Package ‘TeachingSampling’

April 21, 2020

Type Package
Title Selection of Samples and Parameter Estimation in Finite Population
License GPL (>= 2)
Version 4.1.1
Date 2020-04-21
Author Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>
Maintainer Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>
Depends R (>= 3.5), dplyr, magrittr
Description Allows the user to draw probabilistic samples and make inferences from a finite population based on several sampling designs.
Encoding UTF-8
RoxygenNote 7.1.0
NeedsCompilation no
Repository CRAN
Date/Publication 2020-04-21 21:50:03 UTC

R topics documented:

BigCity ................................................................. 3
BigLucy ................................................................. 4
Deltakl ................................................................. 6
Domains ............................................................... 7
E.1SI ................................................................. 8
E.2SI ................................................................. 10
E.BE ................................................................. 13
E.Beta ................................................................. 14
E.piPS .............................................................. 17
E.PO ................................................................. 19
E.PPS .............................................................. 20
E.Quantile .......................................................... 21
R topics documented:

E.SI .................................................. 23
E.STpiPS .......................................... 26
E.STPPS ........................................... 27
E.STSI ............................................. 29
E.SY .................................................. 31
E.Trim .............................................. 32
E.UC .................................................. 33
E.WR .................................................. 37
GREG.SI ........................................... 38
HH ..................................................... 42
HT ..................................................... 45
Ik ...................................................... 51
IkRS .................................................. 52
IkWR .................................................. 53
IPFP .................................................. 54
Lucy ................................................. 56
nk ..................................................... 57
OrderWR ............................................ 58
p.WR .................................................. 60
Pik .................................................... 61
PikHol .............................................. 63
Pikl ................................................... 65
PikPPS .............................................. 66
PikSTPPS ......................................... 68
S.BE .................................................. 70
S.piPS .............................................. 72
S.PO .................................................. 73
S.PPS ................................................ 75
S.SI .................................................. 76
S.STpiPS ......................................... 78
S.STPPS .......................................... 80
S.STSI ............................................. 81
S.SY .................................................. 83
S.WR .................................................. 85
Support ............................................ 86
SupportRS ........................................ 87
SupportWR ....................................... 89
T.SIC ............................................... 90
VarHT .............................................. 92
VarSYGHT ........................................ 93
Wk .................................................... 95

Index ............................................ 100
Description

This data set corresponds to some socioeconomic variables from 150266 people of a city in a particular year.

Usage

data(BigCity)

Format

**HHID** The identifier of the household. It corresponds to an alphanumerical sequence (four letters and five digits).

**PersonID** The identifier of the person within the household. NOTE it is not a unique identifier of a person for the whole population. It corresponds to an alphanumerical sequence (five letters and two digits).

**Stratum** Households are located in geographic strata. There are 119 strata across the city.

**PSU** Households are clustered in cartographic segments defined as primary sampling units (PSU). There are 1664 PSU and they are nested within strata.

**Zone** Segments clustered within strata can be located within urban or rural areas along the city.

**Sex** Sex of the person.

**Income** Per capita monthly income.

**Expenditure** Per capita monthly expenditure.

**Employment** A person’s employment status.

**Poverty** This variable indicates whether the person is poor or not. It depends on income.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

Lucy,BigLucy
Examples

```r
data(BigCity)
attach(BigCity)

estima <- data.frame(Income, Expenditure)
# The population totals
colSums(estima)
# Some parameters of interest
table(Poverty, Zone)
xtabs(Income ~ Poverty + Zone)
# Correlations among characteristics of interest
cor(estima)
# Some useful histograms
hist(Income)
hist(Expenditure)
# Some useful plots
boxplot(Income ~ Poverty)
barplot(table(Employment))
pie(table(MaritalST))
```

---

**BigLucy**  
*Full Business Population Database*

**Description**

This data set corresponds to some financial variables of 85396 industrial companies of a city in a particular fiscal year.

**Usage**

```r
data(BigLucy)
```

**Format**

- **ID** The identifier of the company. It correspond to an alphanumeric sequence (two letters and three digits)
- **Ubication** The address of the principal office of the company in the city
- **Level** The industrial companies are discriminated according to the Taxes declared. There are small, medium and big companies
- **Zone** The country is divided by counties. A company belongs to a particular zone according to its cartographic location.
- **Income** The total amount of a company's earnings (or profit) in the previous fiscal year. It is calculated by taking revenues and adjusting for the cost of doing business
- **Employees** The total number of persons working for the company in the previous fiscal year
- **Taxes** The total amount of a company's income Tax
**SPAM** Indicates if the company uses the Internet and WEBmail options in order to make self-propaganda.

**ISO** Indicates if the company is certified by the International Organization for Standardization.

**Years** The age of the company.

**Segments** Cartographic segments by county. A segment comprises in average 10 companies located close to each other.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

Lucy, BigCity

**Examples**

```r
data(BigLucy)
attach(BigLucy)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
# The population totals
colSums(estima)
# Some parameters of interest
table(SPAM, Level)
xtabs(Income ~ Level + SPAM)
# Correlations among characteristics of interest
cor(estima)
# Some useful histograms
hist(Income)
hist(Taxes)
hist(Employees)
# Some useful plots
boxplot(Income ~ Level)
barplot(table(Level))
pie(table(SPAM))
```
**Description**

Computes the Variance-Covariance matrix of the sample membership indicators in the population given a fixed sample size design.

**Usage**

```r
Deltakl(N, n, p)
```

**Arguments**

- `N`: Population size
- `n`: Sample size
- `p`: A vector containing the selection probabilities of a fixed size without replacement sampling design. The sum of the values of this vector must be one.

**Details**

The $\Delta_{kl}$th unit of the Variance-Covariance matrix of the sample membership indicators is defined as

$$\Delta_{kl} = \pi_{kl} - \pi_k \pi_l$$

**Value**

The function returns a symmetric matrix of size $N \times N$ containing the variances-covariances among the sample membership indicators for each pair of units in the finite population.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

VarHT, Pikl, Pik
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)

# The sample size is n=2
n <- 2

# p is the probability of selection of every sample.
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)

# Note that the sum of the elements of this vector is one
sum(p)

# Computation of the Variance-Covariance matrix of the sample membership indicators
Deltakl(N, n, p)

<table>
<thead>
<tr>
<th>Domains</th>
<th>Domains Indicator Matrix</th>
</tr>
</thead>
</table>

## Description

Creates a matrix of domain indicator variables for every single unit in the selected sample or in the entire population.

## Usage

Domains(y)

## Arguments

- **y**: Vector of the domain of interest containing the membership of each unit to a specified category of the domain.

## Details

Each value of y represents the domain which a specified unit belongs.

## Value

The function returns a \( n \times p \) matrix, where \( n \) is the number of units in the selected sample and \( p \) is the number of categories of the domain of interest. The values of this matrix are zero, if the unit does not belongs to a specified category and one, otherwise.

## Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

## References


See Also

E.SI

Examples

#################
## Example 1
#################
# This domain contains only two categories: “yes” and “no”
x <- as.factor(c("yes","yes","yes","no","no","no","no","yes","yes"))
Domains(x)

#################
## Example 2
#################
# Uses the Lucy data to draw a random sample of units according
# to a SI design
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
n <- 400
sam <- sample(N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variable SPAM is a domain of interest
Doma <- Domains(SPAM)
Doma
# HT estimation of the absolute domain size for every category in the domain
# of interest
E.SI(N,n,Doma)

#################
## Example 3
#################
# Following with Example 2...
# The variables of interest are: Income, Employees and Taxes
# This function allows to estimate the population total of this variables for every
# category in the domain of interest SPAM
estima <- data.frame(Income, Employees, Taxes)
SPAM.no <- estima*Doma[,1]
SPAM.yes <- estima*Doma[,2]
E.SI(N,n,SPAM.no)
E.SI(N,n,SPAM.yes)
Description
This function computes the Horvitz-Thompson estimator of the population total according to a single stage sampling design.

Usage
E.1SI(NI, nI, y, PSU)

Arguments
NI Population size of Primary Sampling Units.
nI Sample size of Primary Sampling Units.
y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample.
PSU Vector identifying the membership to the strata of each unit in the population.

Details
The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest.

Value
This function returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation.

Author(s)
Hugo Andres Gutierrez Rojas <hugogutierrez at gmail.com>

References
Gutierrez, H. A. (2009), Estrategias de muestreo: Diseño de encuestas y estimación de parámetros. Editorial Universidad Santo Tomas

See Also
E.2SI

Examples

data('BigCity')
Households <- BigCity %% group_by(HHID) %%
summarise(Stratum = unique(Stratum),
          PSU = unique(PSU),
          Persons = n(),
          Income = sum(Income),
          )
Expenditure = sum(Expenditure))

attach(Households)
UI <- levels(as.factor(Households$PSU))
NI <- length(UI)
nI <- 100

samI <- S.SI(NI, nI)
sampleI <- UI[samI]

CityI <- Households[which(Households$PSU %in% sampleI), ]
attach(CityI)
area <- as.factor(CityI$PSU)
estima <- data.frame(CityI$Persons, CityI$Income, CityI$Expenditure)

E.1SI(NI, nI, estima, area)

---

E.2SI

**Estimation of the Population Total under Two Stage Simple Random Sampling Without Replacement**

**Description**

Computes the Horvitz-Thompson estimator of the population total according to a 2SI sampling design

**Usage**

E.2SI(NI, nI, Ni, ni, y, PSU)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NI</td>
<td>Population size of Primary Sampling Units</td>
</tr>
<tr>
<td>nI</td>
<td>Sample size of Primary Sampling Units</td>
</tr>
<tr>
<td>Ni</td>
<td>Vector of population sizes of Secondary Sampling Units selected in the first draw</td>
</tr>
<tr>
<td>ni</td>
<td>Vector of sample sizes of Secondary Sampling Units</td>
</tr>
<tr>
<td>y</td>
<td>Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample</td>
</tr>
<tr>
<td>PSU</td>
<td>Vector identifying the membership to the strata of each unit in the population</td>
</tr>
</tbody>
</table>

**Details**

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation
Value

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezstro@gmail.com>

References


See Also

S.SI

Examples

```
############
## Example 1
############
# Uses Lucy data to draw a two-stage simple random sample
# according to a 2SI design. Zone is the clustering variable
data(Lucy)
attach(Lucy)
summary(Zone)
# The population of clusters or Primary Sampling Units
UI<-c("A","B","C","D","E")
NI <- length(UI)
# The sample size is nI=3
nI <- 3
# Selects the sample of PSUs
samI<-S.SI(NI,nI)
dataI<-UI[samI]
dataI
# The sampling frame of Secondary Sampling Unit is saved in Lucy1 ... Lucy3
Lucy1<-Lucy[which(Zone==dataI[1]),]
Lucy2<-Lucy[which(Zone==dataI[2]),]
Lucy3<-Lucy[which(Zone==dataI[3]),]
# The size of every single PSU
N1<-dim(Lucy1)[1]
N2<-dim(Lucy2)[1]
N3<-dim(Lucy3)[1]
Ni<-c(N1,N2,N3)
# The sample size in every PSI is 135 Secondary Sampling Units
n1<-135
n2<-135
n3<-135
ni<-c(n1,n2,n3)
```
# Selects a sample of Secondary Sampling Units inside the PSUs
sam1<-S.SI(N1,n1)
sam2<-S.SI(N2,n2)
sam3<-S.SI(N3,n3)
# The information about each Secondary Sampling Unit in the PSUs
# is saved in data1 ... data3
data1<-Lucy1[sam1,]
data2<-Lucy2[sam2,]
data3<-Lucy3[sam3,]
# The information about each unit in the final selected sample is saved in data
data<-rbind(data1, data2, data3)
attach(data)
# The clustering variable is Zone
Cluster <- as.factor(as.integer(Zone))
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
# Estimation of the Population total
E.2SI(NI,nI,Ni,ni,estima,Cluster)

########################################################################
#### Example 2 Total Census to the entire population
########################################################################
# Uses Lucy data to draw a cluster random sample
# according to a SI design ...
# Zone is the clustering variable
data(Lucy)
attach(Lucy)
summary(Zone)
# The population of clusters
UI<-c("A","B","C","D","E")
NI <- length(UI)
# The sample size equals to the population size of PSU
nI <- NI
# Selects every single PSU
samI<-S.SI(NI,nI)
dataI<-UI[samI]
dataI
# The sampling frame of Secondary Sampling Unit is saved in Lucy1 ... Lucy5
Lucy1<-Lucy[which(Zone==dataI[1]),]
Lucy2<-Lucy[which(Zone==dataI[2]),]
Lucy3<-Lucy[which(Zone==dataI[3]),]
Lucy4<-Lucy[which(Zone==dataI[4]),]
Lucy5<-Lucy[which(Zone==dataI[5]),]
# The size of every single PSU
N1<-dim(Lucy1)[1]
N2<-dim(Lucy2)[1]
N3<-dim(Lucy3)[1]
N4<-dim(Lucy4)[1]
N5<-dim(Lucy5)[1]
Ni<-c(N1,N2,N3,N4,N5)
# The sample size of Secondary Sampling Units equals to the size of each PSU
nI<-NI
n2<-N2
n3<-N3
n4<-N4
n5<-N5
ni<-c(n1,n2,n3,n4,n5)
# Selects every single Secondary Sampling Unit inside the PSU
sam1<-S.SI(N1,n1)
sam2<-S.SI(N2,n2)
sam3<-S.SI(N3,n3)
sam4<-S.SI(N4,n4)
sam5<-S.SI(N5,n5)
# The information about each unit in the cluster is saved in Lucy1 ... Lucy5
data1<-Lucy1[sam1,]
data2<-Lucy2[sam2,]
data3<-Lucy3[sam3,]
data4<-Lucy4[sam4,]
data5<-Lucy5[sam5,]
# The information about each Secondary Sampling Unit
# in the sample (census) is saved in data
data<-rbind(data1, data2, data3, data4, data5)
attach(data)
# The clustering variable is Zone
Cluster <- as.factor(as.integer(Zone))
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
# Estimation of the Population total
E.2SI(NI,nI,Ni,ni,estima,Cluster)
# Sampling error is null

---

**E.BE**

*Estimation of the Population Total under Bernoulli Sampling Without Replacement*

**Description**

Computes the Horvitz-Thompson estimator of the population total according to a BE sampling design

**Usage**

`E.BE(y, prob)`

**Arguments**

- **y**: Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
- **prob**: Inclusion probability for each unit in the population
Details

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation under an BE sampling design.

