# Package ‘easySdcTable’

September 10, 2021

**Type**  Package  

**Title**  Easy Interface to the Statistical Disclosure Control Package  ‘sdcTable’ Extended with the ‘GaussSuppression’ Method  

**Version**  0.9.0  

**Date**  2021-09-10  

**Author**  Øyvind Langsrud [aut, cre]  

**Maintainer**  Øyvind Langsrud <oyl@ssb.no>  

**Depends**  SSBtools  

**Imports**  sdcTable, shiny, methods, Matrix  

**VignetteBuilder**  knitr  

**Suggests**  knitr, rmarkdown, RegSDC, testthat (>= 2.1.0)  

**Description**  The main function, ProtectTable(), performs table suppression according to a frequency rule with a data set as the only required input. Within this function, protectTable(), protect_linked_tables() or runArgusBatchFile() in package 'sdcTable' is called. Lists of level-hierarchy (parameter 'dimList') and other required input to these functions are created automatically.  

The suppression method Gauss (default) is an additional method that is not available in 'sdcTable'. The function, PTgui(), starts a graphical user interface based on the 'shiny' package.  

**License**  Apache License 2.0 | file LICENSE  

**RoxygenNote**  7.1.1  

**Encoding**  UTF-8  

**URL**  https://github.com/statisticsnorway/easySdcTable  

**BugReports**  https://github.com/statisticsnorway/easySdcTable/issues  

**NeedsCompilation**  no  

**Repository**  CRAN  

**Date/Publication**  2021-09-10 13:30:05 UTC
R topics documented:

EasyData .......................................................... 2
ProtectTable ......................................................... 3
ProtectTable1 ......................................................... 8
PTgui ................................................................. 10
PTwrap ................................................................. 12
PTxyz ................................................................. 13

Index ........................................................................ 16

---

EasyData

Function that returns a dataset

Description

Function that returns a dataset

Usage

EasyData(dataset, path = NULL)

Arguments

dataset Name of data set within the easySdcTable package

path When non-NULL the data set is read from "path/dataset.RData"

Value

The dataset

Note

The function returns the same datasets as SSBtoolsData.

Examples

z <- EasyData("sosialFiktiv")
Easy interface to sdcTable: Table suppression according to a frequency rule.

**Description**

`GaussSuppression`, `protectTable` or `protect_linked_tables` is run with a data set as the only required input. One (stacked) or several (unstacked) input variables can hold cell counts. `ProtectTableData` is a tidy wrapper function, which returns a single data frame instead of a list (info omitted).

