

# Package ‘easySdcTable’

May 20, 2021

**Type** Package

**Title** Easy Interface to the Statistical Disclosure Control Package  
'sdcTable'

**Version** 0.7.0

**Date** 2021-05-20

**Author** Øyvind Langsrud [aut, cre]

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

**Depends** SSBtools

**Imports** sdcTable, shiny, methods

**VignetteBuilder** knitr

**Suggests** knitr, rmarkdown, RegSDC, Matrix, 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(), protectLinkedTables() or runArgusBatchFile() in package 'sdcTable' is called. Lists of level-hierarchy (parameter 'dimList') and other required input to these functions are created automatically.  
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** 2021-05-20 06:00:02 UTC

## R topics documented:

EasyData	2
ProtectTable	3
ProtectTable1	8
PTgui	10
PTwrap	12
PTxyz	13
<b>Index</b>	<b>16</b>

---

EasyData	<i>Function that returns a dataset</i>
----------	--

---

### 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("socialFiktiv")
```

---

ProtectTable	<i>Easy interface to sdcTable: Table suppression according to a frequency rule.</i>
--------------	---

---

### Description

protectTable() or protectLinkedTables() in package 'sdcTable' is run with a data set as the only required input. One (stacked) or several (unstacked) input variables can hold cell counts. Output is on a form similar to input.

### Usage

```
ProtectTable(  
  data,  
  dimVar = 1:NCOL(data),  
  freqVar = NULL,  
  protectZeros = TRUE,  
  maxN = 3,  
  method = "SimpleSingle",  
  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,
    ...
)

```

### 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 <a href="#">primarySuppression</a> ).
maxN	All cells having counts <= maxN are set as primary suppressed.
method	<p>Parameter method in <a href="#">protectTable</a>, <a href="#">protectLinkedTables</a> or wrapper methods via <a href="#">PTwrap</a>. Gauss is an additional method that is not available in <a href="#">sdTable</a>.</p> <ul style="list-style-type: none"> <li>"SIMPLEHEURISTIC": This method is default in <a href="#">protectable</a>.</li> <li>"OPT", "HITAS", "HYPERCUBE": Other methods in <a href="#">protectable</a>. "HYPERCUBE" is not possible in cases with two linked tables.</li> <li>"SimpleSingle" (<b>default</b>): "SIMPLEHEURISTIC" with detectSingletons=TRUE when protectZeros=FALSE and "SIMPLEHEURISTIC" with threshold=1 (can be overridden by input) when protectZeros=TRUE.</li> <li>"Simple": "SIMPLEHEURISTIC" with detectSingletons=FALSE.</li> <li>"Gauss": <a href="#">GaussSuppression</a> is called with parameters x, candidates, primary and singleton automatically generated. Other parameters (singletonMethod, printInc) can be specified.</li> </ul> <p>Alternatively this parameter can be a named list specifying parameters for running tau-argus (see details). See <a href="#">PTwrap</a> for other (experimental) wrapper methods (see details).</p>
findLinked	When TRUE, the function may find two linked tables and run <a href="#">protectLinkedTables</a> .
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 <a href="#">FindDimLists</a> ). With non-NULL dimList, these will be changed. In practice this is done by the function <a href="#">ReplaceDimList</a> .
groupVarInd	Possible manual specification of list defining the hierarchical variable groups. When NULL (default) this information will be found automatically by <a href="#">FindTableGroup</a> .
ind1	Coding of table 1 as indices referring to elements of groupVarInd. This information will be found automatically by <a href="#">FindTableGroup</a> when groupVarInd=NULL.

ind2	Coding of table 2 as indices referring to elements of groupVarInd (as ind1 above).
rowData	Input to <a href="#">Stack</a> used to generate extra dimVar variables when stacking in cases with several freqvar variables. When NULL rowData will be created automatically by <a href="#">AutoSplit</a> using varNames (see below) and the the freqvar names as input.
varNames	The names of the extra dimVar variable(s) made when stacking in cases with several freqvar variables. When length(varNames)>1 several variables may be found by <a href="#">AutoSplit</a> .
split	Parameter to <a href="#">AutoSplit</a> - see varNames and rowData above. When NULL (default), automatic splitting without needing a split string.
border	Parameter to <a href="#">AutoSplit</a> - see varNames and rowData above.
revBorder	Parameter to <a href="#">AutoSplit</a> - see varNames and rowData above..
freqName	Input to <a href="#">Stack</a> . The name of the new freqvar variable obtained when stacking in cases with several input freqvar variables.
totalFirst	Parameter controlling how output is sorted.
numericOrder	Parameter controlling how output is sorted. Output is character but sorting can be based on the numeric input variables.
namesAsInput	When TRUE those output variables (created by unstacking) that correspond to input will be named as input.
orderAsInput	When TRUE output corresponding to input will be ordered as input and kept together as one block.
sortByReversedColumns	When TRUE output will be sorted by variables in opposite order.
doUnstack	When FALSE output will not be unstacked (in cases with sever input freqvar variables)
removeTotal	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).
singleOutput	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.
suppression	Value used for suppressed elements in suppressed output data. Default is NA.
outFreq	String used to name output variable(s)
outSdcStatus	String used to name output variable(s)
outSuppressed	String used to name output variable(s)
infoAsFrame	When TRUE output element info is a data frame (useful in Shiny).
IncProgress	A function to report progress (incProgress in Shiny). Set equal to NULL to turn it off.
...	Further parameters sent to <a href="#">protectTable</a> (possibly via <a href="#">protectLinkedTables</a> ) such as verbose (print output while calculating) and timeLimit. Parameters to <a href="#">GaussSuppression</a> , <a href="#">createArgusInput</a> and <a href="#">PTwrap</a> is also possible (see details).

## Details

One or two tables are identified automatically and subjected to cell suppression by `protectTable` (single table) or `protectLinkedTables` (two linked tables). The tables can alternatively be specified manually by `groupVarInd`, `ind1` and `ind2`. The output will be on a form similar 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 appears 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`.
- **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

