

# Package ‘sdcTable’

April 19, 2009

**Title** statistical disclosure control for tabular data

**Version** 0.0.2

**Date** 2009-01-26

**Author** Bernhard Meindl

**Maintainer** Bernhard Meindl <Bernhard.Meindl@statistik.gv.at>

**Description** R-Package for statistical disclosure control for tabular data.

**Depends** lpSolve

**License** GPL-2

**Repository** CRAN

**Date/Publication** 2009-01-26 08:53:51

## R topics documented:

createFullData . . . . .	2
exampleFullData . . . . .	3
exampleMinimalData . . . . .	4
processTableHITAS . . . . .	5
processTableHYPERCUBE . . . . .	6
protectTable . . . . .	7
roundSubtable . . . . .	8
summary.safeTable . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

createFullData      *createFullData*

---

### Description

Function to generate an object of class fullData which can then be used as input object for several other functions in sdcTable such as protectTable.

### Usage

```
createFullData (minimalData, indexvars, l, suppVals=FALSE, suppLimit=NULL, suppZeros=FALSE)
```

### Arguments

minimalData	a data.frame containing a column for each hierarchical variable in a specific coding and a column for the corresponding numerical values. The values need to be integer at the moment.
indexvars	a vector containing the position of the hierarchical variables within minimalData
l	a list which element i defining the hierarchical structure of the i-th hierarchical variable.
suppVals	should during the generation of the complete data set values be suppressed?
suppLimit	if suppVals is TRUE, what is the threshold for primary suppressing values? ()
suppZeros	binary: should values of empty cells be suppressed as well?

### Details

Have a look at the link given below.

### Value

object of class fullData

### Note

fix me: LOTS! more primary suppression rules should be implemented; what to do with real numbers? simplify the generation of object of class fullData?

### Author(s)

Bernhard Meindl

**Examples**

```
## Not run:
# create minimalData (simple 2-dimensional Example)
# first hierarchical variable: sex: we have {total}-> {male, women}
sex <- c("00", "01", "02")

# second hierarchical variable: a total of 13 age-groups including the total
age <- c("000", "001", "002", "003", "004", "005", "006", "007", "008", "009", "010", "011", "012")

# for the minimal dataset we don't need the totals: we generate each combination
minimalData <- expand.grid(sex[-1], age[-1])

# the column with values is added (just random numbers in this case)
minimalData$value <- rpois(nrow(minimalData), 7)

# position of the hierarchical variables in minimalData
indexvars <- c(1,2)

# level-structure of the hierarchical variables:
# the first level (sex) consists of a total of 2 levels (this is total and m
# the first level (age) consists of a total of 12 levels (this is the total a
l <- list()
l[[1]] <- c(1, 1)
l[[2]] <- c(1, 2)

# we generate an object of class fullData which we can then use as input obj
fullDat <- createFullData (minimalData, indexvars, l, suppVals=TRUE, suppLim
class(fullDat)

#result <- protectTable(fullDat, method="GHMITER")
summary(result)

# result$fullData$data is the protected dataset containing all totals. The u

## End(Not run)
```

---

exampleFullData      *Example complete data set used as input for various methods*

---

**Description**

exampleFullData is a data set of class 'fullData'

**Usage**

```
data(exampleFullData)
```

**Format**

A list containing several necessary information.

**data** data set containing a column for each dimension, a column representing the numerical value of each cell and a column which indicates if a cell is marked as sensitive (P) or not.

**dims** a list representing the structure of each dimensional variable.

**indexvars** a vector representing the columns of the dimensional variables in data set.

**supps2check** a numeric vector of indices representing cells that need to be protected.

**numberindexvars** the total number of dimensional variables.

**Examples**

```
## Not run:
  data(exampleFullData)
  head(exampleFullData$data)
  dim(exampleFullData$data)

## End(Not run)
```

---

exampleMinimalData *minimal data set used as input to generate complete data set.*

---

**Description**

a data set containing a total of 3 dimensional variables and the corresponding cell values. Only those characteristics of the dimensional variables are available in exampleMinimalData that cannot be calculated as linear combinations from other characteristics (sums).