Value

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

S.BE

Examples

# Uses the Lucy data to draw a Bernoulli sample
data(Lucy)
attach(Lucy)

N <- dim(Lucy)[1]
n=400
prob=n/N
sam <- S.BE(N,prob)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.BE(estima,prob)

---

**E.Beta**

*Estimation of the population regression coefficients under SI designs*

Description

Computes the estimation of regression coefficients using the principles of the Horvitz-Thompson estimator.
Usage

E.Beta(N, n, y, x, ck=1, b0=FALSE)

Arguments

N  The population size
n  The sample size
y  Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
x  Vector, matrix or data frame containing the recollected auxiliary information for every unit in the selected sample
ck  By default equals to one. It is a vector of weights induced by the structure of variance of the supposed model
b0  By default FALSE. The intercept of the regression model

Details

Returns the estimation of the population regression coefficients in a supposed linear model, its estimated variance and its estimated coefficient of variation under an SI sampling design

Value

The function returns a vector whose entries correspond to the estimated parameters of the regression coefficients

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

GREG.SI

Examples

# Example 1: Linear models involving continuous auxiliary information
# Draws a simple random sample without replacement
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N, n)
# The information about the units in the sample
# is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)

охрана#### common mean model

estima<-data.frame(Income, Employees, Taxes)
x <- rep(1,n)
E.Beta(N, n, estima,x,ck=1,b0=FALSE)

##### common ratio model

estima<-data.frame(Income)
x <- data.frame(Employees)
E.Beta(N, n, estima,x,ck=x,b0=FALSE)

##### Simple regression model without intercept

estima<-data.frame(Income, Employees)
x <- data.frame(Taxes)
E.Beta(N, n, estima,x,ck=1,b0=FALSE)

##### Multiple regression model without intercept

estima<-data.frame(Income)
x <- data.frame(Employees, Taxes)
E.Beta(N, n, estima,x,ck=1,b0=FALSE)

##### Simple regression model with intercept

estima<-data.frame(Income, Employees)
x <- data.frame(Taxes)
E.Beta(N, n, estima,x,ck=1,b0=TRUE)

##### Multiple regression model with intercept

estima<-data.frame(Income)
x <- data.frame(Employees, Taxes)
E.Beta(N, n, estima,x,ck=1,b0=TRUE)

###################################################################
## Example 2: Linear models with discrete auxiliary information
###################################################################

# Draws a simple random sample without replacement
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N,n)
# The information about the sample units is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The auxiliary information
Doma<-Domains(Level)

########### Poststratified common mean model
estima<-data.frame(Income, Employees, Taxes)
E.Beta(N, n, estima, Doma, ck=1, b0=FALSE)

########### Poststratified common ratio model
estima<-data.frame(Income, Employees)
x<-Doma*Taxes
E.Beta(N, n, estima, x, ck=1, b0=FALSE)

---

**E.piPS**

*Estimation of the Population Total under Probability Proportional to Size Sampling Without Replacement*

**Description**

Computes the Horvitz-Thompson estimator of the population total according to a \( \pi \)PS sampling design

**Usage**

E.piPS(y, Pik)

**Arguments**

- **y** Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
- **Pik** Vector of inclusion probabilities for each unit in the selected sample

**Details**

Returns the estimation of the population total of every single variable of interest, its estimated variance and its estimated coefficient of variation under a \( \pi \)PS sampling design. This function uses the results of approximate expressions for the estimated variance of the Horvitz-Thompson estimator.
Value

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

S.piPS

Examples

# Uses the Lucy data to draw a sample according to a piPS
# without replacement design
data(Lucy)
attach(Lucy)
# The inclusion probability of each unit is proportional to the variable Income
# The selected sample of size n=400
n <- 400
res <- S.piPS(n, Income)
sam <- res[,1]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,
attach(data)
names(data)
# Pik.s is the inclusion probability of every single unit in the selected sample
Pik.s <- res[,2]
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.piPS(estima,Pik.s)
# Same results than HT function
HT(estima, Pik.s)
Description

Computes the Horvitz-Thompson estimator of the population total according to a PO sampling design

Usage

E.PO(y, Pik)

Arguments

y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample

Pik Vector of inclusion probabilities for each unit in the selected sample

Details

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation under a PO sampling design

Value

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

S.PO
Examples

# Uses the Lucy data to draw a Poisson sample
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
# The population size is 2396. The expected sample size is 400
# The inclusion probability is proportional to the variable Income
n <- 400
Pik <- n*Income/sum(Income)
# The selected sample
sam <- S.PO(N, Pik)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The inclusion probabilities of each unit in the selected sample
inclusion <- Pik[sam]
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.PO(estima, inclusion)

---

**E.PPS**

*Estimation of the Population Total under Probability Proportional to Size Sampling With Replacement*

**Description**

Computes the Hansen-Hurwitz estimator of the population total according to a probability proportional to size sampling with replacement design

**Usage**

E.PPS(y, pk)

**Arguments**

- `y`: Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
- `pk`: A vector containing selection probabilities for each unit in the sample

**Details**

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation estimated under a probability proportional to size sampling with replacement design
**Value**

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

`S.PPS, HH`

**Examples**

```r
# Uses the Lucy data to draw a random sample according to a
# PPS with replacement design
data(Lucy)
attach(Lucy)
# The selection probability of each unit is proportional to the variable Income
m <- 400
res <- S.PPS(m, Income)
# The selected sample
sam <- res[,1]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# pk.s is the selection probability of each unit in the selected sample
pk.s <- res[,2]
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.PPS(estima, pk.s)
```

---

**E.Quantile**

*Estimation of a Population Quantile*

**Description**

Computes the estimation of a population quantile using the principles of the Horvitz-Thompson estimator.
Usage

E.Quantile(y, Qn, Pik)

Arguments

y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
Qn Quantile of interest
Pik A vector containing inclusion probabilities for each unit in the sample. If missing, the function will assign the same weights to each unit in the sample

Details

Returns the estimation of the population quantile of every single variable of interest

Value

The function returns a vector whose entries correspond to the estimated quantiles of the variables of interest

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

HT

Examples

############
## Example 1
############
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vectors y and x give the values of the variables of interest
y<-c(32, 34, 46, 89, 35)
x<-c(52, 60, 75, 100, 50)
z<-cbind(y,x)
# Inclusion probabilities for a design of size n=2
Pik<-c(0.58, 0.34, 0.48, 0.33, 0.27)
# Estimation of the sample median
E.Quantile(y, 0.5)
# Estimation of the sample Q1
E.Quantile(x, 0.25)
# Estimation of the sample Q3
E.Quantile(z, 0.75)
# Estimation of the sample median
E.Quantile(z, 0.5, Pik)

############
## Example 2
############
# Uses the Lucy data to draw a PPS sample with replacement

data(Lucy)
attach(Lucy)

# The selection probability of each unit is proportional to the variable Income
# The sample size is m=400
m=400
res <- S.PPS(m, Income)
# The selected sample
sam <- res[,1]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
# The vector of selection probabilities of units in the sample
pk.s <- res[,2]
# The vector of inclusion probabilities of units in the sample
Pik.s <- 1-(1-pk.s)^m
# The information about the sample units is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
# Estimation of sample median
E.Quantile(estima, 0.5, Pik.s)

---

**E.SI**

_Estimation of the Population Total under Simple Random Sampling Without Replacement_

**Description**

Computes the Horvitz-Thompson estimator of the population total according to an SI sampling design

**Usage**

E.SI(N, n, y)
Arguments

N   Population size
n   Sample size
y   Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample

Details

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation under an SI sampling design

Value

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

S.SI

Examples

############
## Example 1
############
# Uses the Lucy data to draw a random sample of units according to a SI design
data(Lucy)
attach(Lucy)

N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.SI(N,n,estima)
Example 2

Following with Example 1. The variable SPAM is a domain of interest.

```r
Doma <- Domains(SPAM)
```

This function allows to estimate the size of each domain in SPAM.

```r
estima <- data.frame(Doma)
```

```r
E.SI(N,n,Doma)
```

Example 3

Following with Example 1. The variable SPAM is a domain of interest.

```r
Doma <- Domains(SPAM)
```

This function allows to estimate the parameters of the variables of interest for every category in the domain SPAM.

```r
estima <- data.frame(Income, Employees, Taxes)
```

```r
SPAM.no <- cbind(Doma[,1], estima*Doma[,1])
```

```r
SPAM.yes <- cbind(Doma[,1], estima*Doma[,2])
```

Before running the following lines, notice that:
- The first column always indicates the population size.
- The second column is an estimate of the size of the category in the domain SPAM.
- The remaining columns estimates the parameters of interest within the corresponding category in the domain SPAM.

```r
E.SI(N,n,SPAM.no)
```

```r
E.SI(N,n,SPAM.yes)
```

Example 4

Following with Example 1. The variable SPAM is a domain of interest and the variable ISO is a populational subgroup of interest.

```r
Doma <- Domains(SPAM)
```

```r
estima <- Domains(Zone)
```

Before running the following lines, notice that:
- The first column indicates whether the unit belongs to the first category of SPAM or not.
- The remaining columns indicates whether the unit belongs to the categories of Zone.

```r
SPAM.no <- data.frame(SpamNO=Doma[,1], Zones=estima*Doma[,1])
```

Before running the following lines, notice that:
- The first column indicates whether the unit belongs to the second category of SPAM or not.
- The remaining columns indicates whether the unit belongs to the categories of Zone.

```r
SPAM.yes <- data.frame(SpamYES=Doma[,2], Zones=estima*Doma[,2])
```

Before running the following lines, notice that:
- The first column always indicates the population size.
- The second column is an estimate of the size of the first category in the domain SPAM.
- The remaining columns estimates the size of the categories of Zone within the corresponding category of SPAM.
# Finally, note that the sum of the point estimates of the last two columns gives exactly the point estimate in the second column

\[ E.SI(N,n,\text{SPAM.no}) \]

# Before running the following lines, notice that:
# The first column always indicates the population size
# The second column is an estimate of the size of the second category in the domain SPAM
# The remaining columns estimates the size of the categories of Zone within the corresponding category of SPAM
# Finally, note that the sum of the point estimates of the last two columns gives exactly the point estimate in the second column

\[ E.SI(N,n,\text{SPAM.yes}) \]

---

**E.STpiPS**  
*Estimation of the Population Total under Stratified Probability Proportional to Size Sampling Without Replacement*

### Description
Computes the Horvitz-Thompson estimator of the population total according to a probability proportional to size sampling without replacement design in each stratum.

### Usage

\[
E.STpiPS(y, pik, S)
\]

### Arguments

- **y**: Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample.
- **pik**: A vector containing inclusion probabilities for each unit in the sample.
- **S**: Vector identifying the membership to the strata of each unit in selected sample.

### Details
Returns the estimation of the population total of every single variable of interest, its estimated standard error, its estimated coefficient of variation and its corresponding DEFF in all of the strata and finally in the entire population.

### Value
The function returns an array composed by several matrices representing each variable of interest. The columns of each matrix correspond to the estimated parameters of the variables of interest in each stratum and in the entire population.

