.row=5      # Number of rows
num.col=7      # Number of columns
sampl="Multinomial" # Generate table under multinomial
                    # sampling plan
cell.prob=array(1/35,dim=c(num.row,num.col)) # Enter cell probabilities in RxC
                   # format
num.obs=124    # Number of observations

u=rTable.RxC(p=cell.prob,sampling=sampl,N=num.obs)
print(u)

print.rTableICC

Print Summary of Data Generation Process and Generated Data

Description

Prints summary information on data generation process and generated data.

Usage

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

Arguments

x an object including information and random table to be printed.
... other arguments.

Author(s)

Haydar Demirhan
Maintainer: Haydar Demirhan <haydarde@hacettepe.edu.tr>
Description

Generates random data from a given empirical probability function. It also returns cumulative distribution function corresponding to the entered probability function.

Usage

\texttt{rDiscrete(n = 1, pf)}

Arguments

- \texttt{n} \hspace{1cm} \text{number of observations.}
- \texttt{pf} \hspace{1cm} \text{empirical probability function.}

Details

\texttt{pf} is an array of any dimensionality with all elements sum up to one. If its dimension is greater than one, it is transformed to a row vector column-by-column.

Value

A list including

- \texttt{rdiscrete} \hspace{1cm} \text{an } n \times 1 \text{ vector that gives generated random data.}
- \texttt{cdf} \hspace{1cm} \text{a vector including cumulative distribution function.}

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan \texttt{<haydarde@hacettepe.edu.tr>}

References


Examples

\begin{verbatim}
p = c(0.23, 0.11, 0.05, 0.03, 0.31, 0.03, 0.22, 0.02) rDiscrete(n=2, pf=p)

# \texttt{pf} would be entered as a matrix:

p = matrix(c(0.23, 0.11, 0.05, 0.03, 0.31, 0.03, 0.22, 0.02), nrow=2, ncol=4, byrow = TRUE) rDiscrete(n=2, pf=p)
\end{verbatim}
```
p = matrix(c(0.23, 0.11, 0.05, 0.03, 0.31, 0.03, 0.22, 0.02), nrow=4, ncol=2, byrow = TRUE)
rDiscrete(n=2,pf=p)

# or pf would be entered as a three dimensional array:
p = array(c(0.23, 0.11, 0.05, 0.03, 0.31, 0.03, 0.22, 0.02), dim=c(2,2,2))
rDiscrete(n=2,pf=p)
```

---

**rTable.2x2xK**  
**Randomly Generate 2x2xK Contingency Tables**

**Description**

A generic function that generates 2 x 2 x K contingency tables under product multinomial, multinomial or Poisson sampling plans.

**Usage**

```
rTable.2x2xK(p,sampling="Multinomial",N,K=NULL,lambda=NULL,print.raw=FALSE)
```

**Arguments**

- `p`  
  A finite $2 \times 2 \times K$ matrix of cell probabilities.

- `sampling`  
  Sampling plan. It takes 'Product' for product multinomial sampling, 'Multinomial' for multinomial sampling, and 'Poisson' for Poisson sampling plans.

- `N`  
  Total number of individuals to be generated under product multinomial or multinomial sampling plans. It is a vector of positive integers containing total number of observations in each center under product multinomial sampling plan, a positive integer of total sample size under all centers under multinomial sampling plan, and not required under Poisson sampling plan. If `N` is not a positive integer, its converted to do so.

- `K`  
  Number of centers. It must be supplied if a scalar is entered as the value of lambda.

- `lambda`  
  Mean number of individuals in each cell of table. It is either a $2 \times 2 \times K$ positive matrix or a positive scalar under Poisson sampling plan and not required for both multinomial and product multinomial sampling plans. If a positive scalar is entered, mean number of individuals in each cell will be equal to each other.

- `print.raw`  
  If TRUE, generated raw data is printed on the screen.

**Details**

To generate random tables under multinomial sampling plan, multinomial distribution with entered cell probabilities and total number of observations is directly used.

