# Package 'GaussSuppression'

April 5, 2022

**Type** Package

**Title** Tabular Data Suppression using Gaussian Elimination

**Version** 0.2.0

**Date** 2022-04-05

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

**Depends** Matrix

**Imports** SSBtools, RegSDC, stats, methods

**Description**
A statistical disclosure control tool to protect tables by suppression using the Gaussian elimination secondary suppression algorithm. Primary suppression functions for the minimum frequency rule, the dominance rule and a directly-disclosive rule are included. General primary suppression functions can be supplied as input. Suppressed frequencies can be replaced by synthetic decimal numbers as described in Langsrud (2019) <doi:10.1007/s11222-018-9848-9>.

**License** Apache License 2.0

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

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

**Encoding** UTF-8

**RoxygenNote** 7.1.2

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Øyvind Langsrud [aut, cre],
Daniel Lupp [aut],
Hege Bøvelstad [ctb]

**Repository** CRAN

**Date/Publication** 2022-04-05 13:12:30 UTC

# R topics documented:

AdditionalSuppression . . . . . 2
ChainedSuppression . . . . . 4
DominanceRule . . . . . 6
GaussSuppressDec . . . . . 7
GaussSuppressionFromData . . . . . 9
GaussSuppressionTwoWay . . . . . 12
LazyLinkedTables . . . . . 16
MaxContribution . . . . . 17
Ncontributors . . . . . 18
NcontributorsHolding . . . . . 19
PrimaryFromSuppressedData . . . . . 20
SuppressDirectDisclosure . . . . . 22
SuppressionFromDecimals . . . . . 23

**Index** **25**

---

`AdditionalSuppression` *GaussSuppression from data and suppressed data*

---

## Description

Extended version of `GaussSuppressionFromData` that takes into account suppression pattern in suppressed data sent as input

## Usage

```
AdditionalSuppression(
  data,
  ...,
  primary = PrimaryDefault,
  suppressedData = NULL,
  makePrimary = TRUE,
  makeForced = TRUE,
  forceNotPrimary = TRUE
)
```

## Arguments

| | |
|---|---|
| `data` | Input data as a data frame |
| `...` | Further parameters to `GaussSuppressionFromData` |
| `primary` | As input to `GaussSuppressionFromData` before possible extension caused by `suppressedData`. Supply `NULL` if all primary suppressions are retrieved form `suppressedData`. |
| `suppressedData` | A data frame or a list of data frames as output from `GaussSuppressionFromData`. |
| `makePrimary` | When `TRUE`, suppression in `suppressedData` is preserved. |

makeForced When TRUE, non-suppression in suppressedData is preserved. An exception is possible primary suppression which has priority over forced. Use forceNotPrimary to avoid this exception.

forceNotPrimary
When TRUE, non-suppression in suppressedData is forced to be not primary suppressed.

## Details

This function is an easy alternative to using PrimaryFromSuppressedData and the relating functions manually. See the examples of PrimaryFromSuppressedData. By default, the suppression pattern in suppressedData is preserved. The behavior can be tuned by the parameters.

Note that the variables used in suppressedData in addition to "suppressed" are those with matching names in crossTable. Others are ignored. See examples (d3, d4, d5). NOW A FIX IS INCLUDED by attribute totCode. EXAMPLES NOT YET CHANGED.

## Value

Aggregated data with suppression information

## Examples

```
z1 <- SSBtoolsData("z1")
z2 <- SSBtoolsData("z2")
z3 <- SSBtoolsData("z3")

# Ordinary suppressions
a <- GaussSuppressionFromData(z1, 1:2, 3, maxN = 5)
b <- GaussSuppressionFromData(z2, 1:4, 5, maxN = 1)

# As b and also suppression pattern in a preserved
b1 <- AdditionalSuppression(z2, 1:4, 5, maxN = 1, suppressedData = a)

# Rows with differences
cbind(b, b1)[b1$suppressed != b$suppressed, ]

# All primary from a
b2 <- AdditionalSuppression(z2, 1:4, 5, suppressedData = a, primary = NULL, singleton = NULL)

# Rows with suppression
b2[b2$suppressed, ]

# All primary from b2
d1 <- AdditionalSuppression(data = z3, 1:6, 7, suppressedData = b2, primary = NULL,
                            singleton = NULL)

# No suppression since no common codes
d1[d1$suppressed, ]

# Use another coding of fylke
```

```
z3$fylke_ <- z3$fylke - 4
d2 <- AdditionalSuppression(data = z3, c(1, 3:6, 8), 7, suppressedData = b2, primary = NULL,
                            singleton = NULL)

# Two primary found in b2 -> several secondary
d2[d2$suppressed,]


# Examples demonstrating limitations of AdditionalSuppression
# Variable mnd in suppressedData is not used

# No suppression since unsuppressed rows used by makeForced and forceNotPrimary
d3 <- AdditionalSuppression(data = z3, c(1, 3:4, 8), 7, suppressedData = d2, primary = NULL,
                            singleton = NULL)
d3[d3$suppressed, ]

# Now suppression, but not too much
d4 <- AdditionalSuppression(data = z3, c(1, 3:4, 8), 7, suppressedData = d2,
                            forceNotPrimary = FALSE, primary = NULL, singleton = NULL)
d4[d4$suppressed, ]

# The correct way is to limit the input
d5 <- AdditionalSuppression(data = z3, c(1, 3:4, 8), 7, suppressedData = d2[d2$mnd == "Total", ],
                            primary = NULL, singleton = NULL)
d5[d5$suppressed, ]
```

---

`ChainedSuppression` *Repeated GaussSuppression with forwarding of previous results*

---

## Description

`AdditionalSuppression` is called several times. Each time with all previous results as `suppressedData`.