### Author(s)
Hugo Andres Gutierrez Rojas <hagutierrezrezo@gmail.com>
References


See Also

S.STpiPS

Examples

# Uses the Lucy data to draw a stratified random sample
# according to a PPS design in each stratum

data(Lucy)
attach(Lucy)
# Level is the stratifying variable
summary(Level)

# Defines the size of each stratum
N1<-summary(Level)[[1]]
N2<-summary(Level)[[2]]
N3<-summary(Level)[[3]]
N1;N2;N3

# Defines the sample size at each stratum
n1<-N1
n2<-100
n3<-200
nh<-c(n1,n2,n3)
nh
# Draws a stratified sample
S<-Level
x<-Employees

res <- S.STpiPS(S, x, nh)
sam <- res[,1]
pik <- res[,2]

data <- Lucy[sam,]
attach(data)
estima <- data.frame(Income, Employees, Taxes)
E.STpiPS(estima,pik,Level)
Description

Computes the Hansen-Hurwitz estimator of the population total according to a probability proportional to size sampling with replacement design

Usage

E.STPPS(y, pk, mh, S)

Arguments

y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
pk A vector containing selection probabilities for each unit in the sample
mh Vector of sample size in each stratum
S Vector identifying the membership to the strata of each unit in selected sample

Details

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation in all of the stratum and finally in the entire population

Value

The function returns an array composed by several matrices representing each variable of interest. The columns of each matrix correspond to the estimated parameters of the variables of interest in each stratum and in the entire population

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

S.STPPS

Examples

# Uses the Lucy data to draw a stratified random sample
# according to a PPS design in each stratum

data(Lucy)
attract(Lucy)
E.STSI

Estimation of the Population Total under Stratified Simple Random Sampling Without Replacement

Description

Computes the Horvitz-Thompson estimator of the population total according to a STSI sampling design

Usage

E.STSI(S, Nh, nh, y)

Arguments

S Vector identifying the membership to the strata of each unit in the population
Nh Vector of stratum sizes
nh Vector of sample sizes in each stratum
y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample

Details

Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation in all of the strata and finally in the entire population.
Value

The function returns an array composed by several matrices representing each variable of interest. The columns of each matrix correspond to the estimated parameters of the variables of interest in each stratum and in the entire population.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

S.STSI

Examples

############
## Example 1
############
# Uses the Lucy data to draw a stratified random sample according to a SI design in each stratum

data(Lucy)
attach(Lucy)
# Level is the stratifying variable
summary(Level)
# Defines the size of each stratum
N1<-summary(Level)[[1]]
N2<-summary(Level)[[2]]
N3<-summary(Level)[[3]]
N1;N2;N3
Nh <- c(N1,N2,N3)
# Defines the sample size at each stratum
n1<-N1
n2<-100
n3<-200
nh<-c(n1,n2,n3)
# Draws a stratified sample
sam <- S.STSI(Level, Nh, nh)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.SY

Estimation of the Population Total under Systematic Sampling Without Replacement

Description
Computes the Horvitz-Thompson estimator of the population total according to an SY sampling design

Usage
E.SY(N, a, y)

Arguments
- **N**: Population size
- **a**: Number of groups dividing the population
- **y**: Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample

Details
Returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation under an SY sampling design

Value
The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest

Author(s)
Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>
References


See Also

S.SY

Examples

# Uses the Lucy data to draw a Systematic sample
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
# The population is divided in 6 groups
# The selected sample
sam <- S.SY(N,6)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.SY(N,6,estima)

---

**E.Trim**

Weight Trimming and Redistribution

Description

This function performs a method of trimming sampling weights based on the evenly redistribution of the net amount of weight loss among units whose weights were not trimmed. This way, the sum of the trimmed sampling weights remains the same as the original weights.

Usage

E.Trim(dk, L, U)

Arguments

dk Vector of original sampling weights.
L Lower bound for weights.
U Upper bound for weights.
Details

The function returns a vector of trimmed sampling weights.

Value

This function returns a vector of trimmed weights.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro at gmail.com> with contributions from Javier Nunez <javier_nunez at inec.gob.ec>

References


Examples

```r
# Example 1
dk <- c(1, 1, 1, 10)
summary(dk)
L <- 1
U <- 3.5 * median(dk)
dkTrim <- E.Trim(dk, L, U)
sum(dk)
sum(dkTrim)

# Example 2
dk <- rnorm(1000, 10, 10)
L <- 1
U <- 3.5 * median(dk)
dkTrim <- E.Trim(dk, L, U)
sum(dk)
sum(dkTrim)
summary(dk)
summary(dkTrim)
hist(dk)
hist(dkTrim)
```

---

**E.UC**

*Estimation of the Population Total and its variance using the Ultimate Cluster technique*
Description

This function computes a weighted estimator of the population total and estimates its variance by using the Ultimate Cluster technique. This approximation performs well in many sampling designs. The user specifically needs to declare the variables of interest, the primary sampling units, the strata, and the sampling weights for every single unit in the sample.

Usage

E.UC(S, PSU, dk, y)

Arguments

S Vector identifying the membership to the strata of each unit in selected sample.
PSU Vector identifying the membership to the strata of each unit in the population.
dk Sampling weights of the units in the sample.
y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample.

Details

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest.

Value

This function returns the estimation of the population total of every single variable of interest, its estimated standard error and its estimated coefficient of variation.

Author(s)

Hsugo Andres Gutierrez Rojas <hugogutierrez@gmail.com>

References

Gutierrez, H. A. (2009), Estrategias de muestreo: Diseño de encuestas y estimación de parámetros. Editorial Universidad Santo Tomas

See Also

E.2SI

Examples

########################################################################
## Example 1: 
## Stratified Two-stage SI 
########################################################################
data('BigCity')
FrameI <- BigCity %>% group_by(PSU) %>%
  summarise(Stratum = unique(Stratum),
  Persons = n(),
  Income = sum(Income),
  Expenditure = sum(Expenditure))
attach(FrameI)
sizes = FrameI %>% group_by(Stratum) %>%
  summarise(NIh = n(),
  nIh = 2,
  dI = NIh/nIh)
NIh <- sizes$NIh
nIh <- sizes$nIh
samI <- S.STSI(Stratum, NIh, nIh)
UI <- levels(as.factor(FrameI$PSU))
sampleI <- UI[samI]
FrameII <- left_join(sizes, BigCity[which(BigCity$PSU %in% sampleI), ])
attach(FrameII)
HHdb <- FrameII %>%
  group_by(PSU) %>%
  summarise(Ni = length(unique(HHID)))
Ni <- as.numeric(HHdb$Ni)
ni <- ceiling(Ni * 0.1)
ni
sum(ni)
sam = S.SI(Ni[1], ni[1])
clusterII = FrameII[which(FrameII$PSU == sampleI[1]), ]
sam.HH <- data.frame(HHID = unique(clusterII$HHID)[sam])
clusterHH <- left_join(sam.HH, clusterII, by = "HHID")
clusterHH$dk <- Ni[1]/ni[1]
clusterHH$dk <- clusterHH$dk * clusterHH$dk
data = clusterHH
for (i in 2:length(Ni)) {
  sam = S.SI(Ni[i], ni[i])
clusterII = FrameII[which(FrameII$PSU == sampleI[i]), ]
sam.HH <- data.frame(HHID = unique(clusterII$HHID)[sam])
clusterHH <- left_join(sam.HH, clusterII, by = "HHID")
clusterHH$dk <- Ni[i]/ni[i]
clusterHH$dk <- clusterHH$dk * clusterHH$dk
data1 = clusterHH
data = rbind(data, data1)
}
sum(data$dk)
attach(data)
estima <- data.frame(Income, Expenditure)
area <- as.factor(PSU)
stratum <- as.factor(Stratum)
E.UC(stratum, area, dk, estima)

################################
## Example 2: 
## Self weighted Two-stage SI ##
################################
data('BigCity')
FrameI <- BigCity %>% group_by(PSU) %>%
  summarise(Stratum = unique(Stratum),
             Households = length(unique(HHID)),
             Income = sum(Income),
             Expenditure = sum(Expenditure))

attach(FrameI)
sizes = FrameI %>% group_by(Stratum) %>%
  summarise(NIh = n(),
             nIh = 2)

NIh <- sizes$NIh
nIh <- sizes$nIh

resI <- S.STpiPS(Stratum, Households, nIh)
head(resI)
samI <- resI[,1]
pI <- resI[,2]
UI <- levels(as.factor(FrameI$PSU))
sampleI <- data.frame(PSU = UI[samI], dI = 1/pI)

FrameII <- left_join(sampleI,
                      BigCity[which(BigCity$PSU %in% sampleI[,1]), ])

attach(FrameII)
HHdb <- FrameII %>%
group_by(PSU) %>%
  summarise(Ni = length(unique(HHID)))
Ni <- as.numeric(HHdb$Ni)
ni <- 5

sam = S.SI(Ni[1], ni)
ciusterII = FrameII[which(FrameII$PSU == sampleI$PSU[1]), ]

sam.HH <- data.frame(HHID = unique(clusterII$HHID)[sam])
clusterHH <- left_join(sam.HH, clusterII, by = "HHID")

clusterHH$dki <- Ni[1]/ni
clusterHH$dk <- clusterHH$dI * clusterHH$dki
data = clusterHH
for (i in 2:length(Ni)) {
  sam = S.SI(Ni[i], ni)
  clusterII = FrameII[which(FrameII$PSU == sampleI$PSU[i]), ]
  sam.HH <- data.frame(HHID = unique(clusterII$HHID)[sam])
  clusterII <- left_join(sam.HH, clusterII, by = "HHID")
  clusterHH <- clusterII$PSU == sampleI$PSU[i]
  clusterHH$dk <- Ni[i]/ni
  data1 = clusterHH
  data = rbind(data, data1)
}

sum(data$dk)
attach(data)
estima <- data.frame(Income, Expenditure)
area <- as.factor(PSU)
stratum <- as.factor(Stratum)
E.UC(stratum, area, dk, estima)

---

### E.WR

**Estimation of the Population Total under Simple Random Sampling With Replacement**

**Description**

Computes the Hansen-Hurwitz estimator of the population total according to a simple random sampling with replacement design

**Usage**

E.WR(N, m, y)

**Arguments**

- **N**  
  Population size
- **m**  
  Sample size
- **y**  
  Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample

**Details**

Returns the estimation of the population total of every single variable of interest, its estimated variance and its estimated coefficient of variation estimated under an simple random with replacement design

**Value**

The function returns a data matrix whose columns correspond to the estimated parameters of the variables of interest
**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

S.WR

**Examples**

```r
# Uses the Lucy data to draw a random sample according to a WR design
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
m <- 400
sam <- S.WR(N,m)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.WR(N,m,estima)
```

---

**GREG.SI**

*The Generalized Regression Estimator under SI sampling design*

**Description**

Computes the generalized regression estimator of the population total for several variables of interest under simple random sampling without replacement

**Usage**

`GREG.SI(N, n, y, x, tx, b, b0=FALSE)`
Arguments

- **N**: The population size
- **n**: The sample size
- **y**: Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
- **x**: Vector, matrix or data frame containing the recollected auxiliary information for every unit in the selected sample
- **tx**: Vector containing the populations totals of the auxiliary information
- **b**: Vector of estimated regression coefficients
- **b0**: By default FALSE. The intercept of the regression model

Value

The function returns a vector of total population estimates for each variable of interest, its estimated standard error and its estimated coefficient of variation.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

- E.Beta

Examples

```
# Example 1: Linear models involving continuous auxiliary information
# Draws a simple random sample without replacement
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
```

```
# common mean model
```
estima<-data.frame(Income, Employees, Taxes)
x <- rep(1,n)
model <- E.Beta(N, n, estima, x, ck=1,b0=FALSE)
b <- t(as.matrix(model[1,,]))
tx <- c(N)
GREG.SI(N,n,estima,x,tx, b, b0=FALSE)

################ common ratio model

estima<-data.frame(Income)
x <- data.frame(Employees)
model <- E.Beta(N, n, estima, x, ck=x,b0=FALSE)
b <- t(as.matrix(model[1,,]))
tx <- sum(Lucy$Employees)
GREG.SI(N,n,estima,x,tx, b, b0=FALSE)

################ Simple regression model without intercept

estima<-data.frame(Income, Employees)
x <- data.frame(Taxes)
model <- E.Beta(N, n, estima, x, ck=1,b0=FALSE)
b <- t(as.matrix(model[1,,]))
tx <- sum(Lucy$Taxes)
GREG.SI(N,n,estima,x,tx, b, b0=FALSE)

################ Multiple regression model without intercept

estima<-data.frame(Income)
x <- data.frame(Employees, Taxes)
model <- E.Beta(N, n, estima, x, ck=1, b0=FALSE)
b <- as.matrix(model[1,,])
tx <- c(sum(Lucy$Employees), sum(Lucy$Taxes))
GREG.SI(N,n,estima,x,tx, b, b0=FALSE)

################ Simple regression model with intercept

estima<-data.frame(Income, Employees)
x <- data.frame(Taxes)
model <- E.Beta(N, n, estima, x, ck=1,b0=TRUE)
b <- as.matrix(model[1,,])
tx <- c(N, sum(Lucy$Taxes))
GREG.SI(N,n,estima,x,tx, b, b0=TRUE)

################ Multiple regression model with intercept

estima<-data.frame(Income)
x <- data.frame(Employees, Taxes)
model <- E.Beta(N, n, estima, x, ck=1,b0=TRUE)
b <- as.matrix(model[1,,])
tx <- c(N, sum(Lucy$Employees), sum(Lucy$Taxes))
GREG.SI(N,n,estima,x,tx, b, b0=TRUE)
# Example 2: Linear models with discrete auxiliary information

# Draws a simple random sample without replacement
data(Lucy)

N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)

# The auxiliary information is discrete type
Doma<-Domains(Level)

########### Poststratified common mean model

estima<-data.frame(Income, Employees, Taxes)
model <- E.Beta(N, n, estima, Doma, ck=1,b0=FALSE)
b <- t(as.matrix(model[1,,]))
tx <- colSums(Domains(Lucy$Level))
GREG.SI(N,n,estima,Doma,tx, b, b0=FALSE)

########### Poststratified common ratio model

estima<-data.frame(Income, Employees)
x <- Doma*Taxes
model <- E.Beta(N, n, estima, x ,ck=1,b0=FALSE)
b <- as.matrix(model[1,,])
tx <- colSums(Domains(Lucy$Level)*Lucy$Taxes)
GREG.SI(N,n,estima,x,tx, b, b0=FALSE)

######################################################################
## Example 3: Domains estimation trough the postestratified estimator
######################################################################

# Draws a simple random sample without replacement
data(Lucy)

N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)

# The auxiliary information is discrete type
Doma<-Domains(Level)
HH

The Hansen-Hurwitz Estimator

Description
Computes the Hansen-Hurwitz Estimator estimator of the population total for several variables of interest

Usage

HH(y, pk)

Arguments

y Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample

pk A vector containing selection probabilities for each unit in the selected sample
Details

The Hansen-Hurwitz estimator is given by

\[ \sum_{i=1}^{m} \frac{y_i}{p_i} \]

where \( y_i \) is the value of the variables of interest for the \( i \)th unit, and \( p_i \) is its corresponding selection probability. This estimator is restricted to with replacement sampling designs.