To generate random tables under product multinomial sampling plan, center totals must be entered by `N`. It is not possible to fix any dimension of 2x2 table under each center. Suppose that center
totals are denoted by \( n_{ij+} \), where \( i, j = 1, 2 \). Then with the counts satisfying \( \sum_{ij} n_{ijk} = n_{ij+} \), we have the following multinomial form that `rTable.2x2xK` uses (Agresti, 2002):

\[
\frac{n_{ij+}!}{\prod_{ij} n_{ijk}!} \prod_{ij} p_{ij}^{n_{ijk}},
\]

where \( k = 1, \ldots, K \), \( n_{ijk} \) is the count of cell \((i, j, k)\), and given that an individual is in \( k \)th center, \( p_{ij} \) is the conditional probability of being the cell \((i, j)\) of 2x2 table. This multinomial form is used to generate data under each center.

To generate random tables under Poisson sampling plan, Poisson distribution with entered mean cell counts is directly used.

**Value**

A list with the following elements:

- `rTable`  
  A \( 2 \times 2 \times K \) dimensional array including generated 2 x 2 x K contingency table.

- `rTable.raw`  
  Generated table in a \( N \times 3 \) dimensional matrix in raw data format. First columns represent 2x2 table and the third is for center.

- `N`  
  Total number of generated individuals.

- `sampling`  
  Used sampling plan in data generation.

- `K`  
  Number of centers.

- `ICC`  
  Returns `FALSE` to indicate the data is generated without intraclass-correlated clusters.

- `structure`  
  Returns "2 x 2 x K" to indicate structure of generated table is "2 x 2 x K."

- `print.raw`  
  TRUE if generated table will be printed in raw data format.

- `print.regular`  
  TRUE if generated table will be printed in the format determined by `structure`.

**Author(s)**

Haydar Demirhan  
Maintainer: Haydar Demirhan <haydarde@hacettepe.edu.tr>

**References**


Examples

```r
# --- Generate a 2x2x8 contingency table under multinomial sampling plan ---
num.centers=8  # Number of centers
sampl="Multinomial"  # Generate table under multinomial sampling plan
cell.prob=array(0.03125,dim=c(2,2,num.centers))  # Enter cell probabilities in 2x2xK format
num.obs=124  # Number of observations
x=rTable.2x2xK(p=cell.prob,sampling=sampl,N=num.obs)
print(x)

# --- Generate a 2x2x8 contingency table under product multinomial sampling plan ---
sampl="Product"  # Generate table under product multinomial sampling plan
center.margins=array(10,num.centers)  # Enter center margins
y=rTable.2x2xK(p=cell.prob,sampling=sampl,N=center.margins)
print(y)

# --- Generate a 2x2x8 contingency table under Poisson sampling plan ---
num.centers=3  # Number of centers
sampl="Poisson"  # Generate table under Poisson sampling plan
cell.mean=array(3, dim=c(2,2,num.centers))  # Enter mean number of individuals in each cell
z=rTable.2x2xK(sampling=sampl,lambda=cell.mean)
print(z)
```

---

**rTable.2xC**

**Randomly Generate R x C Contingency Tables**

**Description**

A generic function that generates an RxC contingency table under product multinomial, multinomial, or Poisson sampling plans.

**Usage**

```r
rTable.2xC(p,row.margins=NULL,col.margins=NULL,sampling="Multinomial",N,
lambda=NULL,print.raw=FALSE)
```

**Arguments**

- **p**
  A finite $R \times C$ matrix of cell probabilities. It is not required under Poisson sampling plan.

- **row.margins**
  Includes fixed row margins under product multinomial sampling plan and not required for both multinomial and Poisson sampling plans.
col.margins
Includes fixed column margins under product multinomial sampling plan and not required for both multinomial and Poisson sampling plans.

sampling
Sampling plan. It takes 'Product' for product multinomial sampling, 'Multinomial' for multinomial sampling, and 'Poisson' for Poisson sampling plans.

N
Total number of individuals to be generated under product multinomial or multinomial sampling plans. It is a positive integer of total sample size under all centers for multinomial sampling plan and not required for both product multinomial and Poisson sampling plans. If N is not a positive integer, its converted to do so.

lambda
Mean number of individuals in each cell of table. It is either a $R \times C$ positive matrix or a positive scalar under Poisson sampling plan and not required for both multinomial and product multinomial sampling plans. If a positive scalar is entered, mean number of individuals in each cell will be equal to each other.

print.raw
If TRUE, generated raw data is printed on the screen.

Details
To generate random tables under multinomial sampling plan, multinomial distribution with entered cell probabilities and total number of observations is directly used.