**Usage**

```
data(exampleMinimalData)
```

**Format**

A data.frame containing a column for each dimensional variable and a column for the corresponding cell value.

**Var1** characteristic of dimensional variable 1.

**Var2** characteristic of dimensional variable 2.

**Var3** characteristic of dimensional variable 3.

**val** cell value of cell given by characteristics of Var1, Var2 and Var3.

**Examples**

```

## Not run:
      data(exampleMinimalData)
      head(exampleMinimalData)
      dim(exampleMinimalData)

## End(Not run)

```

---

```
processTableHITAS  processTableHITAS
```

---

**Description**

HITAS - algorithm for secondary cell suppression using optimal suppression algorithm.

**Usage**

```
processTableHITAS (fullData, ub=NULL, lb=NULL, UPLPerc=35, LPLPerc=25, weight="value")
```

**Arguments**

fullData	object from class fullData.
ub	probably known upper bounds for cell values. If not specified, cellvalue * 100 is assumed.
lb	probably known lower bounds for cell values. If not specified, 0 is assumed for each cell (non-negative table).
UPLPerc	percentage of (upper) protection for each cell. Maximum possible value calculated for any sensitive cell must be greater or equal the cell value plus ULPerc %.
LPLPerc	percentage of (lower) protection for each cell. Minimum possible value calculated for any sensitive cell must be less or equal the cell value minus LPLPerc %.
weight	parameter to use for objective function. Possible choices are values and logs.

**Details**

Have a look at the links given below.

**Value**

Manipulated data.

**Note**

processTableHITAS() protects hierarchical tabular data using the HiTaS approach.

**Author(s)**

Bernhard Meindl

**References**

de Wolf, P.P (2002). HiTaS: A Heuristic Approach to Cell Suppression in Hierarchical Tables. In: Domingo-Ferrer, J. (Hrsg.): Inference Control in Statistical Databases. Vol. 2316. \ Fischetti, M., Salazar, J.J. (2000). Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints. In: Journal of the American Statistical Association 95, 916-928.

**Examples**

```
## Not run:
  data(exampleFullData)
  erg <- processTableHITAS(exampleFullData, UPLPerc=15, LPLPerc=15)
  summary(erg)

## End(Not run)
```

---

```
processTableHYPERCUBE
      processTableHYPERCUBE
```

---

**Description**

HYPERCUBE - algorithm for secondary cell suppression.

**Usage**

```
processTableHYPERCUBE (fullData, allowZeros=TRUE, randomResult=FALSE, suppMethod="n
```

**Arguments**

<code>fullData</code>	object from class <code>fullData</code> .
<code>allowZeros</code>	should empty cells be allowed to be part of the suppression scheme?
<code>randomResult</code>	if several possible suppression schemes exist, always use the same or pick a random one?
<code>suppMethod</code>	should the algorithm minimize the number of suppressions ( <code>minSupps</code> ), minimize the suppressed sum of values ( <code>minSum</code> ) or the sum of logarithmized values ( <code>minSumLogs</code> )?
<code>protectionLevel</code>	protection level in percent.

**Details**

Have a look at the link given below.

**Value**

Manipulated data.

**Note**

The algorithm runs only once through the entire data set. To make completely sure that also secondary suppressions are enough protected one should use function `protectTable(x, method="GHMITER")` which loops through the data set to ensure protection.

**Author(s)**

Bernhard Meindl

**References**

Repsilber, D. (1999). Das Quaderverfahren. In: Forum der Bundesstatistik, Band 31/1999

**Examples**

```
## Not run:
  data(exampleFullData)
  erg <- processTableHYPERCUBE(exampleFullData, suppMethod="minSUM")
  summary(erg)

## End(Not run)
```

---

protectTable      *protectTable*

---

**Description**

Wrapper-function to call several algorithms for secondary cell suppression of tabular data.

**Usage**

```
protectTable (fullData, method, ...)
```

**Arguments**

<code>fullData</code>	object from class <code>fullData</code> .
<code>method</code>	either <code>HYPERCUBE</code> , <code>HITAS</code>
<code>...</code>	additional input parameters.

