Package ‘revengc’

January 8, 2019

Type Package

Title Reverse Engineering Summarized Data

Version 1.0.4

Author Samantha Duchscherer [aut, cre],
UT-Battelle, LLC [cph]

Maintainer Samantha Duchscherer <sam.duchscherer@gmail.com>

Description

Decoupled (e.g. separate averages) and censored (e.g. > 100 species) variables are continually reported by many well-established organizations (e.g. World Health Organization (WHO), Centers for Disease Control and Prevention (CDC), World Bank, and various national censuses). The challenge therefore is to infer what the original data could have been given summarized information. We present an R package that reverse engineers decoupled and/or censored count data with two main functions. The cnbinom.pars function estimates the average and dispersion parameter of a censored univariate frequency table. The rec function reverse engineers summarized data into an uncensored bivariate table of probabilities.

URL https://github.com/GIST-ORNL/revengc

Depends R (>= 3.5.0)

License MIT + file LICENSE

LazyData TRUE

Imports stringr, mipfp, dplyr, truncdist

Suggests R.rsp

VignetteBuilder R.rsp

RoxygenNote 6.0.1.9000

NeedsCompilation no

Repository CRAN

Date/Publication 2019-01-08 09:20:03 UTC

R topics documented:

cnbinom.pars 2
column.marginal 4
contingency_table_example 5
rec 6
reweight.contingencytable 11
reweight.univariatetable 13
row.marginal 15
seedmatrix 16
univariate_table_example 18

Index **20**

`cnbinom.pars` *Estimation of negative binomial parameters*

Description

A univariate censored frequency table is fit to a negative binomial distribution using a likelihood function customized to handle left, right, and interval censored data. The output is a list containing the average and dispersion parameter that maximizes the custom function.

Usage

```
cnbinom.pars(censoredtable)
```

Arguments

`censoredtable`	A frequency table. A data.frame and matrix are acceptable classes. See Details section for formatting.

Details

The censored table for the cnbinom.pars function has restrictions. The univariate frequency table, which can be a data.frame or matrix class, must have two columns and n number of rows. The categories must be in the first column with frequencies or probabilities in the second column. Row names should never be placed in this table (the default row names should always be 1:n). Column names can be any character string. The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

Below are three correctly formatted tables.

Category	Frequency
<=6	11800
7-12	57100
13-19	14800
20+	3900

Category	Frequency
LE6	11800
7I12	57100
13I19	14800
GE20	3900

Category	Frequency
<7	11800
7I12	57100
13-19	14800
>=20	3900

Value

The cnbinom.pars function outputs a list consisting of the estimated average (mu) and dispersion (r) parameter.

Examples

```
# create frequency table that follows a Poisson distribution
set.seed(123)
testdata1<-data.frame(table(rpois(100000,lambda = 10)))

# negative binomial converges in distribution to the Poisson
## when dispersion->infinity
cnbinom.pars(testdata1)

# censor table of testdata1 (lambda = 10)
testdata2<-cbind(as.character(c("<=5", "6-10", "11-15", "16-20", ">20")),
  c(6718,51329,37041,4732, 180))
cnbinom.pars(testdata2)

# create frequency table that follows a negative binomial distribution
# different ways to parameterize the negative binomial distribution
mu = 10
r = 2
p = r/(r+mu)
set.seed(123)
testdata3<-data.frame(table(rnbinom(100000, size = r, mu = mu)))
```

```
cnbinom.pars(testdata3)
set.seed(123)
testdata4<-data.frame(table(rnbinom(100000, size = r, p = p)))
cnbinom.pars(testdata4)

# censor table with mu = 10 and r = 2
testdata5<-cbind(as.character(c("<5", "5-15", "16I26", "27-37", "38I48", ">48")),
  c(26130,53979,15899,3267, 593, 132))
cnbinom.pars(testdata5)
```

column.marginal *Column marginal table from contingency table*

Description

This function inputs a contingency table and produces a univariate frequency table from the column marginals.

Usage

```
column.marginal(contingencytable)
```

Arguments

`contingencytable`
A censored contingency table. Accepted classes are a data.frame or matrix. See Details section below formatting.