To generate random tables under product multinomial sampling plan, at least one of row.margins or col.margins must be entered. Because both cell probabilities and fixed row or column margins are entered at the same time, margin probabilities calculated over fixed margins and entered $R \times C$ matrix of cell probabilities must be equal to each other. Suppose that row totals are fixed and $n_{i+}$ denote fixed row margins. Then with the counts satisfying $\sum_j n_{ij} = n_{i+}$, we have the following multinomial form that rTable.RxC uses (Agresti, 2002):

$$\frac{n_{i+}!}{\prod_j n_{ij}!} \prod_j p_{j|i}^{n_{ij}},$$

where $j = 1, \ldots, C$, $n_{ij}$ is the count of cell $(i,j)$, and given that an individual is in $i$th row, $p_{j|i}$ is the conditional probability of being in $j$th column of table. This multinomial form is used to generate data under each row margin. When column totals are fixed the same manner as the case of fixed row totals is followed.

To generate random tables under Poisson sampling plan, Poisson distribution with entered mean cell counts is directly used.

Value
A list with the following elements:

- rTable
  An $R \times C$ dimensional matrix including generated R x C contingency table.

- rTable.raw
  Generated table in a $N \times 2$ dimensional matrix in raw data format. Columns represent row and column numbers where each individual falls.

- N
  Total number of generated individuals.

- sampling
  Used sampling plan in data generation.

- R
  Number of rows.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Number of columns.</td>
</tr>
<tr>
<td>ICC</td>
<td>Returns FALSE to indicate the data is generated without intra-class-correlated clusters.</td>
</tr>
<tr>
<td>structure</td>
<td>Returns &quot;R x C&quot; to indicate structure of generated table is &quot;R x C.&quot;</td>
</tr>
<tr>
<td>print.raw</td>
<td>TRUE if generated table will be printed in raw data format.</td>
</tr>
<tr>
<td>print.regular</td>
<td>TRUE if generated table will be printed in the format determined by structure.</td>
</tr>
</tbody>
</table>

**Author(s)**

Haydar Demirhan

Maintainer: Haydar Demirhan <haydarde@hacettepe.edu.tr>

**References**


**Examples**

```r
# --- Generate a 5x7 contingency table under multinomial sampling plan ---
num.row=5  # Number of rows
num.col=7  # Number of columns
sample="Multinomial"  # Generate table under multinomial sampling plan
cell.prob=array(1/35,dim=c(num.row,num.col))  # Enter cell probabilities in RxC format
num.obs=124  # Number of observations

x=rTable.RxC(p=cell.prob,sampling=sample,N=num.obs)
print(x)

# --- Generate a 3x3 contingency table under product multinomial sampling plan ---
# --- with fixed row margins ---
num.row=3  # Number of rows
num.col=3  # Number of columns
row=c(32,12,11)  # Fixed row counts
sample="Product"  # Generate table under product multinomial sampling plan
cell.prob=array(0,dim=c(num.row,num.col))  # Enter cell probabilities in RxC format
cell.prob[1,1]=0.12
cell.prob[1,2]=0.24
cell.prob[1,3]=32/55-0.36
cell.prob[2,1]=0.07
cell.prob[2,2]=0.1
cell.prob[2,3]=12/55-0.17
cell.prob[3,1]=0.05
cell.prob[3,2]=0.10
```

cell.prob[3,3]=11/55*0.15  # Marginal and cell probabilities
# should be equal to each other

y1=rTable.RxC(p=cell.prob,sampling=sampl,row.margins=row)
print(y1)

# --- Generate a 3x3 contingency table under product multinomial sampling plan ---
# with fixed row margins
---
num.row=3  # Number of rows
num.col=3  # Number of columns
col=c(5,5,10)  # Fixed row counts
sampl="Product"  # Generate table under product
# multinomial sampling plan

cell.prob=array(0,dim=c(num.row,num.col))  # Enter cell probabilities in RxC format

cell.prob[1,1]=0.1

cell.prob[1,2]=0.1

cell.prob[1,3]=0.05

cell.prob[2,1]=0.05

cell.prob[2,2]=0.1

cell.prob[2,3]=0.1

cell.prob[3,1]=0.05

cell.prob[3,2]=0.1

cell.prob[3,3]=0.1  # Marginal and cell probabilities
# should be equal to each other

y2=rTable.RxC(p=cell.prob,sampling=sampl,col.margins=col)
print(y2)