## Usage

```
ChainedSuppression(..., withinArg = NULL)

ChainedSuppressionHi(..., hierarchies)

ChainedSuppressionHi1(..., hierarchies)
```

## Arguments

| | |
|---|---|
| `...` | Arguments to `AdditionalSuppression`/`GaussSuppressionFromData` that are kept constant. |
| `withinArg` | A list of named lists. Arguments to `AdditionalSuppression`/`GaussSuppressionFromData` that are not kept constant. List elements with suppressed data are also allowed. |

`hierarchies` In the wrapper `ChainedSuppressionHi`, this argument will be used to generate the `withinArg` to `ChainedSuppression` with the same length (see examples). Then, element number `i` of `withinArg` is `list(hierarchies = hierarchies[1:i])`. In the similar wrapper, `ChainedSuppressionHi1`, `withinArg` has always two elements: `list(hierarchies = hierarchies[1])` and `list(hierarchies = hierarchies)`.

## Value

List of data frames. The wrappers, `ChainedSuppressionHi` and `ChainedSuppressionHi1`, return a single data frame, which is the last list item.

## Examples

```
z1 <- SSBtoolsData("z1")
z2 <- SSBtoolsData("z2")
z2b <- z2[3:5]
names(z2b)[1] <- "region"

# As GaussSuppressionFromData when a single element within withinArg
a1 <- ChainedSuppression(z1, 1:2, 3, maxN = 5)
a2 <- ChainedSuppression(z1, withinArg = list(list(dimVar = 1:2, freqVar = 3, maxN = 5)))
identical(a1, a2[[1]])

# b[[3]] include results from b[[1]] and b[[2]]
b <- ChainedSuppression(z1, freqVar = 3, withinArg = list(
       list(dimVar = 1,   maxN = 55),
       list(dimVar = 2,   maxN = 55),
       list(dimVar = 1:2, maxN = 5)))

# d[[2]] is same as b1 in AdditionalSuppression examples
d <- ChainedSuppression(withinArg = list(
       list(data = z1,  dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2,  dimVar = 1:4, freqVar = 5, maxN = 1)))

# Common variable names important.
# Therefore kostragr renamed to region in z2b.
f <- ChainedSuppression(withinArg = list(
       list(data = z1,  dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2b, dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2,  dimVar = 1:4, freqVar = 5, maxN = 1)))

# Parameters so that only suppressions are forwarded.
# This is first iteration in linked tables by iterations.
e <- ChainedSuppression(withinArg = list(
       list(data = z1,  dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2b, dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2,  dimVar = 1:4, freqVar = 5, maxN = 1)),
       makeForced = FALSE, forceNotPrimary = FALSE)

# "A" "annet"/"arbeid" could be suppressed here, but not in f since f[[1]]
```

```
e[[3]][which(e[[3]]$suppressed != f[[3]]$suppressed), ]


#### Demonstrate SuppressionByChainedHierarchies

dimLists <- SSBtools::FindDimLists(z2[, 4:1])


# Two ways of doing the same calculations
g1 <- ChainedSuppressionHi(z2, c(1, 3), 5, maxN = 1, hierarchies = dimLists)
g1b <-  ChainedSuppression(z2, c(1, 3), 5, maxN = 1, withinArg = list(
         list(hierarchies = dimLists[1]),
         list(hierarchies = dimLists[1:2]),
         list(hierarchies = dimLists[1:3])))[[3]]

# Results different after combining hierarchies
g2 <- ChainedSuppressionHi(z2, c(1, 3), 5, maxN = 1,
         hierarchies = SSBtools::AutoHierarchies(dimLists))

# In this case, the same results can be obtained by:
g3 <- ChainedSuppressionHi1(z2, c(1, 3), 5, maxN = 1, hierarchies = dimLists)
```

---

`DominanceRule` *Dominance* (n,k) *rule for magnitude tables*

---

## Description

Supports application of multiple values for `n` and `k`. The function works on magnitude tables containing negative cell values by calculating contribution based on absolute values.

## Usage

```
DominanceRule(
  data,
  x,
  crossTable,
  numVar,
  n,
  k,
  protectZeros = FALSE,
  charVar,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | the dataset |
| `x` | ModelMatrix generated by parent function |

| | |
|---|---|
| crossTable | crossTable generated by parent function |
| numVar | vector containing numeric values in the data set |
| n | parameter n in dominance rule. |
| k | parameter k in dominance rule. |
| protectZeros | parameter determining whether cells with value 0 should be suppressed. |
| charVar | Variable in data holding grouping information. Dominance will be calculated after aggregation within these groups. |
| ... | unused parameters |

## Details

This methodn only supports suppressing a single numeric variable.

## Value

logical vector that is TRUE in positions corresponding to cells breaching the dominance rules.

## Author(s)

Daniel Lupp

---

# GaussSuppressDec *Cell suppression with synthetic decimal numbers*

---

## Description

GaussSuppressionFromData is run and decimal numbers are added to output by a modified (for sparse matrix efficiency) version of SuppressDec.

## Usage