Details

Table Format:
The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

The column names should be the Y category values. The first column should be the X category values and the row names can be arbitrary. The inside of the table are X * Y cross tabulation, which are either positive frequency values or probabilities. The row and column marginal totals corresponding to their X and Y category values need to be placed in this table. The top left, top right, and bottom left corners of the table should be NA or blank. The bottom right corner can be a total cross tabulation sum value, NA, or blank. The table below is a formatted example.

NA	<20	20-30	>30	NA

<5	18	19	8	45
5-9	13	8	12	33
>=10	7	5	10	21
NA	38	32	31	NA

Value

A univariate frequency table (data.frame) created from the column marginals.

Examples

```
# contingencytable.csv is a preloaded contingency table example
column.marginal(contingencytable.csv)
```

`contingency_table_example` *Contingency table example*

Description

In 2010, the Population Census Data - Statistics Indonesia provided over 60 censored contingency tables containing Floor Area of Dwelling Unit (square meter) by Household Member Size. The tables are separated by province, urban, and rural. Here we show the household size by area contingency table for Indonesia 's rural Aceh Province.

contingencytable.csv is a preloaded example that is correctly formatted for the revengc package. Below shows how a user should read in a csv file to have the same format:

contingencytable.csv<-read.csv("filename.csv", row.names = NULL, header= TRUE, check.names=FALSE)

Usage

```
contingencytable.csv
```

References

Population Census Data - Statistics Indonesia. (2010). *Household by Floor Area of Dwelling Unit and Households Member Size*. Retrieved from: http://sp2010.bps.go.id/index.php/site/tabel?wid=1100000000&tid=334&fi1=

rec *Reverse engineering censored and decoupled data*

Description

rec is a statistical approach that estimates what "true" uncensored bivariate table could have been given summarized information. Note, there are assumptions used in this function. First, rec relies on a Poisson distribution when a user only provides an average, which is assuming the variance and average of that variable are equal. A more descriptive input variable(s), such as a decoupled univariate table(s) or a censored frequency table, can account for dispersion found in data. However, independence between decoupled variables still has to be assumed when there is no external information about the joint distribution. Because of these assumptions, rec also provides two options for sensitivity analysis: seed matrix the method used in updating the seed matrix are both arbitrary inputs. For more information it is recommended for a user to read the Details section below and more information can be found in the vignettes.

Usage

```
rec(X, Y, Xlowerbound, Xupperbound, Ylowerbound, Yupperbound,
    seed.matrix, seed.estimation.method)
```

Arguments

`X` Argument can be an average, a univariate frequency table, or a censored contingency table. The average value should be a numeric class while a data.frame or matrix are acceptable table classes. Y defaults to NULL if X argument is a censored contingency table. See Details section below formatting.

`Y` Same description as X but this argument is for the Y variable. X defaults to NULL if Y argument is a censored contingency table.

`Xlowerbound` A numeric class value to represent the left bound for X (row in contingency table). The value must strictly be a non-negative integer and cannot be greater than the lowest category/average value provided for X (e.g. the lower bound cannot be 6 if a table has '< 5' as a X or row category).

`Xupperbound` A numeric class value to represent the right bound for X (row in contingency table). The value cannot be less than the highest category/average value provided for X (e.g. the upper bound cannot be 90 if a table has '> 100' as a X or row category).

`Ylowerbound` Same description as Xlowerbound but this argument is for Y (column in contingency table).

`Yupperbound` Same description as Xupperbound but this argument is for Y (column in contingency table).

`seed.matrix` An initial probability matrix to be updated. If decoupled variables is provided the default is a Xlowerbound:Xupperbound by Ylowerbound:Yupperbound matrix with interior cells of 1, which are then converted to probabilities. If a censored contingency table is provided the default is the seedmatrix()\$Probabilities output.

`seed.estimation.method`

A character string indicating which method is used for updating the seed.matrix. The choices are: "ipfp", "ml", "chi2", or "lsq". Default is "ipfp".

Details

Overview:
The rec function handles four cases.