Value

The function returns a vector of total population estimates for each variable of interest, its estimated standard error and its estimated coefficient of variation.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

HT

Examples

```
# Example 1

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")

# Vectors y1 and y2 give the values of the variables of interest
y1<-c(32, 34, 46, 89, 35)
y2<-c(1,1,1,0,0)
y3<-cbind(y1,y2)

# The population size is N=5
N <- length(U)

# The sample size is m=2
m <- 2

# pk is the probability of selection of every single unit
pk <- c(0.35, 0.225, 0.175, 0.125, 0.125)

# Selection of a random sample with replacement
sam <- sample(5, 2, replace=TRUE, prob=pk)

# The selected sample is
U[sam]

# The values of the variables of interest for the units in the sample
y1[sam]
```
The Hansen-Hurwitz estimator

HH(y1[sam], pk[sam])
HH(y2[sam], pk[sam])
HH(y3[sam], pk[sam])

Example 2

Uses the Lucy data to draw a simple random sample with replacement

data(Lucy)
attach(Lucy)

N <- dim(Lucy)[1]
m <- 400
sam <- sample(N, m, replace=TRUE)

pk <- rep(1/N, m)

data <- Lucy[sam,]

estima <- data.frame(Income, Employees, Taxes)

HH(estima, pk)

Example 3 HH is unbiased for with replacement sampling designs

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")

y <- c(32, 34, 46, 89, 35)

N <- length(U)
m <- 2

pk <- c(0.35, 0.225, 0.175, 0.125, 0.125)

p <- p.WR(N, m, pk)

sum(p)

Ind <- nk(N, m)

Qy <- SupportWR(N, m, ID=y)

Qy
# The support with the values of the elements
Qp <- SupportWR(N,m, ID=pk)

# The HT estimates for every single sample in the support
HH1 <- HH(Qy[1,], Qp[1,])[1,]
HH2 <- HH(Qy[2,], Qp[2,])[1,]
HH3 <- HH(Qy[3,], Qp[3,])[1,]
HH4 <- HH(Qy[4,], Qp[4,])[1,]
HH5 <- HH(Qy[5,], Qp[5,])[1,]
HH6 <- HH(Qy[6,], Qp[6,])[1,]
HH7 <- HH(Qy[7,], Qp[7,])[1,]
HH8 <- HH(Qy[8,], Qp[8,])[1,]
HH9 <- HH(Qy[9,], Qp[9,])[1,]
HH10 <- HH(Qy[10,], Qp[10,])[1,]
HH11 <- HH(Qy[11,], Qp[11,])[1,]
HH12 <- HH(Qy[12,], Qp[12,])[1,]
HH13 <- HH(Qy[13,], Qp[13,])[1,]
HH14 <- HH(Qy[14,], Qp[14,])[1,]
HH15 <- HH(Qy[15,], Qp[15,])[1,]

# The HT estimates arranged in a vector
Est <- c(HH1, HH2, HH3, HH4, HH5, HH6, HH7, HH8, HH9, HH10, HH11, HH12, HH13,
HH14, HH15)

# The HT is actually design-unbiased
Est

data.frame(Ind, Est, p)
sum(Est*p)
sum(y)

---

**HT**

The Horvitz-Thompson Estimator

**Description**

Computes the Horvitz-Thompson estimator of the population total for several variables of interest

**Usage**

HT(y, Pik)

**Arguments**

- **y** Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
- **Pik** A vector containing the inclusion probabilities for each unit in the selected sample
Details

The Horvitz-Thompson estimator is given by

\[ \sum_{k \in U} \frac{y_k}{\pi_k} \]

where \( y_k \) is the value of the variables of interest for the \( k \)th unit, and \( \pi_k \) its corresponding inclusion probability. This estimator could be used for without replacement designs as well as for with replacement designs.

Value

The function returns a vector of total population estimates for each variable of interest.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

HH

Examples

#######
## Example 1
#######
# Uses the Lucy data to draw a simple random sample without replacement
data(Lucy)
attach(Lucy)

N <- dim(Lucy)[1]
n <- 400
sam <- sample(N,n)
# The vector of inclusion probabilities for each unit in the sample
pik <- rep(n/N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
HT(estima, pik)
# Uses the Lucy data to draw a simple random sample with replacement
\begin{verbatim}
data(Lucy)
N <- dim(Lucy)[1]
m <- 400
sam <- sample(N,m,replace=TRUE)
# The vector of selection probabilities of units in the sample
pk <- rep(1/N,m)
# Computation of the inclusion probabilities
pik <- 1-(1-pk)^m
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
HT(estima, pik)
\end{verbatim}

## Example 3

# Without replacement sampling
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector y1 and y2 are the values of the variables of interest
y1<-c(32, 34, 46, 89, 35)
y2<-c(1,1,1,0,0)
y3<-cbind(y1,y2)
# The population size is N=5
N <- length(U)
# The sample size is n=2
n <- 2
# The sample membership matrix for fixed size without replacement sampling designs
Ind <- Ik(N,n)
# p is the probability of selection of every possible sample
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)
# Computation of the inclusion probabilities
inclusion <- Pik(p, Ind)
# Selection of a random sample
sam <- sample(5,2)
# The selected sample
U[sam]
# The inclusion probabilities for these two units
inclusion[sam]
# The values of the variables of interest for the units in the sample
y1[sam]
y2[sam]
y3[sam,]
# The Horvitz-Thompson estimator
# Following Example 3... With replacement sampling
# The population size is N=5
N <- length(U)
# The sample size is m=2
m <- 2
# pk is the probability of selection of every single unit
pk <- c(0.9, 0.025, 0.025, 0.025, 0.025)
# Computation of the inclusion probabilities
pik <- 1-(1-pk)^m
# Selection of a random sample with replacement
sam <- sample(5, 2, replace=TRUE, prob=pk)
# The selected sample
U[sam]
# The inclusion probabilities for these two units
inclusion[sam]
# The values of the variables of interest for the units in the sample
y1[sam]
y2[sam]
y3[sam,]
# The Horvitz-Thompson estimator
HT(y1[sam], inclusion[sam])
HT(y2[sam], inclusion[sam])
HT(y3[sam,], inclusion[sam])

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector y1 and y2 are the values of the variables of interest
y <- c(32, 34, 46, 89, 35)
# The population size is N=5
N <- length(U)
# The sample size is n=2
n <- 2
# The sample membership matrix for fixed size without replacement sampling designs
Ind <- Ik(N, n)
Ind
# p is the probability of selection of every possible sample
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)
sum(p)
# Computation of the inclusion probabilities
inclusion <- Pik(p, Ind)
inclusion
The support with the values of the elements
Qy <- Support(N, n, ID=y)
Qy

The HT estimates for every single sample in the support
HT1 <- HT(y[Ind[1,]==1], inclusion[Ind[1,]==1])
HT2 <- HT(y[Ind[2,]==1], inclusion[Ind[2,]==1])
HT3 <- HT(y[Ind[3,]==1], inclusion[Ind[3,]==1])
HT4 <- HT(y[Ind[4,]==1], inclusion[Ind[4,]==1])
HT5 <- HT(y[Ind[5,]==1], inclusion[Ind[5,]==1])
HT6 <- HT(y[Ind[6,]==1], inclusion[Ind[6,]==1])
HT7 <- HT(y[Ind[7,]==1], inclusion[Ind[7,]==1])
HT8 <- HT(y[Ind[8,]==1], inclusion[Ind[8,]==1])
HT9 <- HT(y[Ind[9,]==1], inclusion[Ind[9,]==1])
HT10 <- HT(y[Ind[10,]==1], inclusion[Ind[10,]==1])

The HT estimates arranged in a vector
Est <- c(HT1, HT2, HT3, HT4, HT5, HT6, HT7, HT8, HT9, HT10)
Est

The HT is actually design-unbiased
data.frame(Ind, Est, p)
sum(Est*p)
sum(y)

# Example 6 HT is unbiased for without replacement sampling designs
## Random sample size

Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
Vector y contains the values of the variables of interest
y <- c(32, 34, 46, 89, 35)
The population size is N=5
N <- length(U)
The sample membership matrix for random size without replacement sampling designs
Ind <- IkRS(N)
Ind
# p is the probability of selection of every possible sample
p <- c(0.59049, 0.06561, 0.06561, 0.06561, 0.06561, 0.06561, 0.00729, 0.00729,
      0.00729, 0.00729, 0.00729, 0.00729, 0.00729, 0.00729, 0.00729, 0.00729,
      0.0081, 0.00081, 0.00081, 0.00081, 0.00081, 0.00081, 0.00081, 0.00081,
      0.00009, 0.00009, 0.00009, 0.00009, 0.00009)
sum(p)
# Computation of the inclusion probabilities
inclusion <- Pik(p, Ind)
inclusion
sum(inclusion)

The support with the values of the elements
Qy <- SupportRS(N, ID=y)
Qy

The HT estimates for every single sample in the support
HT1 <- HT(y[Ind[1,]==1], inclusion[Ind[1,]==1])
HT2 <- HT(y[Ind[2,]==1], inclusion[Ind[2,]==1])
HT3<- HT(y[Ind[3,]==1], inclusion[Ind[3,]==1])
HT4<- HT(y[Ind[4,]==1], inclusion[Ind[4,]==1])
HT5<- HT(y[Ind[5,]==1], inclusion[Ind[5,]==1])
HT6<- HT(y[Ind[6,]==1], inclusion[Ind[6,]==1])
HT7<- HT(y[Ind[7,]==1], inclusion[Ind[7,]==1])
HT8<- HT(y[Ind[8,]==1], inclusion[Ind[8,]==1])
HT9<- HT(y[Ind[9,]==1], inclusion[Ind[9,]==1])
HT10<- HT(y[Ind[10,]==1], inclusion[Ind[10,]==1])
HT11<- HT(y[Ind[11,]==1], inclusion[Ind[11,]==1])
HT12<- HT(y[Ind[12,]==1], inclusion[Ind[12,]==1])
HT13<- HT(y[Ind[13,]==1], inclusion[Ind[13,]==1])
HT14<- HT(y[Ind[14,]==1], inclusion[Ind[14,]==1])
HT15<- HT(y[Ind[15,]==1], inclusion[Ind[15,]==1])
HT16<- HT(y[Ind[16,]==1], inclusion[Ind[16,]==1])
HT17<- HT(y[Ind[17,]==1], inclusion[Ind[17,]==1])
HT18<- HT(y[Ind[18,]==1], inclusion[Ind[18,]==1])
HT19<- HT(y[Ind[19,]==1], inclusion[Ind[19,]==1])
HT20<- HT(y[Ind[20,]==1], inclusion[Ind[20,]==1])
HT21<- HT(y[Ind[21,]==1], inclusion[Ind[21,]==1])
HT22<- HT(y[Ind[22,]==1], inclusion[Ind[22,]==1])
HT23<- HT(y[Ind[23,]==1], inclusion[Ind[23,]==1])
HT24<- HT(y[Ind[24,]==1], inclusion[Ind[24,]==1])
HT25<- HT(y[Ind[25,]==1], inclusion[Ind[25,]==1])
HT26<- HT(y[Ind[26,]==1], inclusion[Ind[26,]==1])
HT27<- HT(y[Ind[27,]==1], inclusion[Ind[27,]==1])
HT28<- HT(y[Ind[28,]==1], inclusion[Ind[28,]==1])
HT29<- HT(y[Ind[29,]==1], inclusion[Ind[29,]==1])
HT30<- HT(y[Ind[30,]==1], inclusion[Ind[30,]==1])
HT31<- HT(y[Ind[31,]==1], inclusion[Ind[31,]==1])
HT32<- HT(y[Ind[32,]==1], inclusion[Ind[32,]==1])

# The HT estimates arranged in a vector
Est <- c(HT1, HT2, HT3, HT4, HT5, HT6, HT7, HT8, HT9, HT10, HT11, HT12, HT13, 
         HT14, HT15, HT16, HT17, HT18, HT19, HT20, HT21, HT22, HT23, HT24, HT25, HT26, 
         HT27, HT28, HT29, HT30, HT31, HT32)

# The HT is actually design-unbiased
data.frame(Ind, Est, p)
sum(Est*p)
sum(y)

################################################################
## Example 7 HT is unbiased for with replacement sampling designs
################################################################