```
GaussSuppressDec(
  data,
  ...,
  output = NULL,
  digits = 9,
  nRep = NULL,
  rmse = pi/3,
  sparseLimit = 500,
  rndSeed = 123,
  runIpf = FALSE,
  eps = 0.01,
  iter = 100,
  mismatchWarning = TRUE,
  whenDuplicatedInner = NULL,
  whenMixedDuplicatedInner = warning
)
```

## Arguments

| | |
|---|---|
| data | Input daata as a data frame |
| ... | Further parameters to GaussSuppressionFromData |
| output | NULL (default), "publish", "inner", "publish_inner", or "publish_inner_x" (x also). |
| digits | Parameter to RoundWhole. Values close to whole numbers will be rounded. |
| nRep | NULL or an integer. When >1, several decimal numbers will be generated. |
| rmse | Desired root mean square error of decimal numbers. Variability around the expected, according to the linear model, inner frequencies. The expected frequencies are calculated from the non-suppressed publishable frequencies. |
| sparseLimit | Limit for the number of rows of a reduced x-matrix within the algorithm. When exceeded, a new sparse algorithm is used. |
| rndSeed | If non-NULL, a random generator seed to be used locally within the function without affecting the random value stream in R. |
| runIpf | When TRUE, additional frequencies are generated by iterative proportional fitting using Mipf. |
| eps | Parameter to Mipf. |
| iter | Parameter to Mipf. |
| mismatchWarning | Whether to produce the warning "Mismatch between whole numbers and suppression", when relevant. When nRep>1, all replicates must satisfy the whole number requirement for non-suppressed cells. When mismatchWarning is integer (>0), this will be used as parameter digits to RoundWhole when doing mismatch checking (can be quite low when nRep>1). |
| whenDuplicatedInner | Function to be called when default output and when cells marked as inner correspond to several input cells (aggregated) since they correspond to published cells. |
| whenMixedDuplicatedInner | Function to be called in the case above when some inner cells correspond to published cells (aggregated) and some not (not aggregated). |

## Value

A data frame where inner cells and cells to be published are combined or output according to parameter output.

## Author(s)

Øyvind Langrsud

## Examples

```
z1 <- SSBtoolsData("z1")
GaussSuppressDec(z1, 1:2, 3)
GaussSuppressDec(z1, freqVar = "ant", formula = ~ region + hovedint, maxN = 10)
```

GaussSuppressionFromData *Cell suppression from input data containing inner cells*

## Description

Aggregates are generated followed by primary suppression followed by secondary suppression by Gaussian elimination by GaussSuppression

## Usage

```
GaussSuppressionFromData(
  data,
  dimVar = NULL,
  freqVar = NULL,
  numVar = NULL,
  weightVar = NULL,
  charVar = NULL,
  hierarchies = NULL,
  formula = NULL,
  maxN = suppressWarnings(formals(c(primary)[[1]])$maxN),
  protectZeros = suppressWarnings(formals(c(primary)[[1]])$protectZeros),
  secondaryZeros = suppressWarnings(formals(candidates)$secondaryZeros),
  candidates = CandidatesDefault,
  primary = PrimaryDefault,
  forced = NULL,
  hidden = NULL,
  singleton = SingletonDefault,
  singletonMethod = ifelse(secondaryZeros, "anySumNOTprimary", "anySum"),
  printInc = TRUE,
  output = "publish",
  x = NULL,
  crossTable = NULL,
  preAggregate = is.null(freqVar),
  extraAggregate = preAggregate & !is.null(charVar),
  structuralEmpty = FALSE,
  extend0 = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| data | Input data as a data frame |
| dimVar | The main dimensional variables and additional aggregating variables. This parameter can be useful when hierarchies and formula are unspecified. |
| freqVar | A single variable holding counts (name or number). |

| | |
|---|---|
| numVar | Other numerical variables to be aggregated |
| weightVar | weightVar Weights (costs) to be used to order candidates for secondary suppression |
| charVar | Other variables possibly to be used within the supplied functions |
| hierarchies | List of hierarchies, which can be converted by AutoHierarchies. Thus, the variables can also be coded by "rowFactor" or "", which correspond to using the categories in the data. |
| formula | A model formula |
| maxN | Suppression parameter. Cells with frequency <= maxN are set as primary suppressed. Using the default primary function, maxN is by default set to 3. See details. |
| protectZeros | Suppression parameter. When TRUE, cells with zero frequency or value are set as primary suppressed. Using the default primary function, protectZeros is by default set to TRUE. See details. |
| secondaryZeros | Suppression parameter. When TRUE, cells with zero frequency or value are prioritized to be published so that they are not secondary suppressed. Using the default candidates function, secondaryZeros is by default set to FALSE. See details. |
| candidates | GaussSuppression input or a function generating it (see details) Default: CandidatesDefault |
| primary | GaussSuppression input or a function generating it (see details) Default: PrimaryDefault |
| forced | GaussSuppression input or a function generating it (see details) |
| hidden | GaussSuppression input or a function generating it (see details) |
| singleton | GaussSuppression input or a function generating it (see details) Default: SingletonDefault |
| singletonMethod | GaussSuppression input. The default value depends on parameter secondaryZeros which depends on candidates (see details). |
| printInc | GaussSuppression input |
| output | One of "publish" (default), "inner", "publish_inner", "publish_inner_x", "publish_x", "inner_x", "input2functions" (input to supplied functions), "inputGaussSuppression", "inputGaussSuppression_x", "outputGaussSuppression" "outputGaussSuppression_x", "primary" and "secondary". Here "inner" means input data (possibly pre-aggregated) and "x" means dummy matrix (as input parameter x). All input to and output from GaussSuppression, except ..., are returned when "outputGaussSuppression_x". Excluding x and only input are also possible. |
| x | x (modelMatrix) and crossTable can be supplied as input instead of generating it from ModelMatrix |
| crossTable | See above. |
| preAggregate | When TRUE, the data will be aggregated within the function to an appropriate level. This is defined by the dimensional variables according to dimVar, hierarchies or formula and in addition charVar. |