**Usage**

```r
ProtectTable(
  data,
  dimVar = 1:NCOL(data),
  freqVar = NULL,
  protectZeros = TRUE,
  maxN = 3,
  method = "Gauss",
  findLinked = TRUE,
  total = "Total",
  addName = FALSE,
  sep = "_",
  removeZeros = FALSE,
  dimList = NULL,
  groupVarInd = NULL,
  ind1 = NULL,
  ind2 = NULL,
  rowData = NULL,
  varNames = paste("var", 1:100, sep = ""),
  split = NULL,
  border = sep,
  revBorder = FALSE,
  freqName = "values",
  totalFirst = FALSE,
  numericOrder = TRUE,
  namesAsInput = TRUE,
  orderAsInput = TRUE,
  sortByReversedColumns = FALSE,
  doUnstack = TRUE,
  removeTotal = TRUE,
  singleOutput = NULL,
  suppression = NA,
  outFreq = "freq",
  outSdcStatus = "sdcStatus",
  outSuppressed = "suppressed",
  infoAsFrame = FALSE,
)```

IncProgress = IncDefault,
verbose = FALSE,
...
)

ProtectTableData(data, ...)

Arguments

data data frame

dimVar The main dimensional variables and additional aggregating variables (name or number).

freqVar Variable(s) holding counts or NULL in the case of micro data (name or number).

protectZeros When TRUE empty cells (count=0) is considered sensitive (i.e. same as allowZeros in primarySuppression).

maxN All cells having counts <= maxN are set as primary suppressed.

method Parameter method in protectTable, protect_linked_tables or wrapper methods via PTwrap. Gauss (default) is an additional method that is not available in sdcTable.

- "SIMPLEHEURISTIC": This method is default in protectable.
- "OPT", "HITAS", "HYPERCUBE": Other methods in protectable. "HYPERCUBE" is not possible in cases with two linked tables.
- "SimpleSingle": "SIMPLEHEURISTIC" with detectSingletons=TRUE when protectZeros=FALSE and "SIMPLEHEURISTIC" with threshold=1 (can be overridden by input) when protectZeros=TRUE.
- "Simple": "SIMPLEHEURISTIC" with detectSingletons=FALSE.
- "Gauss": GaussSuppression is called with parameters x, candidates, primary and singleton automatically generated. Other parameters (singletonMethod, printInc) can be specified.

Alternatively this parameter can be a named list specifying parameters for running tau-argus (see details). See PTwrap for other (experimental) wrapper methods (see details).

findLinked When TRUE, the function may find two linked tables and run protect_linked_tables.

total String used to name totals.

addName When TRUE the variable name is added to the level names, except for variables with most levels.

sep A character string to separate when addName apply and when creating variable names.

removeZeros When TRUE, rows with zero count will be removed from the data within the algorithm.

dimList By default, hierarchies will be automatically found from data (see FindDimLists). With non-NULL dimList, these will be changed. In practice this is done by the function ReplaceDimList.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupVarInd</td>
<td>Possible manual specification of list defining the hierarchical variable groups. When NULL (default) this information will be found automatically by <strong>FindTableGroup</strong>.</td>
</tr>
<tr>
<td>ind1</td>
<td>Coding of table 1 as indices referring to elements of groupVarInd. This information will be found automatically by <strong>FindTableGroup</strong> when groupVarInd=NULL.</td>
</tr>
<tr>
<td>ind2</td>
<td>Coding of table 2 as indices referring to elements of groupVarInd (as ind1 above).</td>
</tr>
<tr>
<td>rowData</td>
<td>Input to <strong>Stack</strong> used to generate extra dimVar variables when stacking in cases with several freqvar variables. When NULL rowData will be created automatically by <strong>AutoSplit</strong> using varNames (see below) and the the freqvar names as input.</td>
</tr>
<tr>
<td>varNames</td>
<td>The names of the extra dimVar variable(s) made when stacking in cases with several freqvar variables. When length(varNames)&gt;1 several variables may be found by <strong>AutoSplit</strong>.</td>
</tr>
<tr>
<td>split</td>
<td>Parameter to <strong>AutoSplit</strong> - see varNames and rowData above. When NULL (default), automatic splitting without needing a split string.</td>
</tr>
<tr>
<td>border</td>
<td>Parameter to <strong>AutoSplit</strong> - see varNames and rowData above.</td>
</tr>
<tr>
<td>revBorder</td>
<td>Parameter to <strong>AutoSplit</strong> - see varNames and rowData above.</td>
</tr>
<tr>
<td>freqName</td>
<td>Input to <strong>Stack</strong>. The name of the new freqvar variable obtained when stacking in cases with several input freqvar variables.</td>
</tr>
<tr>
<td>totalFirst</td>
<td>Parameter controlling how output is sorted.</td>
</tr>
<tr>
<td>numericOrder</td>
<td>Parameter controlling how output is sorted. Output is character but sorting can be based on the numeric input variables.</td>