# --- Generate a 6x4 contingency table under Poisson sampling plan ---
# Number of rows
---
num.row=6
num.col=4  # Number of columns
sampl="Poisson"  # Generate table under Poisson
# sampling plan

cell.mean=array(3,dim=c(6,4))  # Enter mean number of individuals
# in each cell

z=rTable.RxC(lambda=cell.mean,sampling=sampl)
print(z)

---

rTableICC-internal  Functions for internal use only
---

Description

Contains functions designed for internal use only. Functions including "default" in name makes basic checks using check function, and then call main function. Those including "main" in name carries on main processes to generate random tables. rtableICC.RxC.engine is the function multiply called by rTable.RxC.main.
Description

A generic function that generates $2 \times 2 \times K$ contingency tables over intraclass-correlated cells under product multinomial, multinomial or Poisson sampling plans.

Usage

```r
rTableICC.2x2xK(p, theta, M, sampling="Multinomial", N=0, lambda=NULL, zero.clusters=FALSE, print.regular=TRUE, print.raw=FALSE)
```

Arguments

- `p`: A finite matrix of cell probabilities. Dimension of `p` is $K \times 4$, where $2 \times 2$ tables under each of $K$ centers is expanded to a vector of 4 elements. Elements in each row of `p` should correspond to the cells $(1,1)$, $(1,2)$, $(2,1)$, and $(2,2)$, respectively.
- `theta`: A finite and positive valued $(T - 1) \times 1$ vector of predetermined ICCs, where $T$ is the maximum number of individuals allowed in each cluster. The first element of `theta` represents the ICC for clusters of size 2.
- `M`: The total number of clusters under each center. If number of clusters under each center is unbalanced, $M$ would be a $K \times 1$ vector.
- `sampling`: Sampling plan. It takes 'Product' for product multinomial sampling, 'Multinomial' for multinomial sampling, and 'Poisson' for Poisson sampling plans.
- `N`: Total number of individuals to be generated under product multinomial or multinomial sampling plans. It is a vector of positive integers containing total number of observations in each center under product multinomial sampling plan, a positive integer of total sample size under all centers under multinomial sampling plan, and not required under Poisson sampling plan. If `N` is not a positive integer, it is converted to do so.
- `lambda`: Mean number of individuals in each cluster. It is either a $K \times 1$ positive vector or a positive scalar under Poisson sampling plan and not required for both multinomial and product multinomial sampling plans. If a positive scalar is entered, mean number of individuals in each cluster under all centers will be equal to each other. If a vector is entered, mean number of individuals in each cluster under each center will be equal to each other.
zero.clusters If TRUE, generation of clusters with no individuals are allowed. Otherwise, each cluster has at least one individual.

print.regular If TRUE, generated random table is printed in 2x2xK format. Otherwise, generated random table is printed in two dimensional format.

print.raw If TRUE, generated raw data is printed on the screen.

Details
To generate random tables under multinomial sampling plan, first total sample size is distributed to clusters with equal probabilities using the code `rmultinom(1, N, rep(1/M, M))`. Then, for each center, the package partitions is utilized to distribute individuals across cells of 2 x 2 tables under the pre-determined intraclass correlations. Let $n$ and $m$ be integer to be partitioned (cluster size) and order of partition, respectively. If there is more than one individual ($n > 1$) in a cluster, all possible compositions of order $RC$ of cluster size $n$ into at most $m$ parts are generated by using compositions function. This provides all possible distributions of individuals in the cluster of interest into 2 x 2 table of interest. If all individuals are at the same cell, the following equation is used to calculate the probability that all individuals in the $i$th cluster fall in the same cell of a contingency table of interest:

$$\theta_i p_{ij} + (1 - \theta_i)(p_{ij})^t,$$

where $i, j = 1, 2$, $0 \leq \theta \leq 1$, $\theta_t$ is the intraclass correlation for clusters of size $t$ for $t = 2, \ldots, T$, and $\theta_1 = 0$. Otherwise, the probability that the individuals are in different but specified cells is calculated as follows:

$$(1 - \theta_i) \prod_{ij} (p_{ij})^{n_{rij}},$$

where $n_{rij}$ be the number of individuals from $r$th cluster falling in the $i$th row and $j$th column of the considered 2 x 2 table (Altham, 1976; Nandram and Choi, 2006; Demirhan, 2013). This provides probability of each possible distribution. Then, calculated probabilities are normalized and the function rDiscrete is utilized to randomly select one of the generated compositions. By this way, a realization is obtained for each cluster having more than one individual. If there is only one individual in a cluster, a realization is obtained by assigning the individual to cells according to the entered cell probabilities using the function rDiscrete. The resulting random 2x2 table under one of $K$ centers is constructed by combining these realizations. This process is repeated for all of $K$ centers.