`extraAggregate` When `TRUE`, the data will be aggregated by the dimensional variables according to `dimVar`, `hierarchies` or `formula`. The aggregated data and the corresponding x-matrix will only be used as input to the singleton function and `GaussSuppression`. This extra aggregation is useful when parameter `charVar` is used. Supply `"publish_inner"`, `"publish_inner_x"`, `"publish_x"` or `"inner_x"` as `output` to obtain extra aggregated results. Supply `"inner"` or `"input2functions"` to obtain other results.

`structuralEmpty`
When `TRUE`, output cells with no contributing inner cells (only zeros in column of `x`) are forced to be not primary suppressed. Thus, these cells are considered as structural zeros. When `structuralEmpty` is `TRUE`, the following error message is avoided: `Suppressed cells with empty input will not be protected. Extend input data with zeros?`. When `removeEmpty` is `TRUE` (see `"..."` below), `structuralEmpty` is superfluous

`extend0` Data is automatically extended by `Extend0` when `TRUE`. Can also be set to `"all"` which means that input codes in hierarchies are considered in addition to those in data. Parameter `extend0` can also be specified as a list meaning parameter `varGroups` to `Extend0`.

`...` Further arguments to be passed to the supplied functions and to `ModelMatrix` (such as `inputInOutput` and `removeEmpty`).

## Details

The supplied functions for generating `GaussSuppression` input takes the following arguments: `crossTable`, `x`, `freq`, `num`, `weight`, `maxN`, `protectZeros`, `secondaryZeros`, `data`, `freqVar`, `numVar`, `weightVar`, `charVar`, `dimVar` and `....` where the two first are `ModelMatrix` outputs (`modelMatrix` renamed to `x`). The vector, `freq`, is aggregated counts (`t(x) %*% data[[freqVar]]`). Similarly, `num`, is a data frame of aggregated numerical variables. It is possible to supply several primary functions joined by `c`, e.g. (`c(FunPrim1,FunPrim2)`). All `NA`s returned from any of the functions force the corresponding cells not to be primary suppressed.

The effect of `maxN` , `protectZeros` and `secondaryZeros` depends on the supplied functions where these parameters are used. Their default values are inherited from the default values of the first `primary` function (several possible) or, in the case of `secondaryZeros`, the `candidates` function. When defaults cannot be inherited, they are set to `NULL`. In practice the function `formals` are still used to generate the defaults when `primary` and/or `candidates` are not functions. Then `NULL` is correctly returned, but `suppressWarnings` are needed.

Singleton handling can be turned off by `singleton = NULL` or `singletonMethod = "none"`. Both of these choices are identical in the sense that `singletonMethod` is set to `"none"` whenever `singleton` is `NULL` and vice versa.

## Value

Aggregated data with suppression information

## Author(s)

Øyvind Langsrud and Daniel Lupp

## Examples

```
z1 <- SSBtoolsData("z1")
GaussSuppressionFromData(z1, 1:2, 3)

z2 <- SSBtoolsData("z2")
GaussSuppressionFromData(z2, 1:4, 5, protectZeros = FALSE)


# Data as in GaussSuppression examples
df <- data.frame(values = c(1, 1, 1, 5, 5, 9, 9, 9, 9, 9, 0, 0, 0, 7, 7),
                 var1 = rep(1:3, each = 5), var2 = c("A", "B", "C", "D", "E"))

GaussSuppressionFromData(df, c("var1", "var2"), "values")
GaussSuppressionFromData(df, c("var1", "var2"), "values", formula = ~var1 + var2, maxN = 10)
GaussSuppressionFromData(df, c("var1", "var2"), "values", formula = ~var1 + var2, maxN = 10,
     protectZeros = TRUE, # Parameter needed by SingletonDefault and default not in primary
      primary = function(freq, crossTable, maxN, ...)
                  which(freq <= maxN & crossTable[[2]] != "A" & crossTable[, 2] != "C"))

# Combining several primary functions
# Note that NA & c(TRUE, FALSE) equals c(NA, FALSE)
GaussSuppressionFromData(df, c("var1", "var2"), "values", formula = ~var1 + var2, maxN = 10,
       primary = c(function(freq, maxN, protectZeros = TRUE, ...) freq >= 45,
                   function(freq, maxN, ...) freq <= maxN,
                   function(crossTable, ...) NA & crossTable[[2]] == "C",
                   function(crossTable, ...) NA & crossTable[[1]]== "Total"
                                             & crossTable[[2]]== "Total"))

# Similar to GaussSuppression examples
GaussSuppressionFromData(df, c("var1", "var2"), "values", formula = ~var1 * var2,
       candidates = NULL, singleton = NULL, protectZeros = FALSE, secondaryZeros = TRUE)
GaussSuppressionFromData(df, c("var1", "var2"), "values", formula = ~var1 * var2,
       singleton = NULL, protectZeros = FALSE, secondaryZeros = FALSE)
GaussSuppressionFromData(df, c("var1", "var2"), "values", formula = ~var1 * var2,
       protectZeros = FALSE, secondaryZeros = FALSE)


# Examples with zeros as singletons
z <- data.frame(row = rep(1:3, each = 3), col = 1:3, freq = c(0, 2, 5, 0, 0, 6:9))
GaussSuppressionFromData(z, 1:2, 3, singleton = NULL)
GaussSuppressionFromData(z, 1:2, 3, singletonMethod = "none") # as above
GaussSuppressionFromData(z, 1:2, 3)
GaussSuppressionFromData(z, 1:2, 3, protectZeros = FALSE, secondaryZeros = TRUE, singleton = NULL)
GaussSuppressionFromData(z, 1:2, 3, protectZeros = FALSE, secondaryZeros = TRUE)
```

---

GaussSuppressionTwoWay

*Two-way iteration variant of* GaussSuppressionFromData

---

## Description

Internally, data is organized in a two-way table.