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector y1 and y2 are the values of the variables of interest
y<-(c(32, 34, 46, 89, 35)
# The population size is N=5
N <- length(U)
# The sample size is m=2
m <- 2
# pk is the probability of selection of every single unit
pk <- c(0.35, 0.225, 0.175, 0.125, 0.125)
# p is the probability of selection of every possible sample
p <- p WR(N, m, pk)
p
sum(p)
# The sample membership matrix for random size without replacement sampling designs
Ind <- Ik WR(N, m)
Ind
# The support with the values of the elements
Qy <- Support WR(N, m, ID = y)
Qy
# Computation of the inclusion probabilities
pik <- 1 - (1 - pk)^m
pik
# The HT estimates for every single sample in the support
HT1 <- HT(y[Ind[1,]==1], pik[Ind[1,]==1])
HT2 <- HT(y[Ind[2,]==1], pik[Ind[2,]==1])
HT3 <- HT(y[Ind[3,]==1], pik[Ind[3,]==1])
HT4 <- HT(y[Ind[4,]==1], pik[Ind[4,]==1])
HT5 <- HT(y[Ind[5,]==1], pik[Ind[5,]==1])
HT6 <- HT(y[Ind[6,]==1], pik[Ind[6,]==1])
HT7 <- HT(y[Ind[7,]==1], pik[Ind[7,]==1])
HT8 <- HT(y[Ind[8,]==1], pik[Ind[8,]==1])
HT9 <- HT(y[Ind[9,]==1], pik[Ind[9,]==1])
HT10 <- HT(y[Ind[10,]==1], pik[Ind[10,]==1])
HT11 <- HT(y[Ind[11,]==1], pik[Ind[11,]==1])
HT12 <- HT(y[Ind[12,]==1], pik[Ind[12,]==1])
HT13 <- HT(y[Ind[13,]==1], pik[Ind[13,]==1])
HT14 <- HT(y[Ind[14,]==1], pik[Ind[14,]==1])
HT15 <- HT(y[Ind[15,]==1], pik[Ind[15,]==1])
# The HT estimates arranged in a vector
Est <- c(HT1, HT2, HT3, HT4, HT5, HT6, HT7, HT8, HT9, HT10, HT11, HT12, HT13,
        HT14, HT15)
Est
# The HT is actually design-unbiased
data.frame(Ind, Est, p)
sum(Est*p)
sum(y)

---

**Ik**  
*Sample Membership Indicator*

**Description**

Creates a matrix of values (0, if the unit belongs to a specified sample and 1, otherwise) for every possible sample under fixed sample size designs without replacement

**Usage**

Ik(N, n)
Arguments

N  Population size
n  Sample size

Value

The function returns a matrix of $\text{binom}(N)(n)$ rows and $N$ columns. The $k$th column corresponds to the sample membership indicator, of the $k$th unit, to a possible sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

Support, Pik

Examples

```
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
n <- 2
# The sample membership matrix for fixed size without replacement sampling designs
Ik(N,n)
# The first unit, Yves, belongs to the first four possible samples
```

Description

Creates a matrix of values (0, if the unit belongs to a specified sample and 1, otherwise) for every possible sample under random sample size designs without replacement

Usage

```
IkRS(N)
```

Arguments

N  Population size
Value

The function returns a matrix of $2^N$ rows and $N$ columns. The $k$th column corresponds to the sample membership indicator, of the $k$th unit, to a possible sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

SupportRS,Pik

Examples

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
n <- 3
# The sample membership matrix for fixed size without replacement sampling designs
IkRS(N)
# The first sample is a null one and the last sample is a census

---

**IkWR**

*Sample Membership Indicator for with Replacements sampling designs*

Description

Creates a matrix of values (1, if the unit belongs to a specified sample and 0, otherwise) for every possible sample under fixed sample size designs without replacement.

Usage

IkWR(N, m)

Arguments

<table>
<thead>
<tr>
<th>N</th>
<th>Population size</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>Sample size</td>
</tr>
</tbody>
</table>
Value

The function returns a matrix of \( \text{binom}(N + m - 1)(m) \) rows and \( N \) columns. The \( k \)th column corresponds to the sample membership indicator, of the \( k \)th unit, to a possible sample. It returns a value of 1, even if the element is selected more than once in a with replacement sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

nk, Support, Pik

Examples

# Vector U contains the label of a population of size N=5
U <- c(“Yves”, “Ken”, “Erik”, “Sharon”, “Leslie”)
N <- length(U)
m <- 2
# The sample membership matrix for fixed size without replacement sampling designs
IkWR(N, m)

IPFP

Iterative Proportional Fitting Procedure

Description

Adjustment of a table on the margins

Usage

IPFP(Table, Col.knw, Row.knw, tol=0.0001)

Arguments

Table | A contingency table
Col.knw | A vector containing the true totals of the columns
Row.knw | A vector containing the true totals of the Rows
tol | The control value, by default equal to 0.0001
Details

Adjust a contingency table on the known margins of the population with the Raking Ratio method.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


Examples

#######
## Example 1
#######
# Some example of Ardilly and Tille
Table <- matrix(c(80,90,10,170,80,80,150,210,130),3,3)
rownames(Table) <- c("a1", "a2","a3")
colnames(Table) <- c("b1", "b2","b3")
# The table with labels
Table
# The known and true margins
Col.knw <- c(150,300,550)
Row.knw <- c(430,360,210)
# The adjusted table
IPFP(Table,Col.knw,Row.knw,tol=0.0001)

#######
## Example 2
#######
# Draws a simple random sample
data(Lucy)
attach(Lucy)
N<-dim(Lucy)[1]
n<-400
sam<sample(N,n)
data<-Lucy[sam,]
attach(data)
dim(data)
# Two domains of interest
Doma1<-Domains(Level)
Doma2<-Domains(SPAM)
# Cross tabulate of domains
SPAM.no<-Doma2[,1]*Doma1
SPAM.yes<-Doma2[,2]*Doma1
# Estimation
E.SI(N,n,Doma1)
E.SI(N,n,Dom2)
est1 <- E.SI(N,n,SPAM.no)[,2:4]
est2 <- E.SI(N,n,SPAM.yes)[,2:4]
est1;est2
# The contingency table estimated from above
Table <- cbind(est1[,1],est2[,1])
rownames(Table) <- c("Big", "Medium","Small")
colnames(Table) <- c("SPAM.no", "SPAM.yes")
# The known and true margins
Col.knw <- colSums(Domains(Lucy$SPAM))
Row.knw <- colSums(Domains(Lucy$Level))
# The adjusted table
IPFP(Table,Col.knw,Row.knw,tol=0.0001)

---

**Lucy**

*Some Business Population Database*

**Description**

This data set corresponds to a random sample of BigLucy. It contains some financial variables of 2396 industrial companies of a city in a particular fiscal year.

**Usage**

data(Lucy)

**Format**

- **ID** The identifier of the company. It corresponds to an alphanumeric sequence (two letters and three digits).
- **Ubication** The address of the principal office of the company in the city.
- **Level** The industrial companies are discriminated according to the Taxes declared. There are small, medium, and big companies.
- **Zone** The city is divided by geographical zones. A company is classified in a particular zone according to its address.
- **Income** The total amount of a company’s earnings (or profit) in the previous fiscal year. It is calculated by taking revenues and adjusting for the cost of doing business.
- **Employees** The total number of persons working for the company in the previous fiscal year.
- **Taxes** The total amount of a company’s income tax.
- **SPAM** Indicates if the company uses the Internet and WEBmail options in order to make self-propaganda.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezrezo@gmail.com>
References


See Also

*BigLucy, BigCity*

Examples

data(Lucy)
attach(Lucy)

# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
# The population totals
colSums(estima)
# Some parameters of interest
table(SPAM, Level)
xtabs(Income ~ Level + SPAM)
# Correlations among characteristics of interest
cor(estima)
# Some useful histograms
hist(Income)
hist(Taxes)
hist(Employees)
# Some useful plots
boxplot(Income ~ Level)
barplot(table(Level))
pie(table(SPAM))

---

nk  

**Sample Selection Indicator for With Replacement Sampling Designs**

Description

The function returns a matrix of \(\text{binom}(N + m - 1)(m)\) rows and \(N\) columns. Creates a matrix of values (0, if the unit does not belong to a specified sample, 1, if the unit is selected once in the sample; 2, if the unit is selected twice in the sample, etc.) for every possible sample under fixed sample size designs with replacement.

Usage

\[
nk(N, m)
\]

Arguments

- \(N\)  
  Population size
- \(m\)  
  Sample size
Value

The function returns a matrix of \( \binom{N + m - 1}{m} \) rows and \( N \) columns. The \( k \)th column corresponds to the sample selection indicator, of the \( k \)th unit, to a possible sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezrez@gmail.com>

References


See Also

SupportWR,Pik

Examples

```r
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
m <- 2
# The sample membership matrix for fixed size without replacement sampling designs
nk(N,m)
```

---

**OrderWR**

**Pseudo-Support for Fixed Size With Replacement Sampling Designs**

Description

Creates a matrix containing every possible ordered sample under fixed sample size with replacement designs

Usage

```
OrderWR(N, m, ID=FALSE)
```

Arguments

- \( N \): Population size
- \( m \): Sample size
- \( ID \): By default FALSE, a vector of values (numeric or string) identifying each unit in the population
Details

The number of samples in a with replacement support is not equal to the number of ordered samples induced by a with replacement sampling design.

Value

The function returns a matrix of $N^m$ rows and $m$ columns. Each row of this matrix corresponds to a possible ordered sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>. The author acknowledges to Hanwen Zhang for valuable suggestions.

References

Tille, Y. (2006), Sampling Algorithms. Springer
Gutierrez, H. A. (2009), Estrategias de muestreo: Diseno de encuestas y estimacion de parametros. Editorial Universidad Santo Tomas

See Also

SupportWR, Support

Examples

```r
# Vector U contains the label of a population
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
# Under this context, there are five (5) possible ordered samples
OrderWR(N,1)
# The same output, but labeled
OrderWR(N,1,ID=U)
# y is the variable of interest
y<-c(32,34,46,89,35)
OrderWR(N,1,ID=y)

# If the sample size is m=2, there are (25) possible ordered samples
OrderWR(N,2)
# The same output, but labeled
OrderWR(N,2,ID=U)
# y is the variable of interest
y<-c(32,34,46,89,35)
OrderWR(N,2,ID=y)

# Note that the number of ordered samples is not equal to the number of
# samples in a well defined with-replacement support
OrderWR(N,2)
SupportWR(N,2)

OrderWR(N,4)
```
Support\(WR(N,4)\)

---

**Description**

Computes the selection probability (sampling design) of each with replacement sample

**Usage**

\[ p.WR(N, m, pk) \]

**Arguments**

- \(N\): Population size
- \(m\): Sample size
- \(pk\): A vector containing selection probabilities for each unit in the population

**Details**

Every with replacement sampling design is a particular case of a multinomial distribution.

\[
p(S = s) = \frac{m!}{n_1!n_2! \cdots n_N!} \prod_{i=1}^{N} p_{k}^{n_{k}}
\]

where \(n_k\) is the number of times that the \(k\)-th unit is selected in a sample.

**Value**

The function returns a vector of selection probabilities for every with-replacement sample.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**

Examples

##############
## Example 1
##############
# With replacement simple random sampling
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector pk is the selection probability of the units in the finite population
pk <- c(0.2, 0.2, 0.2, 0.2, 0.2)
sum(pk)
N <- length(pk)
m <- 3
# The sampling design
p <- p.WR(N, m, pk)
p
sum(p)

##############
## Example 2
##############
# With replacement PPS random sampling
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector x is the auxiliary information and y is the variables of interest
x <- c(32, 34, 46, 89, 35)
y <- c(52, 60, 75, 100, 50)
# Vector pk is the selection probability of the units in the finite population
pk <- x/sum(x)
sum(pk)
N <- length(pk)
m <- 3
# The sampling design
p <- p.WR(N, m, pk)
p
sum(p)

---

Pik

Inclusion Probabilities for Fixed Size Without Replacement Sampling Designs

Description

Computes the first-order inclusion probability of each unit in the population given a fixed sample size design

Usage

Pik(p, Ind)
Arguments

- **p**: A vector containing the selection probabilities of a fixed size without replacement sampling design. The sum of the values of this vector must be one.
- **Ind**: A sample membership indicator matrix.

Details

The inclusion probability of the \( k \)th unit is defined as the probability that this unit will be included in a sample, it is denoted by \( \pi_k \) and obtained from a given sampling design as follows:

\[
\pi_k = \sum_{s \ni k} p(s)
\]

Value

The function returns a vector of inclusion probabilities for each unit in the finite population.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

HT

Examples

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
# The sample size is n=2
n <- 2
# The sample membership matrix for fixed size without replacement sampling designs
Ind <- Ik(N,n)
# p is the probability of selection of every sample.
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)
# Note that the sum of the elements of this vector is one
sum(p)
# Computation of the inclusion probabilities
inclusion <- Pik(p, Ind)
inclusion
# The sum of inclusion probabilities is equal to the sample size n=2
sum(inclusion)
PikHol  

*Optimal Inclusion Probabilities Under Multi-purpose Sampling*

**Description**

Computes the population vector of optimal inclusion probabilities under the Holmberg’s Approach.

**Usage**

\texttt{PikHol(n, sigma, e, Pi)}

**Arguments**

- \texttt{n} Vector of optimal sample sizes for each of the characteristics of interest.
- \texttt{sigma} A matrix containing the size measures for each characteristics of interest.
- \texttt{e} Maximum allowed error under the ANOREL approach.
- \texttt{Pi} Matrix of first order inclusion probabilities. By default, these probabilities are proportional to each sigma.

**Details**

Assuming that all of the characteristic of interest are equally important, the Holmberg’s sampling design yields the following inclusion probabilities

\[
\pi_{(\text{opt})k} = \frac{n^* \sqrt{a_{qk}}}{\sum_{k \in U} \sqrt{a_{qk}}}
\]

where

\[
n^* \geq \frac{\left(\sum_{k \in U} \sqrt{a_{qk}}\right)^2}{(1 + c)Q + \sum_{k \in U} a_{qk}}
\]

and

\[
a_{qk} = \sum_{q=1}^{Q} \frac{\sigma_{qk}^2}{\sum_{k \in U} \left(\frac{1}{\pi_{qk}} - 1\right) \sigma_{qk}^2}
\]

Note that \(\sigma_{qk}^2\) is a size measure associated with the k-th element in the q-th characteristic of interest.