To generate random tables under product multinomial sampling plan, center margins must be entered by N. Because both cell probabilities and fixed center margins are entered at the same time, margin probabilities calculated over the number of individuals in each center and entered $K \times 4$ matrix of cell probabilities must be equal to each other. To ensure intraclass correlations, the same manner as multinomial sampling plan is applied to each center. Suppose that center totals are denoted by $n_{ij+}$, where $i, j = 1, 2$. Then with the counts satisfying $\sum_{i+j} n_{ijk} = n_{ij+}$, we have the following multinomial form that rTableICC.2x2xK uses (Agresti, 2002):

$$\frac{n_{ij+1}}{\prod_{ij} n_{ijk}!} \prod_{ij} p_{ijk}^{n_{ijk}},$$
where \( k = 1, \ldots, K \). \( n_{ijk} \) is the count of cell \((i,j,k)\), and given that an individual is in \( k \)th center, \( p_{ijk} \) is the conditional probability of being in the cell \((i,j)\) of 2 x 2 table. This multinomial form is used to generate data under each center.

To generate random tables under Poisson sampling plan, the same manner as multinomial sampling plan is taken except cluster sizes are generated from Poisson distribution with entered mean cluster counts and total sample size is calculated over the generated cluster sizes. If zero sized clusters are not allowed, truncated Poisson distribution is used to generate cluster counts.

Because total sample size is randomly distributed into the clusters, it is coincidentally possible to have clusters with more individuals than the allowed maximum cluster size. In this case, the following error message is generated:

Maximum number of individuals in one of the clusters is 14, which is greater than maximum allowed cluster size. In this case, the execution is stopped.

Value

Let \( C \) be the set of clusters in which all individuals fall in a single cell of the contingency table and \( C' \) be the complement of \( C \), \( K \) be the number of centers, and \( T \) be the maximum cluster size.

A list with the following elements is generated:

- \( g.t \): A \( 2K \times 2 \times (T - 1) \) dimensional array including the number of clusters of size \( t \) in \( C' \) of size \( t \) with all individuals in cell \((i,j)\), where \( i,j = 1,2 \) and \( t = 2,\ldots,T \).
- \( g.tilde \): A \( (T - 1) \times 1 \) dimensional vector including the number of clusters of size \( t \) in \( C' \), where \( t = 2,\ldots,T \).
- \( rTable \): A \( K \times (2 \times 2) \) dimensional matrix including generated 2 x 2 contingency table under each center in its each row.
- \( rTable.raw \): Generated table in a \( N \times 2K \times 2 \) dimensional array in raw data format.
- \( rTable.regular \): Generated table in a \( 2 \times 2 \times K \) dimensional array.
- \( N \): Total number of generated individuals.
- \( cluster.size \): Size of each generated cluster.
- \( sampling \): Used sampling plan in data generation.
- \( M \): Total number of clusters under each center.
- \( K \): Number of centers.
- \( T \): Maximum allowed cluster size.
- \( ICC \): Returns TRUE to indicate the data is generated under intraclass-correlated clusters.
- \( structure \): Returns "2 x 2 x K" to indicate structure of generated table is "2 x 2 x K."
- \( print.raw \): TRUE if generated table will be printed in raw data format.
- \( print.regular \): TRUE if generated table will be printed in the format determined by \( structure \).