Use parameter `colVar` to choose hierarchies for columns (others will be rows). Iterations start by column by column suppression. The algorithm utilizes `HierarchyCompute2`.

With two-way iterations, larger data can be handled, but there is a residual risk. The method is a special form of linked-table iteration. Separately, the rows and columns are protected by `GaussSuppression` and they have common suppressed cells.

## Usage

```
GaussSuppressionTwoWay(
  data,
  dimVar = NULL,
  freqVar = NULL,
  numVar = NULL,
  weightVar = NULL,
  charVar = NULL,
  hierarchies,
  formula = NULL,
  maxN = suppressWarnings(formals(c(primary)[[1]])$maxN),
  protectZeros = suppressWarnings(formals(c(primary)[[1]])$protectZeros),
  secondaryZeros = suppressWarnings(formals(candidates)$secondaryZeros),
  candidates = CandidatesDefault,
  primary = PrimaryDefault,
  forced = NULL,
  hidden = NULL,
  singleton = SingletonDefault,
  singletonMethod = ifelse(secondaryZeros, "anySumNOTprimary", "anySum"),
  printInc = TRUE,
  output = "publish",
  preAggregate = is.null(freqVar),
  colVar = names(hierarchies)[1],
  removeEmpty = TRUE,
  inputInOutput = TRUE,
  candidatesFromTotal = TRUE,
  structuralEmpty = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | Input data as a data frame |
| `dimVar` | The main dimensional variables and additional aggregating variables. This parameter can be useful when hierarchies and formula are unspecified. |
| `freqVar` | A single variable holding counts (name or number). |
| `numVar` | Other numerical variables to be aggregated |

| | |
|---|---|
| weightVar | weightVar Weights (costs) to be used to order candidates for secondary suppression |
| charVar | Other variables possibly to be used within the supplied functions |
| hierarchies | List of hierarchies, which can be converted by AutoHierarchies. Thus, the variables can also be coded by "rowFactor" or "", which correspond to using the categories in the data. |
| formula | A model formula |
| maxN | Suppression parameter. See GaussSuppressionFromData. |
| protectZeros | Suppression parameter. See GaussSuppressionFromData. |
| secondaryZeros | Suppression parameter. See GaussSuppressionFromData. |
| candidates | GaussSuppression input or a function generating it (see details) Default: CandidatesDefault |
| primary | GaussSuppression input or a function generating it (see details) Default: PrimaryDefault |
| forced | GaussSuppression input or a function generating it (see details) |
| hidden | GaussSuppression input or a function generating it (see details) |
| singleton | NULL or a function generating GaussSuppression input (logical vector not possible) Default: SingletonDefault |
| singletonMethod | GaussSuppression input |
| printInc | GaussSuppression input |
| output | One of "publish" (default), "inner". Here "inner" means input data (possibly pre-aggregated). |
| preAggregate | When TRUE, the data will be aggregated within the function to an appropriate level. This is defined by the dimensional variables according to dimVar, hierarchies or formula and in addition charVar. |
| colVar | Hierarchy variables for the column groups (others in row group). |
| removeEmpty | When TRUE (default) empty output corresponding to empty input is removed. When NULL, removal only within the algorithm (x matrices) so that such empty outputs are never secondary suppressed. |
| inputInOutput | Logical vector (possibly recycled) for each element of hierarchies. TRUE means that codes from input are included in output. Values corresponding to "rowFactor" or "" are ignored. |
| candidatesFromTotal | When TRUE (default), same candidates for all rows and for all columns, computed from row/column totals. |
| structuralEmpty | See GaussSuppressionFromData. |
| ... | Further arguments to be passed to the supplied functions. |

## Details

The supplied functions for generating GaussSuppression input behave as in GaussSuppressionFromData with some exceptions. When candidatesFromTotal is TRUE (default) the candidate function will be run locally once for rows and once for columns. Each time based on column or row totals. The

global x-matrix will only be generated if one of the functions supplied needs it. Non-NULL singleton can only be supplied as a function. This function will be run locally within the algorithm before each call to GaussSuppression.

Note that a difference from GaussSuppressionFromData is that parameter removeEmpty is set to TRUE by default.

Another difference is that duplicated combinations is not allowed. Normally duplicates are avoided by setting preAggregate to TRUE. When the charVar parameter is used, this can still be a problem. See the examples for a possible workaround.

## Value

Aggregated data with suppression information

## Examples