</tr>
<tr>
<td>namesAsInput</td>
<td>When TRUE those output variables (created by unstacking) that correspond to input will be named as input.</td>
</tr>
<tr>
<td>orderAsInput</td>
<td>When TRUE output corresponding to input will be ordered as input and kept together as one block.</td>
</tr>
<tr>
<td>sortByReversedColumns</td>
<td>When TRUE output will be sorted by variables in opposite order.</td>
</tr>
<tr>
<td>doUnstack</td>
<td>When FALSE output will not be unstacked (in cases with sever input freqvar variables)</td>
</tr>
<tr>
<td>removeTotal</td>
<td>When TRUE the total string (see total above) will be removed from the names of output variables created by unstacking (in cases with sever input freqvar variables).</td>
</tr>
<tr>
<td>singleOutput</td>
<td>When TRUE output will be in as single data set. Default is FALSE for unstacked data (in cases with sever input freqvar variables) and otherwise TRUE.</td>
</tr>
<tr>
<td>suppression</td>
<td>Value used for suppressed elements in suppressed output data. Default is NA.</td>
</tr>
<tr>
<td>outFreq</td>
<td>String used to name output variable(s)</td>
</tr>
<tr>
<td>outSdcStatus</td>
<td>String used to name output variable(s)</td>
</tr>
<tr>
<td>outSuppressed</td>
<td>String used to name output variable(s)</td>
</tr>
<tr>
<td>infoAsFrame</td>
<td>When TRUE output element info is a data frame (useful in Shiny).</td>
</tr>
<tr>
<td>IncProgress</td>
<td>A function to report progress (incProgress in Shiny). Set equal to NULL to turn it off.</td>
</tr>
</tbody>
</table>
verbose Parameter sent to protectTable, protect_linked_tables or runArgusBatchFile.

Further parameters sent to protectTable (possibly via protect_linked_tables) such as solve_attackerprobs and timeLimit. Parameters to GaussSuppression, createArgusInput and PTwrap is also possible (see details).

Details

One or two tables are identified automatically and subjected to cell suppression by protectTable (single table) or protect_linked_tables (two linked tables). The tables can alternatively be specified manually by groupVarInd, ind1 and ind2. The output will be on a form similiar to input depending on whether freqVar is a single variable or not. The status of the cells are coded as "u" (primary suppressed), "x" (secondary suppression), and "s" (can be published). This is taken directly from the output from sdcTable. In cases with two linked tables "u" or "x" for common cells are based on output from the first table.

- **To run tau-argus** specify method as a named list containing the parameter exe for runArgusBatchFile and other parameters for createArgusInput.
  - One may specify: method = list(exe="C:/Tau/TauArgus.exe", typ="tabular", path= getwd(), solver= "FREE", method= "OPT") However these values of "exe", "path" and "solver" and "method" are set by default so in this case using "method = list(typ="tabular",method= "OPT")" is equivalent.
  - If typ="microdata" is specified. Necessary transformation to microdata will be made.

- **Wrapper methods (partly experimental):** In the function PTwrap several additional methods are defined. If input to ProtectTable() is one of these methods ProtectTable() will be run via PTwrap(). So making explicit call to PTwrap() is not needed.

- **Singleton and zeros:** The parameter detectSingletons was included in protecttable to handle the so-called singleton problem that appers when protectZeros=FALSE. Not all problems were solved and the parameter threshold has been introduced later. The value of threshold needed depends on the number of singletons in one group. It seems that threshold=3 is equivalent to detectSingletons=TRUE. When protectZeros=TRUE the related “zero problem” occurs. This problem is solved by threshold=1.

- **NOTE:** The use of numVarInd, weightInd and sampWeightInd in sdcTable is not implemented. This also limit possible input to tau-argus.

Value

When singleOutput=TRUE output is a list of two elements.

- **info:** Information as a single column character matrix. This is information about the extra dimVar variables created when stacking, information about the identified (linked) table(s) and summary output from sdcTable. With method="Gauss", a sdcTable function is run with maxN=0 to create a template for the real output. Some of the summary info is therefore misleading in this case.

- **data:** A data frame where variables are named according to outFreq, outSdcStatus and outSuppressed. When singleOutput=FALSE output element data is replaced by three elements and these are named according to outFreq, outSdcStatus and outSuppressed.
Note

ProtectTable makes a call to the function ProtectTable1.

See Also

See also the vignettes.

Examples