**Value**

The function returns a vector of inclusion probabilities.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>
References

Gutierrez, H. A. (2009), Estrategias de muestreo: Diseño de encuestas y estimación de parámetros. Editorial Universidad Santo Tomas

Examples

# Uses the Lucy data to draw an optimal sample
# in a multipurpose survey context
data(Lucy)
attach(Lucy)
# Different sample sizes for two characteristics of interest: Employees and Taxes
N <- dim(Lucy)[1]
n <- c(350,400)
# The size measure is the same for both characteristics of interest,
# but the relationship in between is different
sigy1 <- sqrt(Income^1)
sigy2 <- sqrt(Income^2)
# The matrix containing the size measures for each characteristic of interest
sigma<-cbind(sigy1,sigy2)
# The vector of optimal inclusion probabilities under the Holmberg's approach
Piks<-PikHol(n,sigma,0.03)
# The optimal sample size is given by the sum of piks
n=round(sum(Piks))
# Performing the S.piPS function in order to select the optimal sample of size n
res<-S.piPS(n,Piks)
sam <- res[,1]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# Pik.s is the vector of inclusion probability of every single unit
# in the selected sample
Pik.s <- res[,2]
# The variables of interest are: Income, Employees and Taxes
# This information is stored in a data frame called estima
estima <- data.frame(Income, Employees, Taxes)
E.piPS(estima,Pik.s)

# We can define our own first inclusion probabilities
data(Lucy)
attach(Lucy)

N <- dim(Lucy)[1]
n <- c(350, 400)

sigy1 <- sqrt(Income^1)
sigy2 <- sqrt(Income^2)
sigma <- cbind(sigy1, sigy2)
pikas <- cbind(rep(400/N, N), rep(400/N, N))

Piks <- PikHol(n, sigma, 0.03, pikas)

n = round(sum(Piks))

res <- S.piPS(n, Piks)
sam <- res[, 1]

data <- Lucy[sam,]
attach(data)
names(data)

Pik.s <- res[, 2]
estima <- data.frame(Income, Employees, Taxes)
E.piPS(estima, Pik.s)

---

**Pikl**

*Second Order Inclusion Probabilities for Fixed Size Without Replacement Sampling Designs*

**Description**

Computes the second-order inclusion probabilities of each pair of units in the population given a fixed sample size design

**Usage**

Pikl(N, n, p)

**Arguments**

- **N**: Population size
- **n**: Sample size
- **p**: A vector containing the selection probabilities of a fixed size without replacement sampling design. The sum of the values of this vector must be one
Details

The second-order inclusion probability of the \( k \)/th units is defined as the probability that unit \( k \) and unit \( l \) will be both included in a sample; it is denoted by \( \pi_{kl} \) and obtained from a given sampling design as follows:

\[
\pi_{kl} = \sum_{s \ni k, l} p(s)
\]

Value

The function returns a symmetric matrix of size \( N \times N \) containing the second-order inclusion probabilities for each pair of units in the finite population.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

VarHT, Deltakl, Pik

Examples

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
# The sample size is n=2
n <- 2
# p is the probability of selection of every sample.
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)
# Note that the sum of the elements of this vector is one
sum(p)
# Computation of the second-order inclusion probabilities
Pikl(N, n, p)
Usage

PikPPS(n,x)

Arguments

n 
Integer indicating the sample size

x 
Vector of auxiliary information for each unit in the population

Details

For a given vector of auxiliary information with value \( x_k \) for the \( k \)-th unit and population total \( t_x \), the following expression

\[
\pi_k = n \times \frac{x_k}{t_x}
\]

is not always less than unity. A sequential algorithm must be used in order to ensure that for every unit in the population the inclusion probability gives less or equal to unity.

Value

The function returns a vector of inclusion probabilities of size \( N \). Every element of this vector is a value between zero and one.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

PikHol,E.piPS,S.piPS

Examples

###########
## Example 1
###########
x <- c(30,41,50,170,43,200)
n <- 3
# Two elements yields values bigger than one
n*x/sum(x)
# With this functions, all of the values are between zero and one
PikPPS(n,x)
# The sum is equal to the sample size
sum(PikPPS(n,x))
### Example 2

Vector \( U \) contains the label of a population of size \( N=5 \)

\[
U \leftarrow c("Yves", "Ken", "Erik", "Sharon", "Leslie")
\]

- \( U \) is the auxiliary information

\[
x \leftarrow c(52, 60, 75, 100, 50)
\]

The inclusion probabilities for the population according to a

- \( n=4 \)

\[
pik \leftarrow \text{PikPPS}(4, x)
\]

The selected sample is

\[
\text{sum}(pik)
\]

### Example 3

Uses the Lucy data to compute the vector of inclusion probabilities

- \( \text{pikPS} \) without replacement design

\[
\text{data}(\text{Lucy})
\]

- \( \text{attach}(\text{Lucy}) \)

The sample size

\[
n=400
\]

The selection probability of each unit is proportional to the variable \( \text{Income} \)

\[
pik \leftarrow \text{PikPPS}(n, \text{Income})
\]

The inclusion probabilities of the units in the sample

\[
pik
\]

The sum of the values in \( pik \) is equal to the sample size

\[
\text{sum}(pik)
\]

According to the design some elements must be selected

- \( \text{which}(pik==1) \)

---

**PikSTPPS**

### Inclusion Probabilities in Stratified Proportional to Size Sampling Designs

**Description**

For a given sample size, in each stratum, this function returns a vector of first order inclusion probabilities for an stratified sampling design proportional to an auxiliary variable.

**Usage**

\[
\text{PikSTPPS}(S, x, nh)
\]

**Arguments**

- \( S \) : Vector identifying the membership to the strata of each unit in the population.
- \( x \) : Vector of auxiliary information for each unit in the population.
- \( nh \) : The vector defining the sample size in each stratum.
Details

is not always less than unity. A sequential algorithm must be used in order to ensure that for every unit in the population the inclusion probability gives a proper value; i.e. less or equal to unity.

Value

A vector of inclusion probabilities in a stratified finite population.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro at gmail.com>

References


See Also

PikHol, PikPPS, S.STpiPS

Examples

#######

## Example 1

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# The auxiliary information
x <- c(52, 60, 75, 100, 50)
# Vector Strata contains an indicator variable of stratum membership
Strata <- c("A", "A", "A", "B", "B")
# The sample size in each stratum
nh <- c(2, 2)
# The vector of inclusion probabilities for a stratified piPS sample
# without replacement of size two within each stratum
Pik <- PikSTPPS(Strata, x, nh)

# Some checks
sum(Pik)
sum(nh)

#######

## Example 2

# Uses the Lucy data to compute the vector of inclusion probabilities
# for a stratified random sample according to a piPS design in each stratum

data(Lucy)
attach(Lucy)
# Level is the stratifying variable
summary(Level)

# Defines the size of each stratum
N1<-summary(Level)[[1]]
N2<-summary(Level)[[2]]
N3<-summary(Level)[[3]]
N1;N2;N3

# Defines the sample size at each stratum
n1<70
n2<100
n3<200
nh<-c(n1,n2,n3)
hh

# Computes the inclusion probabilities for the stratified population
S <- Level
x <- Employees
Pik <- PikSTPPS(S, x, nh)

# Some checks
sum(Pik)
sum(hh)

---

**S.BE**

*Bernoulli Sampling Without Replacement*

**Description**

Draws a Bernoulli sample without replacement of expected size $n$ from a population of size $N$.

**Usage**

S.BE(N, prob)

**Arguments**

- **N**
  - Population size
- **prob**
  - Inclusion probability for each unit in the population

**Details**

The selected sample is drawn according to a sequential procedure algorithm based on an uniform distribution. The Bernoulli sampling design is not a fixed sample size one.
Value
The function returns a vector of size $N$. Each element of this vector indicates if the unit was selected. Then, if the value of this vector for unit $k$ is zero, the unit $k$ was not selected in the sample; otherwise, the unit was selected in the sample.

Author(s)
Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References

See Also
E.BE

Examples

```
### Example 1
###
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Draws a Bernoulli sample without replacement of expected size n=3
# The inclusion probability is 0.6 for each unit in the population
sam <- S.BE(5,0.6)
sam
# The selected sample is
U[sam]

### Example 2
###
# Uses the Lucy data to draw a Bernoulli sample

data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
# The population size is 2396. If the expected sample size is 400
# then, the inclusion probability must be 400/2396=0.1669
sam <- S.BE(N,0.01669)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)
```
Description

Draws a probability proportional to size sample without replacement of size $n$ from a population of size $N$.

Usage

\texttt{S.piPS(n, x, e)}

Arguments

- $x$: Vector of auxiliary information for each unit in the population
- $n$: Sample size
- $e$: By default, a vector of size $N$ of independent random numbers drawn from the $\text{Uniform}(0,1)$

Details

The selected sample is drawn according to the Sunter method (sequential-list procedure)

Value

The function returns a matrix of $m$ rows and two columns. Each element of the first column indicates the unit that was selected. Each element of the second column indicates the selection probability of this unit

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

\texttt{E.piPS}
Examples

#######
## Example 1
#######
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# The auxiliary information
x <- c(52, 60, 75, 100, 50)
# Draws a piPS sample without replacement of size n=3
res <- S.piPS(3, x)
res
sam <- res[,1]
sam
# The selected sample is
U[sam]

#######
## Example 2
#######
# Uses the Lucy data to draw a random sample of units accordind to a
# piPS without replacement design

data(Lucy)
attach(Lucy)
# The selection probability of each unit is proportional to the variable Income
res <- S.piPS(400, Income)
# The selected sample
sam <- res[,1]
# The inclusion probabilities of the units in the sample
Pik.s <- res[,2]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)

S.PO

Poisson Sampling

Description

Draws a Poisson sample of expected size $n$ from a population of size $N$.

Usage

S.PO(N, Pik)

Arguments

N  Population size
Pik  Vector of inclusion probabilities for each unit in the population
Details

The selected sample is drawn according to a sequential procedure algorithm based on a uniform distribution. The Poisson sampling design is not a fixed sample size one.

Value

The function returns a vector of size $N$. Each element of this vector indicates if the unit was selected. Then, if the value of this vector for unit $k$ is zero, the unit $k$ was not selected in the sample; otherwise, the unit was selected in the sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

E.PO

Examples

```
# Example 1
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
Pik <- c(0.5, 0.2, 1, 0.9, 0.5)
sam <- S.PO(5, Pik)
sam
U[sam]
```

```
# Example 2
N <- dim(Lucy)[1]
n <- 400
Pik <- n*Income/sum(Income)
which(Pik>1)
```
# The selected sample
sam <- S.PO(N,Pik)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)

---

**S.PPS**

*Probability Proportional to Size Sampling With Replacement*

---

**Description**

Draws a probability proportional to size sample with replacement of size \( m \) from a population of size \( N \).

**Usage**

`S.PPS(m, x)`

**Arguments**

- **m**
  - Sample size
- **x**
  - Vector of auxiliary information for each unit in the population

**Details**

The selected sample is drawn according to the cumulative total method (sequential-list procedure).

**Value**

The function returns a matrix of \( m \) rows and two columns. Each element of the first column indicates the unit that was selected. Each element of the second column indicates the selection probability of this unit.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

`E.PPS`
Examples

#######
## Example 1
#######
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# The auxiliary information
x <- c(52, 60, 75, 100, 50)
# Draws a PPS sample with replacement of size m=3
res <- S.PPS(3,x)
sam <- res[,1]
# The selected sample is
U[sam]

#######
## Example 2
#######
# Uses the Lucy data to draw a random sample according to a
# PPS with replacement design
data(Lucy)
attach(Lucy)
# The selection probability of each unit is proportional to the variable Income
m <- 400
res<-S.PPS(400,Income)
# The selected sample
sam <- res[,1]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)

---

**S.SI**

*Simple Random Sampling Without Replacement*

**Description**

Draws a simple random sample without replacement of size $n$ from a population of size $N$.