```
z3 <- SSBtoolsData("z3")

dimListsA <- SSBtools::FindDimLists(z3[, 1:6])
dimListsB <- SSBtools::FindDimLists(z3[, c(1, 4, 5)])

set.seed(123)
z <- z3[sample(nrow(z3),250),]

out1 <- GaussSuppressionTwoWay(z, freqVar = "ant", hierarchies = dimListsA,
                               colVar = c("hovedint"))
out2 <- GaussSuppressionTwoWay(z, freqVar = "ant", hierarchies = dimListsA,
                               colVar = c("hovedint", "mnd"))
out3 <- GaussSuppressionTwoWay(z, freqVar = "ant", hierarchies = dimListsB,
                               colVar = c("region"))
out4 <- GaussSuppressionTwoWay(z, freqVar = "ant", hierarchies = dimListsB,
                               colVar = c("hovedint", "region"))

# "mnd" not in  hierarchies -> duplicated combinations in input
# Error when  preAggregate is FALSE: Index method failed. Duplicated combinations?
out5 <- GaussSuppressionTwoWay(z, freqVar = "ant", hierarchies = dimListsA[1:3],
                        protectZeros = FALSE, colVar = c("hovedint"), preAggregate = TRUE)


# charVar needed -> Still problem when preAggregate is TRUE
# Possible workaround by extra hierarchy
out6 <- GaussSuppressionTwoWay(z, freqVar = "ant", charVar = "mnd2",
                         hierarchies = c(dimListsA[1:3], mnd2 = "Total"), # include charVar
                              inputInOutput = c(TRUE, TRUE, FALSE),  # FALSE -> only Total
                               protectZeros = FALSE, colVar = c("hovedint"),
                               preAggregate = TRUE,
                               hidden = function(x, data, charVar, ...)
                        as.vector((t(x) %*% as.numeric(data[[charVar]] == "M06M12")) == 0))
```

LazyLinkedTables *Linked tables by full* GaussSuppressionFromData *iterations*

## Description

AdditionalSuppression is called several times as in ChainedSuppression

## Usage

```
LazyLinkedTables(..., withinArg = NULL, maxIterLinked = 1000)
```

## Arguments

| | |
|---|---|
| ... | Arguments to GaussSuppressionFromData that are kept constant. |
| withinArg | A list of named lists. Arguments to GaussSuppressionFromData that are not kept constant. |
| maxIterLinked | Maximum number of GaussSuppressionFromData calls for each table. |

## Details

This function is created as a spin-off from AdditionalSuppression and ChainedSuppression. The calculations run GaussSuppressionFromData from the input each time. There is no doubt that this can be done more efficiently.

A consequence of this lazy implementation is that, in output, primary and suppressed are identical.

Note that there is a residual risk when suppression linked tables by iterations.

## Value

List of data frames

## Note

In this function, the parameters makeForced and forceNotPrimary to AdditionalSuppression are forced to be FALSE.

## Examples

```
z1 <- SSBtoolsData("z1")
z2 <- SSBtoolsData("z2")
z2b <- z2[3:5]  # As in ChainedSuppression example
names(z2b)[1] <- "region"

# The two region hierarchies as two linked tables
a <- LazyLinkedTables(z2, freqVar = 5, withinArg = list(
       list(dimVar = c(1, 2, 4)),
```

```
       list(dimVar = c(1, 3, 4))))

# As 'f' and 'e' in ChainedSuppression example.
# 'A' 'annet'/'arbeid' suppressed in b[[1]], since suppressed in b[[3]].
b <- LazyLinkedTables(withinArg = list(
       list(data = z1, dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2b, dimVar = 1:2, freqVar = 3, maxN = 5),
       list(data = z2, dimVar = 1:4, freqVar = 5, maxN = 1)))
```

---

`MaxContribution` *Find major contributors to aggregates*

---

## Description

Assuming aggregates are calculated via a dummy matrix by `z = t(x) %*% y`, the `n` largest contributors are found (value or index) for each aggregate.

## Usage

```
MaxContribution(x, y, n = 1, decreasing = TRUE, index = FALSE, groups = NULL)
```

## Arguments

| | |
|---|---|
| `x` | A (sparse) dummy matrix |
| `y` | Vector of input values (contributors) |
| `n` | Number of contributors to be found |
| `decreasing` | Ordering parameter. Smallest contributors found when `FALSE`. |
| `index` | Indices to `y` returned when TRUE |
| `groups` | When non-NULL, major contributions after aggregation within groups. Cannot be combined with `index = TRUE`. The missing group category is excluded. |

## Value

Matrix with lagest contributors in first column, second largest in second column and so on.

## Author(s)

Øyvind Langsrud

## See Also

`ModelMatrix`

## Examples

```
library(SSBtools)

z <- SSBtoolsData("sprt_emp_withEU")
z$age[z$age == "Y15-29"] <- "young"
z$age[z$age == "Y30-64"] <- "old"

a <- ModelMatrix(z, formula = ~age + geo, crossTable = TRUE)

cbind(as.data.frame(a$crossTable), MaxContribution(a$modelMatrix, z$ths_per, 1))
cbind(a$crossTable, MaxContribution(a$modelMatrix, z$ths_per, 10))
cbind(a$crossTable, MaxContribution(a$modelMatrix, z$ths_per, 10, index = TRUE))

b <- ModelMatrix(z[, -4], crossTable = TRUE, inputInOutput = c(TRUE, FALSE, TRUE))

k <- cbind(b$crossTable, MaxContribution(b$modelMatrix, z$ths_per, 10))

gr18 <- paste0("g", 1:18)                          # Each row is a group
k18 <- cbind(b$crossTable, MaxContribution(b$modelMatrix, z$ths_per, 10, groups = gr18))
identical(k, k18) # TRUE

gr9 <- paste0("g", as.integer(10 * z$ths_per)%%10) # 9 groups from decimal
k9 <- cbind(b$crossTable, MaxContribution(b$modelMatrix, z$ths_per, 10, groups = gr9))

k18[c(4, 13, 17, 33), ]
k9[c(4, 13, 17, 33), ]
```

---

# Ncontributors *Find the number of unique groups contributing to aggregates*

---

## Description

Assuming aggregates are calculated via a dummy matrix by `z = t(x) %*% y`, the the number of unique contributing groups, according to a grouping variable, are found for each aggregate. The missing group category is not counted.

## Usage