- **Case I.** When provided an average for both X and Y, the averages represent lambda values. These lambdas create truncated Poisson X and Y probability densities for uncensored vectors ranging from Xlowerbound:Xupperbound and Ylowerbound:Yupperbound, respectively. The Xlowerbound:Xupperbound vector with its corresponding density values represents the new row marginal. The Ylowerbound:Yupperbound vector with its corresponding density values represents the new column marginal. This is a decoupled case, and thus the seed (initial cross tabulation matrix to be updated) defaults to (a matrix of ones)/(length(Xlowerbound:Xupperbound) * length(Ylowerbound:Yupperbound)). The mipfp R package then estimates cross tabulations with a selected seed estimation method, new uncensored marginals, and seed matrix. The final result is an uncensored contingency table of probabilities.
- **Case II.** When provided an univariate frequency table for both X and Y, the negative binomial average and dispersion parameters are estimated with a customized maximum likelihood function. These parameters then create truncated negative binomial X and Y probability densities for uncensored vectors, Xlowerbound:Xupperbound and Ylowerbound:Yupperbound, respectively. The methods listed in Case I are then implemented.
- **Case III.** When provided a combination of an average and frequency table (X and Y could be either), the same methods stated in Case I and II are implemented.
- **Case IV.** When provided a censored X*Y contingency table, the row marginals create a univariate X frequency table while the column marginals create a univariate Y frequency table. Both tables estimate there corresponding negative binomial average and dispersion parameters. These parameters then create truncated negative binomial X and Y probability densities for uncensored vectors Xlowerbound:Xupperbound and Ylowerbound:Yupperbound, respectively. The Xlowerbound:Xupperbound vector with its corresponding density values represents the new row marginal. The Ylowerbound:Yupperbound vector with its corresponding density values represents the new column marginal. This is not a decoupled case, and thus the default seed repeats the probability cells, which corresponding to the censored contingency table, for the newly created and compatible uncensored cross tabulations and then makes the matrix equal 1 (i.e. for each cell value j in the seed: seed[j]/sum(seed)). The mipfp R package then estimates cross tabulations with a selected seed estimation method, new uncensored marginals, and seed matrix. The final result is an uncensored contingency table of probabilities.

Table Format:
The table(s) for Case II and III has restrictions. The univariate frequency table, which can be a data.frame or matrix class, must have two columns and n number of rows. The categories must be in the first column with frequencies or probabilities in the second column. Row names should never be placed in this table (the default row names should always be 1:n). Column names can be any character string. The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

Below are three correctly formatted tables.

Category	Frequency
<=6	11800
7-12	57100
13-19	14800
20+	3900

Category	Frequency
LE6	11800
7I12	57100
13I19	14800
GE20	3900

Category	Frequency
<7	11800
7I12	57100
13-19	14800
>=20	3900

The table for Case IV also has restrictions. The censored symbols should follow the requirements listed above. The table's class can be a data.frame or a matrix. The column names should be the Y category values. The first column should be the X category values and the row names can be arbitrary. The inside of the table are X * Y frequency values, which are either nonnegative frequencies or probabilities if seed_estimation_method is "ipfp" or strictly positive when method is "ml", "lsq" or "chi2". The row and column marginal totals corresponding to their X and Y category values need to be placed in this table. The top left, top right, and bottom left corners of the table should be NA or blank. The bottom right corner can be a total cross tabulation sum value, NA, or blank. The table below is a formatted example.

NA	<20	20-30	>30	NA
<5	18	19	8	45
5-9	13	8	12	33
>=10	7	5	10	21
NA	38	32	31	NA

Bounds:
Ideally, the four bounds should be chosen based off prior knowledge and expert elicitation, but they can also be selected intuitively with a brute force method. If rec outputs a final contingency table with higher probabilities near the edge(s) of the table, then it would make sense to increase the range of the bound(s). For both the X and Y variables, this would just involve making the lower bound less, making the upper bound more, or doing a combination of the two. The opposite holds true as well. If the final contingency table has very low probabilities near the edge(s) of the table, then a user should decrease the range of the particular bound(s).

Seed Estimation Methods:
This function implements the *mipfp* R package, which offers four methods to estimate cross tabulations when provided fixed marginals.

Method	Abbreviation
Iterative proportional fitting procedure	ipfp
Maximum likelihood method	ml
Minimum chi-squared	chi2
Weighted least squares	lsq

For a summary and understanding of all methods please refer to the vignettes and/or the papers by Little et al. (1991) and Suesse et al. (2017).