```

# ==== Example 1 , 8 regions ====
z1 <- EasyData("z1")
ProtectTable(z1,1:2, 3)
ProtectTable(z1,1:2, 3, method="Gauss")$data
ProtectTable(z1, c("region","hovedint"), "ant") # Input by name
# --- Unstacked input data ---
z1w = EasyData("z1w")
ProtectTable(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, method = "Simple") # With region-variable fylke
ProtectTable(z2w, 1:3, 4:7, method = "SIMPLEHEURISTIC") # 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, method = "SIMPLEHEURISTIC") # Two linked tables
# --- Unstacked input data with coded column names
z3w <- EasyData("z3w")
ProtectTable(z3w,1:3,4:15, varName="g12", method = "Simple") # 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"),
             method = "Simple", removeTotal=FALSE) # Keep "Total" in variable names
# --- Unstacked input data with three level column name coding
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"),
             method = "Simple", split="_") # Three variables when splitting
ProtectTable(z3wb,1:3,4:15,varName=c("hovedint", "mnd", "mnd2"), method = "SIMPLEHEURISTIC",
             split="_",namesAsInput=FALSE,orderAsInput=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(z3,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)))

## End(Not run)

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

```

---

ProtectTable1

*Easy input interface to sdcTable*


---

### Description

protectTable or protectLinkedTables is run with a data set as the only required input.

### Usage

```

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,
  ...
)

```

### 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 micro data.
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 protectLinkedTables. Alternatively a list defining parameters for running tau-argus (see <a href="#">ProtectTable</a> ).
findLinked	When TRUE, the function may find two linked tables and run protectLinkedTables.
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 <a href="#">ProtectTable</a> .
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).
...	Further parameters sent to protectTable, protectLinkedTables 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 <a href="#">protectTable</a> or first element of output from <a href="#">protectLinkedTables</a> or output from <a href="#">runArgusBatchFile</a> .
primary	Output from <a href="#">primarySuppression</a> .
problem	Output from <a href="#">makeProblem</a> .
dimList	Generated input to <a href="#">makeProblem</a> .
ind	Indices referring to elements of <code>groupVarInd</code> 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 protectLinkedTables.
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**

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

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

### Note

PTguiApp(): New for server

### Examples

```

## 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	<i>Wrapper to ProtectTable() with additional methods (partly experimental)</i>
--------	--

---

## Description

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

## Usage

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

## Arguments

...	Parameters to <code>ProtectTable</code>
maxN	Parameter to <code>ProtectTable</code>
method	Parameter to <code>ProtectTable</code> or a wrapper method (see details)
exeArgus	Parameter to <a href="#">runArgusBatchFile</a>
pathArgus	Parameter to <a href="#">createArgusInput</a>
solverArgus	Parameter "solver" to <a href="#">createArgusInput</a>
methodArgus	Parameter "method" to <a href="#">createArgusInput</a>
rgArgus	Parameter "rg" in "primSuppRules" in <a href="#">createArgusInput</a>

## 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](#)

---

 PTxyz

---

*ProtectTable with output ready for SuppressDec in package RegSDC*


---

## Description

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

## Usage

```
PTxyz(data, dimVar, freqVar, ...)
```

## Arguments

data	data frame
dimVar	The main dimensional variables and additional aggregating variables (name or number).
freqVar	Variable(s) holding counts (name or number).
...	Further parameters sent to <a href="#">ProtectTable</a>

**Details**

Within this r package this function will be used for testing

**Value**

List of three matrices ready as input to SuppressDec

x	Sparse dummy matrix where the dimensions match z and y.
z	Frequencies to be published with suppressed as NA.
y	Inner cell frequencies.

**Author(s)**

Øyvind Langsrud

**Examples**

```
# 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 x, y, z similar to xAll, y, zAllSupp in RegSDCdata examples
  # But different suppressed cells and 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
  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))
}
```

```
# An example involving two linked tables.  
# It is demonstrated that the approach to suppression is not safe.  
# That is, perfect fit (whole numbers) for some suppressed cells.  
a <- PTxyz(EasyData("z2"), 1:4, 5)  
cbind(a$z, t(a$x) %*% cbind(a$y, SuppressDec(a$x, a$z, rmse = 1, nRep = 3)))[which(is.na(a$z)), ]  
}
```

# Index

AutoSplit, [5](#)

createArgusInput, [5](#), [6](#), [12](#)

EasyData, [2](#)

FactorLevCorr, [10](#)

FindCommonCells, [10](#)

FindDimLists, [4](#), [10](#)

FindTableGroup, [4](#), [9](#)

GaussSuppression, [4](#), [5](#)

HierarchicalGroups, [10](#)

invisible, [11](#)

makeProblem, [9](#)

primarySuppression, [4](#), [9](#)

protectLinkedTables, [4–6](#), [9](#)

ProtectTable, [3](#), [9–11](#), [13](#)

protectTable, [4–6](#), [9](#)

ProtectTable1, [6](#), [8](#)

PTgui, [10](#)

PTguiApp (PTgui), [10](#)

PTguiAppNO (PTgui), [10](#)

PTguiNO (PTgui), [10](#)

PTwrap, [4–6](#), [12](#)

PTxyz, [13](#)

ReplaceDimList, [4](#)

runArgusBatchFile, [6](#), [9](#), [12](#)

SSBtoolsData, [2](#)

Stack, [5](#)