```
Ncontributors(x, groups)
```

## Arguments

| | |
|---|---|
| `x` | A (sparse) dummy matrix |
| `groups` | Vector of group categories |

## Value

Vector of numbers of unique groups

### Author(s)

Øyvind Langsrud

### See Also

ModelMatrix

### Examples

```
library(SSBtools)

z <- SSBtoolsData("sprt_emp_withEU")
z$age[z$age == "Y15-29"] <- "young"
z$age[z$age == "Y30-64"] <- "old"
z$groups <- c("A", "A", "B", "A", "B", "C")

a <- ModelMatrix(z, formula = ~age*eu + geo + year, crossTable = TRUE)

cbind(as.data.frame(a$crossTable), nGroups = Ncontributors(a$modelMatrix, z$groups))
cbind(as.data.frame(a$crossTable), nYears = Ncontributors(a$modelMatrix, z$year))
cbind(as.data.frame(a$crossTable), nUnique_ths_per = Ncontributors(a$modelMatrix, z$ths_per))
```

---

## NcontributorsHolding  Ncontributors *with holding-indicator*

---

### Description

The aggregates (columns of x) are grouped by a holding indicator. Within each holding group, the number of unique groups (output) is set to be equal.

### Usage

```
NcontributorsHolding(x, groups, holdingInd = NULL)
```

### Arguments

| | |
|---|---|
| x | A (sparse) dummy matrix |
| groups | Vector of group categories |
| holdingInd | Vector of holding group categories |

### Details

A representative within the holding group is used to calculate output by Ncontributors. The one with maximal column sum of x is chosen as the representative. Normally this will be an aggregate representing the holding group total. When holdingInd is NULL (default), the function is equivalent to Ncontributors.

## Value

Vector of numbers of unique groups

## Author(s)

Øyvind Langsrud

---

`PrimaryFromSuppressedData` `primary` *and* `forced` *from suppressed data*

---

## Description

Function for `GaussSuppressionFromData`

## Usage

```
PrimaryFromSuppressedData(
  x,
  crossTable,
  suppressedData,
  forcedData = FALSE,
  totCode = FindTotCode2(x, crossTable),
  ...
)

ForcedFromSuppressedData(..., forcedData = TRUE)

NotPrimaryFromSuppressedData(..., forcedData = TRUE)
```

## Arguments

| | |
|---|---|
| `x` | A (sparse) dummy matrix |
| `crossTable` | crossTable generated by parent function |
| `suppressedData` | A data frame or a list of data frames as output from `GaussSuppressionFromData`. If the variable `suppressed` is not included, all rows are considered suppressed. |
| `forcedData` | When `TRUE`, the suppressed coding is swapped. |
| `totCode` | A named list of totals codes |
| `...` | Unused parameters |

## Details

ForcedFromSuppressedData uses forcedData = TRUE and hence a vector to be use as forced is generated. NotPrimaryFromSuppressedData is similar, but TRUE elements are replaced by NA's. Hence the result can be used as an extra primary vector to ensure that code combinations not suppressed according to suppressedData are forced not to be primary suppressed.

The variables used in suppressedData in addition to "suppressed" are those with matching names in crossTable. Others are ignored. For variables in crossTable not in suppressedData, only totals are considered. Others rows are ignored when mathing with suppressedData.

When suppressedData is a list, the final result is the union of individual results of each data frame.

## Value

Logical vector to be used as GaussSuppression input

## Examples

```
z2 <- SSBtoolsData("z2")

# Data to be used as suppressedData
a <- GaussSuppressionFromData(z2, c(1, 3, 4), 5, protectZeros = FALSE)

# For alternative ways to suppress the same table
b1 <- GaussSuppressionFromData(z2, 1:4, 5)
b2 <- GaussSuppressionFromData(z2, 1:4, 5, primary = c(PrimaryDefault, PrimaryFromSuppressedData),
                               suppressedData = a)
b3 <- GaussSuppressionFromData(z2, 1:4, 5, primary = c(PrimaryDefault, PrimaryFromSuppressedData),
                               suppressedData = a, forced = ForcedFromSuppressedData)
b4 <- GaussSuppressionFromData(z2, 1:4, 5,
         primary = c(PrimaryDefault, PrimaryFromSuppressedData, NotPrimaryFromSuppressedData),
                               suppressedData = a, forced = ForcedFromSuppressedData)

# Reducing data to rows mathing a
b1r <- b1[SSBtools::Match(a[1:2], b1[1:2]), ]
b2r <- b2[SSBtools::Match(a[1:2], b2[1:2]), ]
b3r <- b3[SSBtools::Match(a[1:2], b3[1:2]), ]
b4r <- b4[SSBtools::Match(a[1:2], b4[1:2]), ]