```r
# ==== Example 1 , 8 regions ====
z1 <- EasyData("z1")
ProtectTable(z1,1:2,3)
ProtectTableData(z1,1:2,3)
ProtectTable(z1, c("region","hovedint"), "ant") # Input by name
# --- Unstacked input data ---
z1w = EasyData("z1w")
ProtectTable(z1w, 1, 2:5)
ProtectTableData(z1w, 1, 2:5)
ProtectTable(z1w, 1, 2:5, varName="hovedint")
ProtectTable(z1w, 1, 2:5, method="HITAS")
ProtectTable(z1w, 1, 2:5, totalFirst = TRUE, method ="Simple")

# ==== Example 2 , 11 regions ====
z2 <- EasyData("z2")
ProtectTable(z2,c(1,3,4), 5) # With region-variable kostragr
# --- Unstacked input data ---
z2w <- EasyData("z2w")
ProtectTable(z2w, 1:2, 4:7) # With region-variable fylke
ProtectTable(z2w, 1:3, 4:7) # Two linked tables

## Not run:

# ==== Example 3 , 36 regions ====
z3 <- EasyData("z3")
ProtectTable(z3, c(1,4,5), 7) # Three dimensions. No subtotals
ProtectTable(z3, 1:6, 7) # Two linked tables
# --- Unstacked input data with coded column names
z3w <- EasyData("z3w")
ProtectTable(z3w,1:3,4:15, varName="g12") # coding not used when single varName
ProtectTable(z3w,1:3,4:15, varName=c("hovedint","mnd")) # Two variables found automatically
ProtectTable(z3w,1:3,4:15, varName=c("hovedint","mnd"),
removeTotal=FALSE) # Keep "Total" in variable names
# --- Unstacked input data with three level column name coding
z3wb <- EasyData("z3wb")
ProtectTable(z3wb,1:3,4:15,varName=c("hovedint","mnd","mnd2")) # Two variables found automatically
ProtectTable(z3wb,1:3,4:15,varName=c("hovedint","mnd","mnd2"),
split="_") # Three variables when splitting
ProtectTable(z3wb,1:3,4:15,varName=c("hovedint","mnd","mnd2"),
split="_",namesAsInput=FALSE,orderByInput=FALSE) # Alternative output format

# ==== Examples Tau-Argus ====
exeArgus <- "C:/TauArgus4.1.4/TauArgus.exe" # Change to TauArgus.exe-path in your computer
pathArgus <- "C:/Users/nnn/Documents" # Change to an existing folder
```
z1 = EasyData("z1")
ProtectTable(z1,1:2,3,method=list(exe=exeArgus, path=pathArgus, typ="tabular", method="OPT"))
ProtectTable(z1,1:2,3,method=list(exe=exeArgus, path=pathArgus, typ="tabular", method="MOD"))
ProtectTable(z1,1:2,3,method=list(exe=exeArgus, path=pathArgus, typ="tabular", method="GH"))
ProtectTable(z1,1:2,3,maxN=-1,
    method=list(path=pathArgus, exe=exeArgus, method="OPT",
        primSuppRules= list(list(type="freq", n=4, rg=20))))
z3w <- EasyData("z3")
ProtectTable(z3w,c(1:2,4,5),7,maxN=-1,
    method=list(path=pathArgus, exe=exeArgus, method="OPT",
        primSuppRules=list(list(type="freq", n=4, rg=20))))

# ==== Examples with parameter dimList =====
z2 <- EasyData("z2")
dList <- FindDimLists(z2[-5])
ProtectTable(z2[,c(1,4,5)], 1:2, 3, dimList = dList[c(1,3)])
ProtectTable(z2[,c(1,4,5)], 1:2, 3, dimList = dList[2])
ProtectTable(z2[,c(1,4,5)], 1:2, 3, dimList = DimList2Hrc(dList[c(2,3)]))

## End(Not run)

---

**ProtectTable1**  
*Easy input interface to sdcTable*

**Description**

`protectTable` or `protect_linked_tables` is run with a data set at the only required input.

**Usage**