Value

The output is a list containing an uncensored contingency table of probabilities (rows range from Xlowerbound:Xupperbound and the columns range from Ylowerbound:Yupperbound) as well as the row X and column Y parameters used in making the margins for the mipfp R package.

References

Frederick Novomestky and Saralees Nadarajah (2016). truncdist: Truncated Random Variables. R package version 1.0-2. https://CRAN.R-project.org/package=truncdist

Johan Barthelemy and Thomas Suesse (2018). mipfp: Multidimensional Iterative Proportional Fitting and Alternative Models. R package version 3.2. https://CRAN.R-project.org/package=mipfp

Little, R. J., Wu, M. M. (1991) Models for contingency tables with known margins when target and sampled populations differ. *Journal of the American Statistical Association*, 86(413): 87-95. doi: https://doi.org/10.2307/2289718

Suesse, T., Namazi-Rad, M., Mokhtarian, P., & Barthelemy, J. (2017). Estimating Cross-Classified Population Counts of Multidimensional Tables: An Application to Regional Australia to Obtain Pseudo-Census Counts, *Journal of Official Statistics*, 33(4), 1021-1050. doi: https://doi.org/10.1515/jos-2017-0048

Examples

```
# provide two averages
# seed.matrix defaults to a matrix of ones
# seed.estimation.method defaults to ipfp
twoaverages.results<-rec(
   X= 4.4,
   Y = 571.3,
   Xlowerbound = 1,
   Xupperbound = 20,
   Ylowerbound = 520,
   Yupperbound = 620)


# provide one average and one table
# create a censored univariate table
# seed.matrix defaults to a matrix of ones
# seed.estimation.method defaults to ipfp
Y.table = cbind(as.character(c("<7", "7-12", "13-19", ">19")),
  c(11800,57100,14800,3900))
combo.results<-rec(X= 2.3,
   Y = Y.table,
   Xlowerbound = 1,
   Xupperbound = 15,
   Ylowerbound = 1,
   Yupperbound = 30)


# provide a censored contingency table
contingencytable<-matrix(c(6185,9797,16809,11126,6156,3637,908,147,69,4,
                       5408,12748,26506,21486,14018,9165,2658,567,196,78,
                       7403,20444,44370,36285,23576,15750,4715,994,364,136,
                       4793,17376,44065,40751,28900,20404,6557,1296,555,228,
                       2354,11143,32837,33910,26203,19301,6835,1438,618,245,
                       1060,6038,19256,21298,17774,13864,4656,1039,430,178,
                       273,2521,9110,11188,9626,7433,2608,578,196,112,
                       119,1130,4183,5566,5053,3938,1367,318,119,66,
                       33,388,1707,2367,2328,1972,719,171,68,37,
                       38,178,1047,1672,1740,1666,757,193,158,164),
                         nrow=10,ncol=10, byrow=TRUE)
rowmarginal<-apply(contingencytable,1,sum)
contingencytable<-cbind(contingencytable, rowmarginal)
colmarginal<-apply(contingencytable,2,sum)
contingencytable<-rbind(contingencytable, colmarginal)
row.names(contingencytable)[row.names(contingencytable)=="colmarginal"]<-""
contingencytable<-data.frame(c("1","2","3","4","5","6", "7", "8","9","10+", NA),
  contingencytable)
colnames(contingencytable)<-c(NA,"<20","20-29","30-39","40-49","50-69","70-99",
                              "100-149","150-199","200-299","300+", NA)

# the contingencytable input could be put in X or Y (opposing argument = NULL)
# X = rows and Y = columns
# seed.matrix default = repeating the cross tabulations in the censored contingency
## table for the newly created and compatible uncensored cross tabulations
# seed.estimation.method defaults to ipfp
contingencytable.results<-rec(
   X= contingencytable,
   Xlowerbound = 1,
   Xupperbound = 15,
   Ylowerbound = 10,
   Yupperbound = 310)
```

reweight.contingencytable

Reweighting a contingency table

Description

This function is used in the main function: rec.

Usage

```
reweight.contingencytable(observed.table, estimated.table)
```

Arguments

`observed.table`	A censored contingency table. See Details section below formatting this data.frame.
`estimated.table`	A data.frame with uncensored row names and column names.