**Usage**

```r
S.SI(N, n, e=runif(N))
```

**Arguments**

- **N**: Population size
- **n**: Sample size
- **e**: By default, a vector of size $N$ of independent random numbers drawn from the $Uniform(0, 1)$
Details

The selected sample is drawn according to a selection-rejection (list-sequential) algorithm

Value

The function returns a vector of size \( N \). Each element of this vector indicates if the unit was selected. Then, if the value of this vector for unit \( k \) is zero, the unit \( k \) was not selected in the sample; otherwise, the unit was selected in the sample.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezrezo@gmail.com>

References


See Also

E.SI

Examples

#######
## Example 1
#######
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Fixes the random numbers in order to select a sample
# Ideal for teaching purposes in the blackboard
e <- c(0.4938, 0.7044, 0.4585, 0.6747, 0.0640)
# Draws a simple random sample without replacement of size n=3
sam <- S.SI(5,3,e)
sam

# The selected sample is
U[sam]

#######
## Example 2
#######
# Uses the Marco and Lucy data to draw a random sample according to a SI design
data(Marco)
data(Lucy)
N <- dim(Lucy)[1]
n <- 400
S.STpiPS

**Stratified Sampling Applying Without Replacement piPS Design in all Strata**

**Description**

Draws a probability proportional to size simple random sample without replacement of size \( n_h \) in stratum \( h \) of size \( N_h \)

**Usage**

\[
S.STpiPS(S, x, nh)
\]

**Arguments**

- **S** Vector identifying the membership to the strata of each unit in the population
- **x** Vector of auxiliary information for each unit in the population
- **nh** Vector of sample size in each stratum

**Details**

The selected sample is drawn according to the Sunter method (sequential-list procedure) in each stratum

**Value**

The function returns a matrix of \( n = n_1 + \cdots + n_h \) rows and two columns. Each element of the first column indicates the unit that was selected. Each element of the second column indicates the inclusion probability of this unit

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

\[ \text{E.STpiPS} \]
Examples

##### Example 1
### Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
### The auxiliary information
x <- c(52, 60, 75, 100, 50)
### Vector Strata contains an indicator variable of stratum membership
Strata <- c("A", "A", "A", "B", "B")
### Then sample size in each stratum
mh <- c(2, 2)
### Draws a stratified PPS sample with replacement of size n=4
res <- S.STPPS(Strata, x, mh)
### The selected sample
sam <- res[,1]
U[sam]
### The selection probability of each unit selected to be in the sample
pk <- res[,2]
pk

##### Example 2
### Uses the Lucy data to draw a stratified random sample
### according to a piPS design in each stratum

data(Lucy)
attach(Lucy)
### Level is the stratifying variable
summary(Level)

### Defines the size of each stratum
N1<-summary(Level)[[1]]
N2<-summary(Level)[[2]]
N3<-summary(Level)[[3]]
N1;N2;N3

### Defines the sample size at each stratum
n1<-70
n2<-100
n3<-200
nh<-c(n1,n2,n3)
nh
### Draws a stratified sample
S <- Level
x <- Employees

res <- S.STpiPS(S, x, nh)
sam<-res[,1]
### The information about the units in the sample is stored in an object called data
data <- Lucy[sam]
S.STPPS

Stratified Sampling Applying PPS Design in all Strata

Description

Draws a probability proportional to size simple random sample with replacement of size \( m_h \) in stratum \( h \) of size \( N_h \)

Usage

S.STPPS(S,x,mh)

Arguments

- **S**: Vector identifying the membership to the strata of each unit in the population
- **x**: Vector of auxiliary information for each unit in the population
- **mh**: Vector of sample size in each stratum

Details

The selected sample is drawn according to the cumulative total method (sequential-list procedure) in each stratum

Value

The function returns a matrix of \( m = m_1 + \cdots + m_h \) rows and two columns. Each element of the first column indicates the unit that was selected. Each element of the second column indicates the selection probability of this unit

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

E.STPPS
Examples

############
## Example 1
############
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# The auxiliary information
x <- c(52, 60, 75, 100, 50)
# Vector Strata contains an indicator variable of stratum membership
Strata <- c("A", "A", "A", "B", "B")
# Then sample size in each stratum
mh <- c(2, 2)
# Draws a stratified PPS sample with replacement of size n=4
res <- S.STPPS(Strata, x, mh)
# The selected sample
sam <- res[,1]
U[sam]
# The selection probability of each unit selected to be in the sample
pk <- res[,2]
pk

############
## Example 2
############
# Uses the Lucy data to draw a stratified random sample
# according to a PPS design in each stratum

data(Lucy)
attach(Lucy)
# Level is the stratifying variable
summary(Level)
# Defines the sample size at each stratum
m1<-70
m2<-100
m3<-200
mh<-c(m1, m2, m3)
# Draws a stratified sample
res<-S.STPPS(Level, Income, mh)
# The selected sample
sam<-res[,1]
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)
# The selection probability of each unit selected in the sample
pk <- res[,2]
pk
Description

draws a simple random sample without replacement of size \( n_h \) in stratum \( h \) of size \( N_h \).

Usage

S.STSI(S, Nh, nh)

Arguments

S Vector identifying the membership to the strata of each unit in the population
Nh Vector of stratum sizes
nh Vector of sample size in each stratum

Details

The selected sample is drawn according to a selection-rejection (list-sequential) algorithm in each stratum.

Value

The function returns a vector of size \( n = n_1 + \cdots + n_H \). Each element of this vector indicates the unit that was selected.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

E.STSI

Examples

```
# Example 1

# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector Strata contains an indicator variable of stratum membership
Strata <- c("A", "A", "A", "B", "B")
# The stratum sizes
Nh <- c(3,2)
```

nh <- c(2,1)
# Draws a stratified simple random sample without replacement of size n=3
sam <- S.STSI(Strata, Nh, nh)
sam
# The selected sample is
U[sam]

##############
## Example 2
##############
# Uses the Lucy data to draw a stratified random sample
# accordind to a SI design in each stratum
data(Lucy)
attach(Lucy)
# Level is the stratifying variable
summary(Level)
# Defines the size of each stratum
N1<-summary(Level)[[1]]
N2<-summary(Level)[[2]]
N3<-summary(Level)[[3]]
N1;N2;N3
Nh <- c(N1,N2,N3)
# Defines the sample size at each stratum
n1<-70
n2<-100
n3<-200
nh<-c(n1,n2,n3)
# Draws a stratified sample
sam <- S.STSI(Level, Nh, nh)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)

---

**S.SY**

**Systematic Sampling**

**Description**

Draws a Systematic sample of size $n$ from a population of size $N$.

**Usage**

S.SY(N, a)

**Arguments**

- **N**
  - Population size

- **a**
  - Number of groups dividing the population
Details

The selected sample is drawn according to a random start.

Value

The function returns a vector of size \( n \). Each element of this vector indicates the unit that was selected.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>. The author acknowledges to Kristina Stodolova <Kristyna.Stodolova@seznam.cz> for valuable suggestions.

References


See Also

E.SY

Examples

```
# Example 1
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# The population of size N=5 is divided in a=2 groups
# Draws a Systematic sample.
sam <- S.SY(5,2)
sam
# The selected sample is
U[sam]
# There are only two possible samples

# Example 2
# Uses the Lucy data to draw a Systematic sample
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
# The population is divided in 6 groups
# The selected sample
sam <- S.SY(N,6)
```
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)

---

**S.WR**  
**Simple Random Sampling With Replacement**

**Description**

Draws a simple random sample with replacement of size \( m \) from a population of size \( N \)

**Usage**

\[ S.WR(N, m) \]

**Arguments**

- \( N \)  
  Population size
- \( m \)  
  Sample size

**Details**

The selected sample is drawn according to a sequential procedure algorithm based on a binomial distribution

**Value**

The function returns a vector of size \( m \). Each element of this vector indicates the unit that was selected.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**


**See Also**

E.WR
Examples

############
## Example 1
############
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Draws a simple random sample with replacement of size m=3
sam <- S.WR(5, 3)
sam
# The selected sample
U[sam]

############
## Example 2
############
# Uses the Lucy data to draw a random sample of units according to a
# simple random sampling with replacement design
data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
m <- 400
sam <- S.WR(N, m)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
data
dim(data)

---

Support

Sampling Support for Fixed Size Without Replacement Sampling Designs

Description

Creates a matrix containing every possible sample under fixed sample size designs

Usage

Support(N, n, ID=FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Population size</td>
</tr>
<tr>
<td>n</td>
<td>Sample size</td>
</tr>
<tr>
<td>ID</td>
<td>By default FALSE, a vector of values (numeric or string) identifying each unit in the population</td>
</tr>
</tbody>
</table>
SupportRS

Details
A support is defined as the set of samples such that for any sample in the support, all the permutations of the coordinates of the sample are also in the support.

Value
The function returns a matrix of $\binom{N}{n}$ rows and $n$ columns. Each row of this matrix corresponds to a possible sample.

Author(s)
Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References
Tille, Y. (2006), Sampling Algorithms. Springer
Gutierrez, H. A. (2009), Estrategias de muestreo: Diseño de encuestas y estimación de parámetros. Editorial Universidad Santo Tomas

See Also
Ik

Examples
```r
# Vector U contains the label of a population
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
n <- 2
# The support for fixed size without replacement sampling designs
# Under this context, there are ten (10) possibles samples
Support(N,n)
# The same support, but labeled
Support(N,n,ID=U)
# y is the variable of interest
y<-c(32,34,46,89,35)
# The following output is very useful when checking
# the design-unbiasedness of an estimator
Support(N,n,ID=y)
```

---

**SupportRS**

*Sampling Support for Random Size Without Replacement Sampling Designs*

**Description**

Creates a matrix containing every possible sample under random sample size designs.
Usage

SupportRS(N, ID=FALSE)

Arguments

- **N**: Population size
- **ID**: By default FALSE, a vector of values (numeric or string) identifying each unit in the population

Details

A support is defined as the set of samples such that for any sample in the support, all the permutations of the coordinates of the sample are also in the support

Value

The function returns a matrix of \(2^N\) rows and \(N\) columns. Each row of this matrix corresponds to a possible sample

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References

Gutierrez, H. A. (2009), *Estrategias de muestreo: Diseno de encuestas y estimacion de parametros*. Editorial Universidad Santo Tomas

See Also

IkRS

Examples

```r
# Vector U contains the label of a population
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
# The support for fixed size without replacement sampling designs
# Under this context, there are ten (10) possibles samples
SupportRS(N)
# The same support, but labeled
SupportRS(N, ID=U)
# y is the variable of interest
y<-c(32,34,46,89,35)
# The following output is very useful when checking
# the design-unbiasedness of an estimator
SupportRS(N, ID=y)
```
SupportWR

Sampling Support for Fixed Size With Replacement Sampling Designs

Description

Creates a matrix containing every possible sample under fixed sample size with replacement designs

Usage

SupportWR(N, m, ID=FALSE)

Arguments

- N: Population size
- m: Sample size
- ID: By default FALSE, a vector of values (numeric or string) identifying each unit in the population

Details

A support is defined as the set of samples such that, for any sample in the support, all the permutations of the coordinates of the sample are also in the support

Value

The function returns a matrix of $\binom{N + m - 1}{m}$ rows and $m$ columns. Each row of this matrix corresponds to a possible sample

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References

Ortiz, J. E. (2009), Simulacion y metodos estadisticos. Editorial Universidad Santo Tomas.

See Also

Support
Examples

```r
# Vector U contains the label of a population
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
N <- length(U)
m <- 2
# The support for fixed size without replacement sampling designs
# Under this context, there are ten (10) possibles samples
SupportWR(N, m)
# The same support, but labeled
SupportWR(N, m, ID=U)
# y is the variable of interest
y<-c(32,34,46,89,35)
# The following output is very useful when checking
# the design-unbiasedness of an estimator
SupportWR(N, m, ID=y)
```

---

T.SIC

**Computation of Population Totals for Clusters**

**Description**

Computes the population total of the characteristics of interest in clusters. This function is used in order to estimate totals when doing a Pure Cluster Sample.

**Usage**

```
T.SIC(y, Cluster)
```

**Arguments**

- `y` Vector, matrix or data frame containing the recollected information of the variables of interest for every unit in the selected sample
- `Cluster` Vector identifying the membership to the cluster of each unit in the selected sample of clusters

**Value**

The function returns a matrix of clusters totals. The columns of each matrix correspond to the totals of the variables of interest in each cluster

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

**References**

**Examples**

#### Example 1

Vector U contains the label of a population of size $N=5$

$$\text{U} <- \text{c("Yves", "Ken", "Erik", "Sharon", "Leslie")}$$

Vector $y_1$ and $y_2$ are the values of the variables of interest

$$y_1 <- \text{c(32, 34, 46, 89, 35)}$$

$$y_2 <- \text{c(1, 1, 1, 0, 0)}$$

$$y_3 <- \text{cbind}(y_1, y_2)$$

Vector Cluster contains an indicator variable of cluster membership

$$\text{Cluster} <- \text{c("C1", "C2", "C1", "C2", "C1")}$$

Draws a stratified simple random sample without replacement of size $n=3$

$$\text{T.SIC}(y_1, \text{Cluster})$$

$$\text{T.SIC}(y_2, \text{Cluster})$$

$$\text{T.SIC}(y_3, \text{Cluster})$$

#### Example 2 Sampling and estimation in Cluster sampling

Uses Lucy data to draw a cluster sample according to a SI design

$$\text{Zone}$$ is the clustering variable

```r
\text{data(Lucy)}
\text{attach(Lucy)}
\text{summary(Zone)}
```

The population of clusters

$$\text{UI} <- \text{c("A", "B", "C", "D", "E")}$$

$$\text{NI} = \text{length(}UI\text{)}$$

The sample size

$$\text{ni} = 2$$

Draws a simple random sample of two clusters

$$\text{samI} <- \text{S.SI(}\text{NI, ni}\text{)}$$

$$\text{dataI} <- \text{UI[samI]}$$

The information about each unit in the cluster is saved in Lucy1 and Lucy2

```r
\text{data(Lucy)}
\text{Lucy1} <- \text{Lucy[which(Zone==dataI[1]),]}$
\text{Lucy2} <- \text{Lucy[which(Zone==dataI[2]),]}$
\text{LucyI} <- \text{rbind(Lucy1, Lucy2)}$
\text{attach(LucyI)}$
```

The clustering variable is Zone

$$\text{Cluster} <- \text{as.factor(as.integer(}\text{Zone}\text{)}$$

The variables of interest are: Income, Employees and Taxes

This information is stored in a data frame called estima

$$\text{estima} <- \text{data.frame(Income, Employees, Taxes)}$$

$$\text{Ty} <- \text{T.SIC(estima, Cluster)}$$

Estimation of the Population total
Description

Computes the theoretical variance of the Horvitz-Thompson estimator given a without replacement fixed sample size design.