**Details**

For correct input parameters for function `protectTable()` please have a look at Please have a look at the corresponding functions `processTableHYPERCUBE()` or rather `processTableHITAS()`.

**Value**

Manipulated data.

**Author(s)**

Bernhard Meindl

**References**

Repsilber, D. (1999). Das Quaderverfahren. In: Forum der Bundesstatistik, Band 31/1999. \ de  
 Wolf, P.P (2002). HiTaS: A Heuristic Approach to Cell Suppression in Hierarchical Tables. In:  
 Domingo-Ferrer, J. (Hrsg.): Inference Control in Statistical Databases. Vol. 2316. \ Fischetti, M.,  
 Salazar, J.J. (2000). Models and Algorithms for Optimizing Cell Suppression in Tabular Data with  
 Linear Constraints. In: Journal of the American Statistical Association 95, 916-928.

**Examples**

```
## Not run:
  data(exampleFullData)
  result1 <- protectTable(exampleFullData, method="HYPERCUBE", allowZeros=FALSE)
  result2 <- protectTable(exampleFullData, method="HITAS", LPLPerc=15, UPLPerc=15)
  summary(result1)
  summary(result2)

## End(Not run)
```

---

roundSubtable      *roundSubtable*

---

**Description**

Wrapper-function to call several algorithms for rounding algorithms.

**Usage**

```
roundSubtable(subtab, indexvars=NULL, base=5, maxS=NULL, method=NULL)
```

**Arguments**

subtab	subtable derived from object of class fullData.
indexvars	position of dimensional variables (needed only for controlled rounding procedure!).
base	the base-number to which cell values should be rounded.
maxS	maximum number of multiples of the base numbers at which a cell value is allowed to change (rounded up or down).
method	either simple; random or controlled are valid choices.

**Details**

Depending on the choice for parameter method, simple (or rather conventional), random or controlled rounding will be done.

**Value**

Manipulated data.

**Author(s)**

Bernhard Meindl

**References**

Salazar, J.J., Bycroft C., Staggemeier A. (2005). Controlled Rounding Implementation. Supporting paper for Joint UNECE/EUROSTAT work session on statistical data confidentiality.

**Examples**

```
## Not run:
V1 <- c(rep("01", 4), rep("02", 4), rep("03", 4), rep("00", 4))
V2 <- rep(c("01", "02", "03", "00" ), 4)
vals <- c(20, 50, 10, 80, 8, 19, 22, 49, 17, 32, 12, 61, 45, 101, 44, 190)
subtab <- data.frame(V1, V2, vals)
indexvars <- 1:2

sr <- roundSubtable(subtab, base=3, method="simple")
rr <- roundSubtable(subtab, base=4, method="random")
cr <- roundSubtable(subtab, indexvars, base=5, maxS=3, method="controlled")
print(cbind(subtab, simpleRounding=sr$val, randomRounding=rr$val, controlledRounding=cr$val))
## End(Not run)
```

---

summary.safeTable *summary.safeTable*

---

**Description**

summary method for objects of class safeTable.

**Usage**

```
## S3 method for class 'safeTable':
summary(object, ...)
```

**Arguments**

object            object from class safeTable  
 ...              additional parameters. Not used yet.

**Details**

object is an object of class `safeTable`. The summary functions returns several statistics from the anonymisation process.

**Value**

Manipulated data.

**Author(s)**

Bernhard Meindl

**Examples**

```
## Not run:
  data(exampleFullData)
  erg <- protectTable(exampleFullData, method="OPT")
  class(erg)
  summary(erg)

## End(Not run)
```

# Index

## \*Topic **datasets**

exampleFullData, 3

exampleMinimalData, 4

## \*Topic **methods**

createFullData, 1

processTableHITAS, 4

processTableHYPERCUBE, 6

protectTable, 7

roundSubtable, 8

## \*Topic **print**

summary.safeTable, 9

createFullData, 1

exampleFullData, 3

exampleMinimalData, 4

processTableHITAS, 4

processTableHYPERCUBE, 6

protectTable, 7

roundSubtable, 8

summary.safeTable, 9