Details

Format for observed.table:
The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

The column names should be the Y category values. The first column should be the X category values and the row names can be arbitrary. The inside of the table are X * Y cross tabulation, which are either positive frequency values or probabilities. The row and column marginal totals corresponding to their X and Y category values need to be placed in this table. The top left, top right, and bottom left corners of the table should be NA or blank. The bottom right corner can be a total cross tabulation sum value, NA, or blank. The table below is a formatted example.

NA	<20	20-30	>30	NA
<5	18	19	8	45
5-9	13	8	12	33
>=10	7	5	10	21
NA	38	32	31	NA

Value

Interior probability cells of a censored contingency table (observed.table) is reweighted to match interior probability cells of an uncensored contingency table (estimated.table). If observed.table consist of frequencies, the reweight.contingencytable() function changes frequencies to probabilities.

Examples

```
## going through the coding step of rec ##

# first create contingency table
contingencytable<-matrix(1:9,
                         nrow = 3, ncol = 3)
rowmarginal<-apply(contingencytable,1,sum)
contingencytable<-cbind(contingencytable, rowmarginal)
colmarginal<-apply(contingencytable,2,sum)
contingencytable<-rbind(contingencytable, colmarginal)
row.names(contingencytable)[row.names(contingencytable)==”colmarginal”]<-””
contingencytable<-data.frame(c(”<5”, ”5I9”, ”>9”, NA), contingencytable)
colnames(contingencytable)<-c(NA,”<=19”,”20-30”,”>=31”, NA)

# provided upper and lower bound values for table
# X=row and Y=column
Xlowerbound=1
Xupperbound=15
Ylowerbound=15
Yupperbound=35

# table of row marginals provides average x and phi x
row.marginal.table<-row.marginal(contingencytable)
x<-cnbinom.pars(row.marginal.table)
# table of column marginals provides average y and phi y
column.marginal.table<-column.marginal(contingencytable)
y<-cnbinom.pars(column.marginal.table)

# create row and column ranges
rowrange<-Xlowerbound:Xupperbound
colrange<-Ylowerbound:Yupperbound

library(truncdist)
# new uncensored row marginal table = truncated negative binomial distribution
# rowrange = X is distributed given a < X <= b
uncensored.row.margin<-dtrunc(rowrange, mu=x$Average, size = x$Dispersion,
                              a = Xlowerbound-1, b = Xupperbound, spec = ”nbinom”)
# new uncensored column margin table = = truncated negative binomial distribution
# colrange = Y is distributed given a < Y <= b
uncensored.column.margin<-dtrunc(colrange, mu=y$Average, size = y$Dispersion,
                                 a = Ylowerbound-1, b = Yupperbound, spec = ”nbinom”)

# sum of truncated distributions equal 1
sum(uncensored.row.margin)
sum(uncensored.column.margin)
```

```
# look at the seed for this example (probabilities)
seed.output<-seedmatrix(contingencytable, Xlowerbound,
                        Xupperbound, Ylowerbound, Yupperbound)$Probabilities

# run mipfp
# store the new margins in a list
tgt.data<-list(uncensored.row.margin, uncensored.column.margin)
# list of dimensions of each marginal constrain
tgt.list<-list(1,2)
# calling the estimated function
## seed has to be in array format for mipfp package
## ipfp is the selected seed.estimation.method
## $p.hat gives probabilies = x.hat/sum(x.hat)
library(mipfp)
final1<-Estimate(array(seed.output,dim=c(length(Xlowerbound:Xupperbound),
              length(Ylowerbound:Yupperbound))), tgt.list, tgt.data, method="ipfp")$x.hat

# filling in names of updated seed
final1<-data.frame(final1)
row.names(final1)<-Xlowerbound:Xupperbound
names(final1)<-Ylowerbound:Yupperbound

# reweight estimates to known censored interior cell probabilities
final1<-reweight.contingencytable(observed.table = contingencytable, estimated.table = final1)

# see that they sum to one
sum(final1)

# rec function outputs the same table
# default of rec seed.estimation.method is ipfp
# default of rec seed.matrix is the output of the seedmatrix() function
final2<-rec(X= contingencytable,
            Xlowerbound = 1,
            Xupperbound = 15,
            Ylowerbound = 15,
            Yupperbound = 35)

# check that both data.frame results have same values
all(final1 == final2$Probability.Estimates)
```

reweight.univariatetable

Reweighting a univariate table

Description

This function is used in the main function: rec.