# Look at rows where new suppression is different from that in a

# Both TRUE and FALSE changed
cbind(a, b1r)[b1r$suppressed != a$suppressed, c(1:5, 9:10)]

# Only  FALSE changed to TRUE (suppression is preserved)
cbind(a, b2r)[b2r$suppressed != a$suppressed, c(1:5, 9:10)]

# Only change is due to new primary suppression rule (protectZeros = TRUE)
cbind(a, b3r)[b3r$suppressed != a$suppressed, c(1:5, 9:10)]

# No changes
```

```
cbind(a, b4r)[b4r$suppressed != a$suppressed, c(1:5, 9:10)]
```

---

`SuppressDirectDisclosure` *Suppression of directly-disclosive cells*

---

## Description

Function for suppressing directly-disclosive cells in frequency tables. The method detects and primary suppresses directly-disclosive cells with the FindDisclosiveCells function, and applies a secondary suppression using Gauss suppression (see GaussSuppressionFromData).

## Usage

```
SuppressDirectDisclosure(
  data,
  dimVar,
  freqVar,
  coalition = 1,
  secondaryZeros = coalition,
  candidates = DirectDisclosureCandidates,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | the input data |
| `dimVar` | main dimensional variables for the output table |
| `freqVar` | variable containing frequency counts |
| `coalition` | numeric variable, parameter for primary suppression. Default value is 1. |
| `secondaryZeros` | logical or numeric value for secondary suppression. If logical, it is converted to resp numeric value (0 or 1). If numeric, it describes the largest number that is prioritized over zeroes in secondary suppression. Default value is equal to coalition. |
| `candidates` | function parameter for gauss suppression. |
| `...` | optional parameters that can be passed to the primary suppression method. See FindDisclosiveCells for details. |

## Details

Currently, the method has no support for hierarchical data.

## Value

data.frame containing the result of the suppression

## Author(s)

Daniel Lupp

## Examples

```
tex <- data.frame(v1 = rep(c('a', 'b', 'c'), times = 4),
                  v2 = c('i','i', 'i','h','h','h','i','i','i','h','h','h'),
                  v3 = c('y', 'y', 'y', 'y', 'y', 'y','z','z', 'z', 'z', 'z', 'z'),
                  freq = c(0,0,5,0,2,3,1,0,3,1,1,2))
SuppressDirectDisclosure(tex, c("v1", "v2", "v3"), "freq")
SuppressDirectDisclosure(tex, c("v1", "v2", "v3"), "freq", coalition = 2, unknown.threshold = 10)
```

---

`SuppressionFromDecimals` *Cell suppression from synthetic decimal numbers*

---

## Description

Decimal numbers, as calculated by `GaussSuppressDec`, are used to decide suppression (whole numbers or not). Technically, the calculations are done via `GaussSuppressionFromData`, but without running `GaussSuppression`. All suppressed cells are primary suppressed.

## Usage

```
SuppressionFromDecimals(
  data,
  decVar,
  freqVar = NULL,
  numVar = NULL,
  preAggregate = FALSE,
  digits = 9,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | Input data as a data frame |
| `decVar` | One ore several (`nRep>1`) decimal number variables. |
| `freqVar` | A single variable holding counts (not needed) |
| `numVar` | Other numerical variables to be aggregated |
| `preAggregate` | Parameter to `GaussSuppressionFromData` |
| `digits` | Parameter to `RoundWhole`. Values close to whole numbers will be rounded. |
| `...` | Other parameters to `GaussSuppressionFromData` |

## Details

Several decimal number variables reduce the probability of obtaining whole numbers by chance.

## Value

Aggregated data with suppression information

## Author(s)

Øyvind Langsrud

## Examples

```
z2 <- SSBtoolsData("z2")

# Find suppression and decimal numbers with "fylke" in model
a <- GaussSuppressDec(z2, dimVar = c("region", "fylke", "hovedint"),
                      freqVar = "ant", protectZeros = FALSE, maxN = 2,
                      output = "inner")

# Add decimal numbers to data
z2$freqDec <- a$freqDec

# Find suppression with "kostragr" in model
b <- SuppressionFromDecimals(z2, dimVar = c("region", "kostragr", "hovedint"),
                             freqVar = "ant", decVar = "freqDec")
```

# Index


AdditionalSuppression, 2, *4*, *16*
AutoHierarchies, *10*, *14*

CandidatesDefault, *10*, *14*
ChainedSuppression, 4, *16*
ChainedSuppressionHi
        *(*ChainedSuppression*)*, 4
ChainedSuppressionHi1
        *(*ChainedSuppression*)*, 4

DominanceRule, 6

FindDisclosiveCells, *22*
ForcedFromSuppressedData
        *(*PrimaryFromSuppressedData*)*, 20

GaussSuppressDec, 7, *23*
GaussSuppression, *9–11*, *13–15*, *21*, *23*
GaussSuppressionFromData, *2*, *7*, *8*, 9, *12*, *14*, *16*, *20*, *22*, *23*
GaussSuppressionTwoWay, 12

HierarchyCompute2, *13*

LazyLinkedTables, 16

MaxContribution, 17
Mipf, *8*
ModelMatrix, *10*, *11*, *17*, *19*

Ncontributors, 18, *19*
NcontributorsHolding, 19
NotPrimaryFromSuppressedData
        *(*PrimaryFromSuppressedData*)*, 20

PrimaryDefault, *10*, *14*
PrimaryFromSuppressedData, *3*, 20

RoundWhole, *8*, *23*

SingletonDefault, *10*, *14*
SuppressDec, *7*
SuppressDirectDisclosure, 22
SuppressionFromDecimals, 23