Author(s)

Haydar Demirhan
Maintainer: Haydar Demirhan <haydarde@hacettepe.edu.tr>
References


Examples

```r
# --- Generate a 2x2x4 contingency table under multinomial sampling plan ---
num.centers=4          # Number of centers
max.cluster.size=9     # Maximum allowed cluster size
num.cluster=95         # Total number of clusters under
                        # centers each center is equal across the
ICCs=array(0.1,dim=max.cluster.size) # Assign equal ICCs for this example
ICCs[1]=0              # Assign zero ICC to clusters with
                        # one individual
sampl="Multinomial"   # Generate table under multinomial
                        # sampling plan
num.obs=900            # Number of observations to be
                        # generated
cell.prob=array(0.0625,dim=c(num.centers,4)) # Cell probabilities sum up to one
zeros=FALSE            # Do not allow zero sized clusters
x=rTableICC.2x2xK(p=cell.prob,theta=ICCs,M=num.cluster,sampling=sampl,zero.clusters=zeros,
                  N=num.obs,print.regular=TRUE,print.raw=FALSE)
print(x)

# --- Generate a 2x2x4 contingency table under product multinomial sampling plan ---
sampl="Product"        # Generate table under product
                        # multinomial sampling plan
num.obs=c(200,200,200,200) # Number of observations to be generated
                        # under each center
cell.prob=array(0.0625,dim=c(num.centers,4)) # Cell probabilities sum up to one
zeros=FALSE            # Do not allow zero sized clusters
y=rTableICC.2x2xK(p=cell.prob,theta=ICCs,M=num.cluster,sampling=sampl,
                  zero.clusters=zeros,N=num.obs,print.regular=TRUE,print.raw=FALSE)
print(y)

# --- Generate a 2x2x4 contingency table under Poisson sampling plan ---
sampl="Poisson"        # Generate table under Poisson
```
Description

A generic function that generates R x C contingency tables over intraclass-correlated cells under product multinomial, multinomial or Poisson sampling plans.

Usage

rTableICC.RxC(p=NULL, theta, M, row.margins=NULL, col.margins=NULL, sampling="Multinomial", N=1, lambda=NULL, zero.clusters=FALSE, print.regular=TRUE, print.raw=FALSE)

Arguments

p A finite \( R \times C \) matrix of cell probabilities.

theta A finite and positive valued \((T - 1) \times 1\) vector of predetermined ICCs, where \(T\) is the maximum number of individuals allowed in each cluster. The first element of \(\theta\) represents the ICC for clusters of size 2.

M The total number of clusters under each factor. It must be a positive scalar under multinomial and Poisson sampling plans and can be a vector of length equal to that of number of rows or columns under product multinomial sampling plan. If it is a scalar, number of clusters under levels of fixed margin is assigned equal.

row.margins Includes fixed row margins under product multinomial sampling plan and not required for both multinomial and Poisson sampling plans.
col.margins: Includes fixed column margins under product multinomial sampling plan and not required for both multinomial and Poisson sampling plans.

sampling: Sampling plan. It takes 'Product' for product multinomial sampling, 'Multinomial' for multinomial sampling, and 'Poisson' for Poisson sampling plans.

N: Total number of individuals to be generated under product multinomial or multinomial sampling plans. It is a positive integer of total sample size under multinomial sampling plan and not required under both product multinomial and Poisson sampling plans. If N is not a positive integer, it is converted to do so.

lambda: Mean number of individuals in each cell of table. It is an $R \times C$ positive matrix under Poisson sampling plan and not required for both multinomial and product multinomial sampling plans.

zero.clusters: If TRUE, generation of clusters with no individuals are allowed. Otherwise, each cluster has at least one individual.

print.regular: If TRUE, generated random table is printed in $2 \times 2 \times K$ format. Otherwise, generated random table is printed in two dimensional format.

print.raw: If TRUE, generated raw data is printed on the screen.

Details

To generate random tables under multinomial sampling plan, first total sample size is distributed to clusters with equal probabilities using the code `rmultinom(1, N, rep(1/M, M))`. Then the package `partitions` is utilized to distribute individuals across cells under the pre-determined intraclass correlations. Let $n$ and $m$ be integer to be partitioned (cluster size) and order of partition, respectively. If there is more than one individual ($n > 1$) in a cluster, all possible compositions of order $RC$ of cluster size $n$ into at most $m$ parts are generated by using `compositions` function. This provides all possible distributions of individuals in the cluster of interest into $RC$ cells. If all individuals are at the same cell, the following equation is used to calculate the probability that all individuals in the $i$th cluster fall in the same cell of a contingency table of interest:

$$
\theta_t p_{ij} + (1 - \theta_t)(p_{ij})^t,
$$

where $i = 1, \ldots, R$, $j = 1, \ldots, C$, $0 \leq \theta \leq 1$, $\theta_t$ is the intraclass correlation for clusters of size $t$ for $t = 2, \ldots, T$, and $\theta_1 = 0$. Otherwise, the probability that the individuals are in different but specified cells is calculated as follows:

$$
(1 - \theta_t) \prod_{ij} (p_{ij})^{n_{kij}},
$$

where $n_{kij}$ be the number of individuals from $k$th cluster falling in the $i$th row and $j$th column of the considered RxC table (Altham, 1976; Nandram and Choi, 2006; Demirhan, 2013). This provides probability of each possible distribution. Then, calculated probabilities are normalized and the function `rdiscrete` is utilized to randomly select one of the generated compositions. By this way, a realization is obtained for each cluster having more than one individual. If there is only one individual in a cluster, a realization is obtained by assigning the individual to cells according to the entered cell probabilities using the function `rdiscrete`. The resulting random RxC table is constructed by combining these realizations.
To generate random tables under product multinomial sampling plan, at least one of row margins or col. margins must be entered. Because both cell probabilities and fixed row or column margins are entered at the same time, margin probabilities calculated over fixed margins and entered $R \times C$ matrix of cell probabilities must be equal to each other. To ensure intraclass correlations, the same manner as multinomial sampling plan is applied to the fixed margin. Suppose that row totals are fixed and $n_{i+}$ denotes fixed row margins. Then with the counts satisfying $\sum_j n_{ij} = n_{i+}$, we have the following multinomial form that rTableICC.RxC uses (Agresti, 2002):

$$\frac{n_{i+1}}{\prod_j n_{ij}} \prod_j p_{rij},$$

where $j = 1, \ldots, C$, $n_{ij}$ is the count of cell $(i, j)$, and given that an individual is in $i$th row, $p_{rij}$ is the conditional probability of being in the $j$th column of table. This multinomial form is used to generate data under each row margin. When column totals are fixed the same manner as the case of fixed row totals is followed.

To generate random tables under Poisson sampling plan, the same manner as multinomial sampling plan is taken except cell counts are generated from Poisson distribution with entered mean cell counts and total sample size is calculated over the generated cell counts. If zero sized clusters are not allowed, truncated Poisson distribution is used to generate cluster counts.

Because total sample size is randomly distributed into the clusters, it is coincidentally possible to have clusters with more individuals than the allowed maximum cluster size. In this case, the following error message is generated:

Maximum number of individuals in one of the clusters is 14, which is greater than maximum allowed cluster size. Increase total number of clusters or decrease total number of individuals and execution is stopped.

Value

Let $C$ be the set of clusters in which all individuals fall in a single cell of the contingency table and $C'$ be the complement of $C$, $K$ be the number of centers, and $T$ be the maximum cluster size.

A list with the following elements is generated:

- g.t: A $R \times C \times (T-1)$ dimensional array including the number of clusters of size $t$ in $C'$ of size $t$ with all individuals in cell $(i, j)$, where $i = 1, \ldots, R$, $j = 1, \ldots, C$, and $t = 2, \ldots, T$.
- g.tilde: A $(T-1) \times 1$ dimensional vector including the number of clusters of size $t$ in $C'$, where $t = 2, \ldots, T$.
- rTable: A $RC \times 1$ dimensional vector including generated RxC contingency table in a row.
- rTable.raw: Generated table in a $R \times C \times N$ dimensional array in raw data format.
- rTable.regular: Generated table in an $R \times C$ dimensional matrix.
- N: Total number of generated individuals.
- cluster.size: Size of each generated cluster.
- sampling: Used sampling plan in data generation.
- M: Total number of clusters.
rTableICC.RxC

R  Number of rows.
C  Number of columns.
T  Maximum allowed cluster size.
ICC Returns TRUE to indicate the data is generated under intraclass-correlated clusters.
structure Returns "R x C" to indicate structure of generated table is "R x C."
print.raw TRUE if generated table will be printed in raw data format.
print.regular TRUE if generated table will be printed in the format determined by structure.