```r
ProtectTable1(data,
    dimVarInd = 1:ncol(data),
    freqVarInd = NULL,
    protectZeros = TRUE,
    maxN = 3,
    method = "SIMPLEHEURISTIC",
    findLinked = TRUE,
    total = "Total",
    addName = FALSE,
    sep = ".",
    removeZeros = FALSE,
    dimList = NULL,
    groupVarInd = NULL,
    ind1 = NULL,
    ind2 = NULL,
    dimDataReturn = FALSE,
)```
IncProgress = IncDefault, verbose = FALSE, ...
)

Arguments

- **data**: Matrix or data frame
- **dimVarInd**: Column-indices of the main dimensional variables and additional aggregating variables.
- **freqVarInd**: Column-indices of a variable holding counts or NULL in the case of microdata.
- **protectZeros**: When TRUE empty cells (count=0) is considered sensitive (i.e. same as allowZeros in primarySuppression).
- **maxN**: All cells having counts <= maxN are set as primary suppressed.
- **method**: Parameter "method" in protectTable or protect_linked_tables. Alternatively a list defining parameters for running tau-argus (see ProtectTable).
- **findLinked**: When TRUE, the function may find two linked tables and run protect_linked_tables.
- **total**: String used to name totals.
- **addName**: When TRUE the variable name is added to the level names, except for variables with most levels.
- **sep**: A character string to separate when addName apply.
- **removeZeros**: When TRUE, rows with zero count will be removed from the data.
- **dimList**: See ProtectTable.
- **groupVarInd**: Possible manual specification if list defining the hierarchical variable groups
- **ind1**: Coding of table 1 as indices referring to elements of groupVarInd
- **ind2**: Coding of table 2 as indices referring to elements of groupVarInd
- **dimDataReturn**: When TRUE a data frame containing the dimVarInd variables is returned
- **IncProgress**: A function to report progress (incProgress in Shiny).
- **verbose**: Parameter sent to protectTable, protect_linked_tables or runArgusBatchFile.
- **...**: Further parameters sent to protectTable, protect_linked_tables or createArgusInput.

Details

One or two tables are identified automatically and subjected to cell suppression methods in package sdcTable. The tables can alternatively be specified manually by groupVarInd, ind1 and ind2 (see FindTableGroup).

Value

Output is a list of three elements.

**table1** consists of the following elements:
**secondary**  Output from `protectTable` or first element of output from `protect_linked_tables` or output from `runArgusBatchFile`.

**primary**  Output from `primarySuppression`.

**problem**  Output from `makeProblem`.

**dimList**  Generated input to `makeProblem`.

**ind**  Indices referring to elements of groupVarInd in the output element common.

**table2** consists of elements of the same type as `table1` in cases of two linked tables. Otherwise `table2` is NULL.

**common** consists of the following elements:

- `commonCells`  Input to `protect_linked_tables`.
- `groupVarInd`  List defining the hierarchical variable groups
- `info`  A table summarizing the tables using variable names
- `nLevels`  The number of levels of each variable (only when groupVarInd input is NULL)
- `dimData`  Data frame containing the dimVarInd variables when dimDataReturn=TRUE. Otherwise NULL.

**See Also**

- `ProtectTable`, `HierarchicalGroups`, `FactorLevCorr`, `FindDimLists`, `FindCommonCells`

**Examples**

```r
# Not run:
z2 <- EasyData("z2")
a <- ProtectTable1(z2, c(1, 3, 4), 5)
head(as.data.frame(sdcTable::getInfo(a[[1]][[1]], type = "finalData"))) # The table (not linked)

z3 <- EasyData("z3")
b <- ProtectTable1(z3, 1:6, 7)
head(as.data.frame(sdcTable::getInfo(b[[1]][[1]], type = "finalData"))) # First table
head(as.data.frame(sdcTable::getInfo(b[[2]][[1]], type = "finalData"))) # Second table

# End(Not run)
```