Usage

\texttt{VarHT}(y, N, n, p)

Arguments

- \texttt{y}: Vector containing the recollected information of the characteristic of interest for every unit in the population.
- \texttt{N}: Population size.
- \texttt{n}: Sample size.
- \texttt{p}: A vector containing the selection probabilities of a fixed size without replacement sampling design. The sum of the values of this vector must be one.

Details

The variance of the Horvitz-Thompson estimator, under a given sampling design \( p \), is given by

\[
\text{Var}_p(\hat{t}_{y,\pi}) = \sum_{k \in U} \sum_{l \in U} \Delta_{kl} \frac{y_k}{\pi_k} \frac{y_l}{\pi_l}
\]

Value

The function returns the value of the theoretical variances of the Horvitz-Thompson estimator.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


See Also

\texttt{HT}, \texttt{Deltakl}, \texttt{Pik1}, \texttt{Pik}
Examples

# Without replacement sampling
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector y1 and y2 are the values of the variables of interest
y1 <- c(32, 34, 46, 89, 35)
y2 <- c(1, 1, 1, 0, 0)
# The population size is N=5
N <- length(U)
# The sample size is n=2
n <- 2
# p is the probability of selection of every possible sample
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)

# Calculates the theoretical variance of the HT estimator
VarHT(y1, N, n, p)
VarHT(y2, N, n, p)

VarSYGHT

Two different variance estimators for the Horvitz-Thompson estimator

Description

This function estimates the variance of the Horvitz-Thompson estimator. Two different variance estimators are computed: the original one, due to Horvitz-Thompson and the one due to Sen (1953) and Yates, Grundy (1953). The two approaches yield unbiased estimator under fixed-size sampling schemes.

Usage

VarSYGHT(y, N, n, p)

Arguments

y Vector containing the information of the characteristic of interest for every unit in the population.
N Population size.
n Sample size.
p A vector containing the selection probabilities of a fixed size without replacement sampling design. The sum of the values of this vector must be one.

Details

The function returns two variance estimator for every possible sample within a fixed-size sampling support. The first estimator is due to Horvitz-Thompson and is given by the following expression:

$$\hat{Var}_1(t_y, \pi) = \sum_{k \in U} \sum_{l \in U} \frac{\Delta_{kl}}{\pi_{kl}} \frac{y_k}{\pi_k} \frac{y_l}{\pi_l}$$
The second estimator is due to Sen (1953) and Yates-Grundy (1953). It is given by the following expression:

\[
\hat{\text{Var}}_2(\hat{t}_{y,\pi}) = -\frac{1}{2} \sum_{k \in U} \sum_{l \in U} \Delta_{kl} \pi_{kl} \left( \frac{y_k}{\pi_k} - \frac{y_l}{\pi_l} \right)^2
\]

**Value**

This function returns a data frame of every possible sample in within a sampling support, with its corresponding variance estimates.

**Author(s)**

Hugo Andres Gutierrez Rojas <hagutierrezro at gmail.com>

**References**


**Examples**

```r
# Example 1
# Without replacement sampling
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector y1 and y2 are the values of the variables of interest
y1<-c(32, 34, 46, 89, 35)
y2<-c(1,1,1,0,0)
# The population size is N=5
N <- length(U)
# The sample size is n=2
n <- 2
# p is the probability of selection of every possible sample
p <- c(0.13, 0.2, 0.15, 0.1, 0.15, 0.04, 0.02, 0.06, 0.07, 0.08)

# Calculates the estimated variance for the HT estimator
VarSYGHT(y1, N, n, p)
VarSYGHT(y2, N, n, p)

# Unbiasedness holds in the estimator of the total
sum(y1)
sum(VarSYGHT(y1, N, n, p)*p * VarSYGHT(y1, N, n, p)$Est.HT)
sum(y2)
sum(VarSYGHT(y2, N, n, p)*p * VarSYGHT(y2, N, n, p)$Est.HT)

# Unbiasedness also holds in the two variances
VarHT(y1, N, n, p)
sum(VarSYGHT(y1, N, n, p)*p * VarSYGHT(y1, N, n, p)$Est.Var1)
sum(VarSYGHT(y1, N, n, p)*p * VarSYGHT(y1, N, n, p)$Est.Var2)
```
\begin{verbatim}
VarHT(y2, N, n, p)
sum(VarSYGHT(y2, N, n, p)$p * VarSYGHT(y2, N, n, p)$Est.Var1)
sum(VarSYGHT(y2, N, n, p)$p * VarSYGHT(y2, N, n, p)$Est.Var2)

# Example 2: negative variance estimates
x = c(2.5, 2.0, 1.1, 0.5)
N = 4
n = 2
p = c(0.31, 0.20, 0.14, 0.03, 0.01, 0.31)
VarSYGHT(x, N, n, p)

# Unbiasedness holds in the estimator of the total
sum(x)
sum(VarSYGHT(x, N, n, p)$p * VarSYGHT(x, N, n, p)$Est.HT)

# Unbiasedness also holds in the two variances
VarHT(x, N, n, p)
sum(VarSYGHT(x, N, n, p)$p * VarSYGHT(x, N, n, p)$Est.Var1)
sum(VarSYGHT(x, N, n, p)$p * VarSYGHT(x, N, n, p)$Est.Var2)
\end{verbatim}

---

**Wk**

*The Calibration Weights*

**Description**

Computes the calibration weights (Chi-squared distance) for the estimation of the population total of several variables of interest.

**Usage**

```r
Wk(x, tx, Pik, ck, b0)
```

**Arguments**

- `x` Vector, matrix or data frame containing the recollected auxiliary information for every unit in the selected sample
- `tx` Vector containing the populations totals of the auxiliary information
- `Pik` A vector containing inclusion probabilities for each unit in the sample
- `ck` A vector of weights induced by the structure of variance of the supposed model
- `b0` By default FALSE. The intercept of the regression model

**Details**

The calibration weights satisfy the following expression

$$\sum_{k \in S} w_k x_k = \sum_{k \in U} x_k$$
Value

The function returns a vector of calibrated weights.

Author(s)

Hugo Andres Gutierrez Rojas <hagutierrezro@gmail.com>

References


Examples

#############
## Example 1
#############
# Without replacement sampling
# Vector U contains the label of a population of size N=5
U <- c("Yves", "Ken", "Erik", "Sharon", "Leslie")
# Vector x is the auxiliary information and y is the variables of interest
x<-c(32, 34, 46, 89, 35)
y<-c(52, 60, 75, 100, 50)
# pik is some vector of inclusion probabilities in the sample
# In this case the sample size is equal to the population size
pik<-rep(1,5)
w1<-Wk(x,tx=236,pik,ck=1,b0=FALSE)
sum(x*w1)
# Draws a sample size without replacement
sam <- sample(5,2)
pik <- c (0.8,0.2,0.2,0.5,0.3)
# The auxiliary information an variable of interest in the selected smaple
x.s<-x[sam]
y.s<-y[sam]
# The vector of inclusion probabilities in the selected smaple
pik.s<-pik[sam]
# Calibration weights under some specifics model
w2<-Wk(x.s,tx=236,pik.s,ck=1,b0=FALSE)
sum(x.s*w2)
w3<-Wk(x.s,tx=c(5,236),pik.s,ck=1,b0=TRUE)
sum(w3)
sum(x.s*w3)
w4<-Wk(x.s,tx=c(5,236),pik.s,ck=x.s,b0=TRUE)
sum(w4)
sum(x.s*w4)
w5<-Wk(x.s,tx=236,pik.s,ck=x.s,b0=FALSE)
sum(x.s*w5)
## Example 2: Linear models involving continuous auxiliary information

# Draws a simple random sample without replacement  
data(Lucy)  
attach(Lucy)  

N <- dim(Lucy)[1]  
n <- 400  
Pik <- rep(n/N, n)  
sam <- S.SI(N, n)  
# The information about the units in the sample is stored in an object called data  
data <- Lucy[sam,]  
attach(data)  
names(data)  

########### common ratio model ###################  
estima <- data.frame(Income)  
x <- Employees  
rx <- sum(Lucy$Employees)  
w <- Wk(x, rx, Pik, ck=1, b0=FALSE)  
sum(x*w)  
rx  
# The calibration estimation  
colSums(estima*w)  

########### Simple regression model without intercept ###################  
estima <- data.frame(Income, Employees)  
x <- Taxes  
rx <- sum(Lucy$Taxes)  
w <- Wk(x, rx, Pik, ck=x, b0=FALSE)  
sum(x*w)  
rx  
# The calibration estimation  
colSums(estima*w)  

########### Multiple regression model without intercept ###################  
estima <- data.frame(Income)  
x <- cbind(Employees, Taxes)  
rx <- c(sum(Lucy$Employees), sum(Lucy$Taxes))  
w <- Wk(x, rx, Pik, ck=1, b0=FALSE)  
sum(x[,1]*w)  
sum(x[,2]*w)  
rx  
# The calibration estimation  
colSums(estima*w)  

########### Simple regression model with intercept ###################
estima<-data.frame(Income, Employees)
x <- Taxes
tx <- c(N, sum(Lucy$Taxes))
w <- Wk(x,tx,Pik,ck=1,b0=TRUE)
sum(1*w)
sum(x*w)
tx
# The calibration estimation
colSums(estima*w)

############################ Multiple regression model with intercept ################################
estima<-data.frame(Income)
x <- cbind(Employees, Taxes)
tx <- c(N, sum(Lucy$Employees), sum(Lucy$Taxes))
w <- Wk(x,tx,Pik,ck=1,b0=TRUE)
sum(1*w)
sum(x[,1]*w)
sum(x[,2]*w)
tx
# The calibration estimation
colSums(estima*w)

# Example 3: Linear models involving discrete auxiliary information
Data(Lucy)
attach(Lucy)
N <- dim(Lucy)[1]
n <- 400
sam <- S.SI(N,n)
# The information about the units in the sample is stored in an object called data
data <- Lucy[sam,]
attach(data)
names(data)
# Vector of inclusion probabilities for units in the selected sample
Pik<-rep(n/N,n)
# The auxiliary information is discrete type
Doma<-Domains(Level)

############################ Poststratified common mean model ####################################
estima<-data.frame(Income, Employees, Taxes)
tx <- colSums(Domains(Lucy$Level))
w <- Wk(Doma,tx,Pik,ck=1,b0=FALSE)
sum(Doma[,1]*w)
sum(Doma[,2]*w)
sum(Doma[,3]*w)
tx
# The calibration estimation
colSums(estima*w)

########### Poststratified common ratio model ###################

estima<-data.frame(Income, Employees)
x<-Doma*Taxes
x <- colSums(Domains(Lucy$Level))
w <- Wk(x, tx, Pik, ck=1, b0=FALSE)
sum(x[,1]*w)
sum(x[,2]*w)
sum(x[,3]*w)
# The calibration estimation
colSums(estima*w)
Index

*Topic datasets

- BigCity, 3
- BigLucy, 4
- Lucy, 56

*Topic survey

- Deltakl, 6
- Domains, 7
- E.2SI, 10
- E.BE, 13
- E.Beta, 14
- E.piPS, 17
- E.PO, 19
- E.PPS, 20
- E.Quantile, 21
- E.SI, 23
- E.STpiPS, 26
- E.STPPS, 27
- E.STSI, 29
- E.SY, 31
- E.WR, 37
- GREG.SI, 38
- HH, 42
- HT, 45
- Ik, 51
- IkRS, 52
- IkWR, 53
- IPFP, 54
- nk, 57
- OrderWR, 58
- p.WR, 60
- Pik, 61
- PikHol, 63
- Pikl, 65
- PikPPS, 66
- S.BE, 70
- S.piPS, 72
- S.PO, 73
- S.PPS, 75
- S.SI, 76

- S.STpiPS, 78
- S.STPPS, 80
- S.STSI, 81
- S.SY, 83
- S.WR, 85
- Support, 86
- SupportRS, 87
- SupportWR, 89
- T.SIC, 90
- VarHT, 92
- Wk, 95

- BigCity, 3, 5, 57
- BigLucy, 3, 4, 57

- Deltakl, 6, 66, 92
- Domains, 7

- E.1SI, 8
- E.2SI, 9, 10, 34
- E.BE, 13, 71
- E.Beta, 14, 39
- E.piPS, 17, 67, 72
- E.PO, 19, 74
- E.PPS, 20, 75
- E.Quantile, 21
- E.SI, 8, 23, 77, 91
- E.STpiPS, 26, 78
- E.STPPS, 27, 80
- E.STSI, 29, 82
- E.SY, 31, 84
- E.Trim, 32
- E.UC, 33
- E.WR, 37, 85

- GREG.SI, 15, 38

- HH, 21, 42, 46
- HT, 22, 43, 45, 62, 92

- Ik, 51, 87
INDEX

IkRS, 52, 88
IkWR, 53
IPFP, 54

Lucy, 3, 5, 56

nk, 54, 57

OrderWR, 58
p.WR, 60
Pik, 6, 52–54, 58, 61, 66, 92
PikHol, 63, 67, 69
Pikl, 6, 65, 92
PikPPS, 66, 69
PikSTPPS, 68

S.BE, 14, 70
S.piPS, 18, 67, 72
S.PO, 19, 73
S.PPS, 21, 75
S.SI, 11, 24, 76, 91
S.STpiPS, 27, 69, 78
S.STPPS, 28, 80
S.STSI, 30, 81
S.SY, 32, 83
S.WR, 38, 85
Support, 52, 54, 59, 86, 89
SupportRS, 53, 87
SupportWR, 58, 59, 89

T.SIC, 90

VarHT, 6, 66, 92
VarSYGHT, 93

Wk, 95