Usage

```
reweight.univariatetable(observed.table, estimated.table)
```

Arguments

`observed.table`	A data.frame or matrix with two columns. See Details section for formatting.
`estimated.table`	A numeric vector with uncensored row names.

Details

Format for observed.table:
This univariate frequency table, which can be a data.frame or matrix class, must have two columns and n number of rows. The categories must be in the first column with frequencies or probabilities in the second column. Row names should never be placed in this table (the default row names should always be 1:n). Column names can be any character string. The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

Below are three correctly formatted tables.

Category	Frequency
<=6	11800
7-12	57100
13-19	14800
20+	3900

Category	Frequency
LE6	11800
7I12	57100
13I19	14800
GE20	3900

Category	Frequency
<7	11800
7I12	57100
13-19	14800
>=20	3900

Value

A censored univariate table (observed.table) of probabilities is reweighted to match probabilities in an uncensored numeric vector (estimated.table). If observed.table consist of frequencies, the reweight.contingencytable() function changes the frequencies to probabilities.

Examples

```
# use preloaded univariate table for observed table
observed.table<-univariatetable.csv

# estimate parameters from observed table
pars=cnbinom.pars(observed.table)
mu=pars$Average
r = pars$Dispersion

# create estimated table
# truncated negative binomial probabilities
# uncensored range is from 1:15
library(truncdist)
estimated.table<-dtrunc(1:15, size = r, mu = mu, spec = "nbinom", a = 1-1, b = 15)
names(estimated.table)<-1:15

# reweight observed table to estimated table
results<-reweight.univariatetable(observed.table, estimated.table)

# check results for >=9
reweightedresults<-sum(results[9:15])
# observed.table$V2[5] = 4.1
observedresults<-observed.table$V2[5]/sum(observed.table$V2)
# matching probabilities
all.equal(reweightedresults, observedresults)
```

row.marginal *Row marginal table from contingency table*

Description

This function inputs a contingency table and produces a univariate frequency table from the row marginals.

Usage

```
row.marginal(contingencytable)
```

Arguments

`contingencytable`
A censored contingency table. Accepted classes are a data.frame or matrix. See Details section for formatting.

Details

Table Format:
The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

The column names should be the Y category values. The first column should be the X category values and the row names can be arbitrary. The inside of the table are X * Y cross tabulation, which are either positive frequency values or probabilities. The row and column marginal totals corresponding to their X and Y category values need to be placed in this table. The top left, top right, and bottom left corners of the table should be NA or blank. The bottom right corner can be a total cross tabulation sum value, NA, or blank. The table below is a formatted example.

NA	<20	20-30	>30	NA
<5	18	19	8	45
5-9	13	8	12	33
>=10	7	5	10	21
NA	38	32	31	NA

Value

A univariate frequency table (data.frame) created from the row marginals.

Examples

```
# contingencytable.csv is a preloaded contingency table example
row.marginal(contingencytable.csv)
```

`seedmatrix` *An uncensored seed matrix from censored contingency table*

Description

To implement the mipfp R package in our 'rec' function, an initial N-dimensional array (called a seed) needs to be provided. This function creates a seed matrix for an uncensored contingency table (Xlowerbound:Xupperbound, Ylowerbound:Yupperbound) when provided a censored contingency table.

Usage

```
seedmatrix(censoredtable, Xlowerbound, Xupperbound, Ylowerbound, Yupperbound)
```

Arguments

`censoredtable`	A censored contingency table. A data.frame and matrix are accepted table classes. See Details section for formatting.
`Xlowerbound`	A numeric class value to represent a lower bound for the row values. The value must strictly be >= 0 and cannot be greater than the lowest row category value (e.g. the lower bound cannot be 6 if a table has '< 5' as a row category value)
`Xupperbound`	A numeric class value to represent an upper bound for the row values. The value cannot be less than the highest row category value (e.g. the upper bound cannot be 90 if a table has '> 100' as a row category value).
`Ylowerbound`	Same description as Xlowerbound but this argument is for the columns in contingency table).
`Yupperbound`	Same description as Xupperbound but this argument is for the columns in contingency table).