---

**PTgui**

---

**Table suppression - Shiny Gui**

**Description**

Run PTgui from the R console or use PTguiApp to make a server application
Usage

PTgui(
    data = NULL,
    language = "English",
    exeArgus = NULL,
    pathArgus = getwd(),
    maxNchoices = c(1:10, 12, 15, 20),
    ...
)

PTguiApp(
    language = "English",
    exeArgus = NULL,
    pathArgus = "",
    maxNchoices = c(1:10, 12, 15, 20),
    ...
)

PTguiNO(
    data = NULL,
    language = "Norwegian",
    exeArgus = NULL,
    pathArgus = getwd(),
    maxNchoices = c(1:10, 12, 15, 20),
    ...
)

PTguiAppNO(
    language = "Norwegian",
    exeArgus = NULL,
    pathArgus = "",
    maxNchoices = c(1:10, 12, 15, 20),
    ...
)

Arguments

data            NULL or a data.frame
language        Menu language, "English" or "Norwegian".
exeArgus        Tau-argus executable
pathArgus       Folder for (temporary) tau-argus files
maxNchoices     Choices of maxN
...              Further parameters sent to ProtectTable

Value

Output from ProtectTable. The output is returned invisibly (via invisible) which means that it is not automatically printed to the console.
PTguiApp(): New for server

Examples

```r
## Not run:
#
# Start the gui.
PTgui()

# Start Norwegian gui with example data and catch output
out <- PTguiNO(data=EasyData("z1w"))

# Note: Change to TauArgus.exe-path in your computer
exeArgus <- "C:/TauArgus4.2.0b2/TauArgus.exe"

# Note: Change to an existing folder
pathArgus <- "C:/Users/nnn/Documents"

# Start the gui with possibility to run tau-argus.
PTgui(exeArgus=exeArgus, pathArgus=pathArgus)
```

## End(Not run)

---

PTwrap

Wrapper to ProtectTable() with additional methods (partly experimental)

Description

Additional values of "method" is possible. Each new method (wrapper method) will make a call to ProtectTable() using a specific parameter setting.

Usage

```r
PTwrap(
   ..., 
   maxN = 3, 
   method = "SimpleSingle", 
   exeArgus = "C:/Tau/TauArgus.exe", 
   pathArgus = getwd(), 
   solverArgus = "FREE", 
   methodArgus = "OPT", 
   rgArgus = 0 
)
```
Arguments

... Parameters to ProtectTable
maxN Parameter to ProtectTable or a wrapper method (see details)
method Parameter to runArgusBatchFile
exeArgus Parameter to createArgusInput
pathArgus Parameter "solver" to createArgusInput
solverArgus Parameter "method" to createArgusInput
methodArgus Parameter "rg" in "primSuppRules" in createArgusInput
rgArgus Parameter "primSuppRules" in createArgusInput

Details

The wrapper methods are:

Simple: "SIMPLEHEURISTIC" with detectSingletons=FALSE

SimpleSingle: "SIMPLEHEURISTIC" with detectSingletons=TRUE when protectZeros=FALSE and "SIMPLEHEURISTIC" with threshold=1 (can be overridden by input) when protectZeros=TRUE

SimpleSingleOld: "SIMPLEHEURISTIC" with detectSingletons=TRUE

TauArgus: Tau-argus will be run according to the settings of the other input parameters.

Using rgArgus=0 is equivalent to calling ProtectTable() with
method = list(exe=exeArgus, typ="tabular", path=pathArgus, solver=solverArgus, method=methodArgus))

Other values of rgArgus is equivalent to calling ProtectTable() with
method = list(exe=exeArgus, typ="microdata", path=pathArgus, solver=solverArgus, method=methodArgus, primSuppRules=list(list(type="freq", n=maxN+1, rg=rgArgus )))))

TauArgusOPT: As "TauArgus" with methodArgus="OPT"

TauArgusMOD: As "TauArgus" with methodArgus="MOD"

TauArgusGH: As "TauArgus" with methodArgus="GH"