Author(s)
Haydar Demirhan
Maintainer: Haydar Demirhan <haydarde@hacettepe.edu.tr>

References

Examples

```r
# --- Generate a 2x3 contingency table under multinomial sampling plan ---
max.cluster.size=9  # Maximum allowed cluster size
num.cluster=12      # Total number of clusters
ICCs=array(0.1,dim=max.cluster.size)  # Assign equal ICCs for this example
ICCs[1]=0           # Assign zero ICC to clusters with
                    # one individual
sampl="Multinomial" # Generate table under multinomial
                    # sampling plan
num.obs=24          # Number of observations to be
                    # generated
cell.prob=array(1/6,dim=c(2,3))  # Cell probabilities sum up to one
zeros=FALSE         # Do not allow zero sized clusters

x=rTableICC.RxC(p=cell.prob,theta=ICCs,M=num.cluster,sampling=sampl,
    N=num.obs,zero.clusters=zeros,print.regular=TRUE,
    print.raw=FALSE)
print(x)
```

# --- Generate a 2x3 contingency table under product multinomial sampling plan ---
```
# --- with fixed row margins
sampl="Product"  # Generate table under product
row=c(12,12)  # multinomial sampling plan
    # Fixed row margins
cell.prob=array(0,dim=c(2,3))  # Cell probabilities sum up to one
cell.prob[1,1:2]=0.2
cell.prob[1,3]=0.1
cell.prob[2,1:2]=0.1
cell.prob[2,3]=0.3  # Marginal and cell probabilities
    # should be equal to each other

y1=rTableICC.RxC(p=cell.prob,theta=ICCs,row.margins=row,M=num.cluster,
    sampling=sampl,print.regular=TRUE,print.raw=FALSE)
print(y1)

# --- Generate a 3x2 contingency table under product multinomial sampling plan ---
# --- with fixed column margins
col=c(12,12)
cell.prob=array(0,dim=c(3,2))  # Cell probabilities sum up to one
cell.prob[1:2,1]=0.2
cell.prob[1,2]=0.1
cell.prob[2,2]=0.1
cell.prob[3,1]=0.1
cell.prob[3,2]=0.3

y2=rTableICC.RxC(p=cell.prob,theta=ICCs,col.margins=col,M=num.cluster,
    sampling=sampl,print.regular=TRUE,print.raw=FALSE)
print(y2)

# --- Generate a 4x3 contingency table under Poisson sampling plan ---
sampl="Poisson"  # Generate table under product
    # multinomial sampling plan
cell.prob=array(1/12,dim=c(4,3))  # Cell probabilities sum up to one
cell.mean=array(4,dim=c(4,3))  # Define mean number of individuals
    # in each cell
max.cluster.size=19  # Maximum allowed cluster size
ICCs=array(0.1,dim=max.cluster.size)  # Assign equal ICCs for this example
ICCs[1]=0

z=rTableICC.RxC(p=cell.prob,lambda=cell.mean,theta=ICCs,row.margins=row,
    M=num.cluster,sampling=sampl,print.regular=TRUE,print.raw=FALSE)
print(z)
```
Index

*Topic Poisson
  rTable.2x2xK, 6
  rTable.RxC, 8
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
*Topic contingency
  rTable.2x2xK, 6
  rTable.RxC, 8
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
*Topic correlation
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
*Topic datagen, contingency, correlation, intraclass, Poisson, multinomial, product
  rTableICC-package, 2
*Topic datagen
  rDiscrete, 5
  rTable.2x2xK, 6
  rTable.RxC, 8
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
*Topic discrete
  rDiscrete, 5
*Topic distribution
  rDiscrete, 5
*Topic intraclass
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
* Topic multinomial
  rTable.2x2xK, 6
  rTable.RxC, 8
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
*Topic product
  rTable.2x2xK, 6
  rTable.RxC, 8
  rTableICC.2x2xK, 12
  rTableICC.RxC, 16
check (rTableICC-internal), 11
print.rTableICC, 4
rDiscrete, 3, 5, 12
rTable.2x2xK, 3, 6, 12
rTable.2x2xK.default
  (rTableICC-internal), 11
rtable.2x2xK.main (rTableICC-internal), 11
rTable.RxC, 3, 8, 12
rTable.RxC.default
  (rTableICC-internal), 11
rtable.RxC.main (rTableICC-internal), 11
rTableICC-internal, 11
rTableICC-package, 2
rTableICC.2x2xK, 3, 5, 12
rTableICC.2x2xK.default
  (rTableICC-internal), 11
rtableICC.2x2xK.main
  (rTableICC-internal), 11
rTableICC.RxC, 3, 5, 12
rTableICC.RxC.default
  (rTableICC-internal), 11
rtableICC.RxC.engine
  (rTableICC-internal), 11
rTableICC.RxC.main
  (rTableICC-internal), 11