Details

Table Format:
The only symbols accepted for censored data are listed below. Note, less than or equal to (<= and LE) is not equivalent to less than (< and L) and greater than or equal to (>=, +, and GE) is not equivalent to greater than (> and G). Also, calculations use closed intervals.

- left censoring: <, L, <=, LE
- interval censoring: - or I (symbol has to be placed in the middle of the two category values)
- right censoring: >, >=, +, G, GE
- uncensored: no symbol (only provide category value)

The column names should be the Y category values. The first column should be the X category values and the row names can be arbitrary. The inside of the table are X * Y cross tabulation, which are either positive frequency values or probabilities. The row and column marginal totals corresponding to their X and Y category values need to be placed in this table. The top left, top right, and bottom left corners of the table should be NA or blank. The bottom right corner can be a total cross tabulation sum value, NA, or blank. The table below is a formatted example.

NA	<20	20-30	>30	NA
<5	18	19	8	45
5-9	13	8	12	33

>=10	7	5	10	21
NA	38	32	31	NA

Value

The output is a list of two tables: 'Exact' and 'Probabilities'. Both tables have rows ranging from Xlowerbound, Xlowerbound + 1,..., Xupperbound and columns ranging from Ylowerbound, Ylowerbound + 1,..., Yupperbound. Also, the marginals are not placed in either table. The 'Exact' table uses the cross tabulation probabilities corresponding to the censored categories and repeats these values to the newly created and compatible uncensored cross tabulations (frequencies are changed to probabilities). The 'Probabilities' table takes the the cells of the 'Exact' table are divides by sum(Exact). The rec() function uses the 'Probabilities' table.

Examples

```
# create a censored contingency table
contingencytable<-matrix(1:9,
                         nrow = 3, ncol = 3)
rowmarginal<-apply(contingencytable,1,sum)
contingencytable<-cbind(contingencytable, rowmarginal)
colmarginal<-apply(contingencytable,2,sum)
contingencytable<-rbind(contingencytable, colmarginal)
row.names(contingencytable)[row.names(contingencytable)=="colmarginal"]<-""
contingencytable<-data.frame(c("<5", "5I9", ">9", NA), contingencytable)
colnames(contingencytable)<-c(NA,"<=19","20-30",">=31", NA)

# look at the seed for this example (probabilities)
seed.output<-seedmatrix(contingencytable, 1,
                        15, 15, 35)$Probabilities
```

univariate_table_example

Univariate frequency table example

Description

In 2011, a Nepal Living Standards Survey provided multiple censored freqency tables containing household size (percent). Here we show the urban household size table.

univariatetable.csv is a preloaded example that is correctly formatted for the revengc package. Below shows how a user should read in a csv file to have the same format:

univariatetable.csv<-read.csv("filename.csv", row.names = NULL, header= FALSE, check.names=FALSE)

Usage

```
univariatetable.csv
```

References

National Planning Commissions Secretariat, Government of Nepal. (2011). *Nepal Living Standards Survey*. Page 28. Retrieved from: http://siteresources.worldbank.org/INTLSMS/Resources/3358986-1181743055198/3877319-1329489437402/Statistical_Report_Vol1.pdf

Index

∗Topic **Poisson**
- rec, 6

∗Topic **censored**
- cnbinom.pars, 2
- rec, 6

∗Topic **contingency**
- rec, 6

∗Topic **count data**
- cnbinom.pars, 2
- rec, 6

∗Topic **dispersion**
- cnbinom.pars, 2
- rec, 6

∗Topic **frequency table**
- cnbinom.pars, 2
- rec, 6

∗Topic **mu**
- cnbinom.pars, 2

∗Topic **negative binomial**
- cnbinom.pars, 2
- rec, 6

∗Topic **truncated**
- rec, 6

∗Topic **univariate table**
- cnbinom.pars, 2
- rec, 6

cnbinom.pars, 2
column.marginal, 4
contingency_table_example, 5
contingencytable.csv
(contingency_table_example), 5

rec, 6
reweight.contingencytable, 11
reweight.univariatetable, 13
row.marginal, 15

seedmatrix, 16

univariate_table_example, 18
univariatetable.csv
(univariate_table_example), 18