Value

See ProtectTable


Description

Assuming correct suppression, suppressed values become decimal numbers (not whole numbers) instead of missing.
Usage

\texttt{PTxyz(data, dimVar, freqVar, \ldots)}

Arguments

data \hspace{1cm} \text{data frame}
dimVar \hspace{1.5cm} \text{The main dimensional variables and additional aggregating variables (name or number).}
freqVar \hspace{1.5cm} \text{Variable(s) holding counts (name or number).}
\ldots \hspace{1.5cm} \text{Further parameters sent to ProtectTable}

Details

Within this R package this function will be used for testing

Value

List of three matrices ready as input to SuppressDec

\begin{itemize}
  \item \textit{x} \hspace{1cm} \text{Sparse dummy matrix where the dimensions match \texttt{z} and \texttt{y}.}
  \item \textit{z} \hspace{1cm} \text{Frequencies to be published with suppressed as NA.}
  \item \textit{y} \hspace{1cm} \text{Inner cell frequencies.}
\end{itemize}

Author(s)

Øyvind Langsrud

Examples

\begin{verbatim}
# Same examples as in ProtectTable
a1 <- PTxyz(EasyData("z1"), c("region", "hovedint"), "ant")
a2 <- PTxyz(EasyData("z2"), c(1,3,4), 5)

if (require(RegSDC)) { # RegSDCdata and SuppressDec
  # Same data as in RegSDCdata examples (and in paper)
  data7 <- RegSDCdata("sec7data")
  data7 <- data7[!is.na(data7$y), 1:3]
  data7

  # Generate \texttt{x}, \texttt{y}, \texttt{z} similar to \texttt{xAll}, \texttt{y}, \texttt{zAllSupp} in RegSDCdata examples
  # But different suppressed cells and \texttt{z} ordered differently
  a <- PTxyz(data7, 1:2, 3, maxN = 3, method = "HITAS")
  a

  # Suppressed inner cells as decimal numbers
  yDec <- SuppressDec(a$x, a$z, a$y, rmse = 1)
  yDec

  # All cells as decimal numbers
\end{verbatim}
cbind(a$z, t(a$x) %*% cbind(a$y, yDec))

# As ProtectTable example
z1 <- EasyData("z1")
a <- PTxyz(z1, c("region", "hovedint"), "ant")

# Inner cells as decimal numbers. 3 replicates.
yDec <- SuppressDec(a$x, a$z, a$y, nRep = 3)
yDec

# All cells with 3 replicates.
cbind(a$z, t(a$x) %*% cbind(a$y, yDec))

## Not run:
if (require(RegSDC)) {
  # An example involving two linked tables.
  # It is demonstrated that the SIMPLEHEURISTIC approach to suppression is not safe.
  # That is, perfect fit (whole numbers) for some suppressed cells.
  a <- PTxyz(EasyData("z3"), 1:5, 7, method = "SIMPLEHEURISTIC", protectZeros= FALSE)
  cbind(a$z, t(a$x) %*% cbind(a$y, SuppressDec(a$x, a$z, rmse=pi/3, nRep=3)))[which(is.na(a$z)),]
}
## End(Not run)
Index

AutoSplit, 5
createArgusInput, 6, 13
EasyData, 2
FactorLevCorr, 10
FindCommonCells, 10
FindDimLists, 4, 10
FindTableGroup, 5, 9
GaussSuppression, 3, 4, 6
HierarchicalGroups, 10
invisible, 11
makeProblem, 10
primarySuppression, 4, 10
protect_linked_tables, 3, 4, 6, 10
ProtectTable, 3, 9–11, 13, 14
protectTable, 3, 4, 6, 10
ProtectTable1, 7, 8
ProtectTableData (ProtectTable), 3
PTgui, 10
PTguiApp (PTgui), 10
PTguiAppNO (PTgui), 10
PTguiNO (PTgui), 10
PTwrap, 4, 6, 12
PTxyz, 13

ReplaceDimList, 4
runArgusBatchFile, 6, 10, 13

SSBtoolsData, 2
Stack, 5