Package ‘stratifyR’

October 14, 2022

Type Package
Title Optimal Stratification of Univariate Populations
Version 1.0-3
Date 2021-11-28
Author Karuna G. Reddy [aut, cre],
M. G. M. Khan [aut]
Maintainer Karuna G. Reddy <karuna.reddy@usp.ac.fj>
Description The stratification of univariate populations under stratified sampling designs is implemented according to Khan et al. (2002) <doi:10.1177/0008068320020518> and Khan et al. (2015) <doi:10.1080/02664763.2015.1018674> in this library. It determines the Optimum Strata Boundaries (OSB) and Optimum Sample Sizes (OSS) for the study variable, y, using the best-fit frequency distribution of a survey variable (if data is available) or a hypothetical distribution (if data is not available). The method formulates the problem of determining the OSB as mathematical programming problem which is solved by using a dynamic programming technique. If a dataset of the population is available to the surveyor, the method estimates its best-fit distribution and determines the OSB and OSS under Neyman allocation directly. When the dataset is not available, stratification is made based on the assumption that the values of the study variable, y, are available as hypothetical realizations of proxy values of y from recent surveys. Thus, it requires certain distributional assumptions about the study variable. At present, it handles stratification for the populations where the study variable follows a continuous distribution, namely, Pareto, Triangular, Right-triangular, Weibull, Gamma, Exponential, Uniform, Normal, Log-normal and Cauchy distributions.

LazyData true
License GPL-2
NeedsCompilation yes
Depends R (>= 3.4.0), fitdistrplus, zipfR, stats, actuar, triangle, mc2d
RoxygenNote 7.1.2
Suggests knitr, rmarkdown
VignetteBuilder knitr
Repository CRAN
Description

The Anaemia data comes from the Fiji National Nutritional Survey in 2004 on the "Micronutrient Status of Women in Fiji".

Usage

data(anaemia)

Format

A population data frame with 724 rows on some of the key components collected in the survey. The variables are:

- **Haemoglobin** Level of Haemoglobin (mmol/L)
- **Iron** Level of Iron (ng/mL)
- **Folate** Level of Folate (mmol/L)
create.mat

Source
This survey was conducted by the Ministry of Heath in Fiji. More details can be found at: http://ghdx.healthdata.org/record/fiji-national-nutrition-survey-2004

Examples

```r
data(anaemia)
head(anaemia)
Iron <- anaemia$Iron
min(Iron); max(Iron)
hist(anaemia$Haemoglobin)
boxplot(anaemia$Folate)
```

create.mat

To create and store calculated values of the objective function

Description
This function creates a matrix whose rows and columns depend on the range or distance of the data and the number of strata solutions that the user is seeking to compute. The matrix stores the objective function values calculated by the algorithm only to be accessed later for the purpose of presenting the OSB.

Usage

```r
create.mat(my_env)
```

Arguments

- **my_env**
  
  The environment my_env has various constants stored from earlier operations dealing with information on the data

Value

stores numerical quantities of the objective function and stores in the two matrices inside the my_env to be accessed by other functions

Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>
**data.alloc**

*To calculate the stratum sample sizes (nh) for a fixed sample size (n) directly based on the data*

**Description**

This function is called towards the final stages of the stratification process after OSB have been determined. It uses the boundaries to calculate the stratum sample allocations using Neyman allocation for all individual strata using the raw data.

**Usage**

```r
data.alloc(data, my_env)
```

**Arguments**

- **data**: A vector: provided as an input to the function
- **my_env**: The environment my_env has various constants and outputs stored from earlier operations through various other functions

**Value**

Calculates and stores quantities such as nh, Nh, Vh, etc. in the my_env to be accessed and printed as outputs

**Author(s)**

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

---

**data.optim**

*To implement the Dynamic Programming (DP) solution procedure on the stratification problem presented in the form of a Mathematical Programming Problem (MPP)*

**Description**

This function uses the Dynamic Programming (DP) solution procedure in solving the objective function for the univariate stratification problem. It calculates the objective function values using the brute-force algorithm and stores those values in the matrices and keeps a copy in my_env so that a global minimum could be obtained.

**Usage**

```r
data.optim(k, n, incf, minYk, maxYk, isFirstRun = TRUE, my_env)
```
**Arguments**

- **k**: A numeric: number of strata
- **n**: A numeric: is the distance*1000
- **incf**: A numeric: 10e-3 when k=1 and 10e-5 for k>=2
- **minYk**: A numeric: index to access minimum elements in the matrix
- **maxYk**: A numeric: index to access maximum elements in the matrix
- **isFirstRun**: A boolean: TRUE/FALSE parameter
- **my_env**: The environment my_env has various constants and calculations stored from earlier operations through various other functions

**Value**

returns the array filled with calculations of objective function values

**Author(s)**

Karuna Reddy <karuna.reddy@usp.ac.fj>

M GM Khan <khan_mg@usp.ac.fj>

---

**Description**

This function is called within other important functions in the stratifyR package to calculate the objective function values at systematic incremental progressions of stratum width and range of the data

**Usage**

data.root(d, y, c, my_env)

**Arguments**

- **d**: A numeric: distance or range of data
- **y**: A numeric: stratum width
- **c**: A numeric: stratum cost
- **my_env**: The environment my_env contains the constants and calculations stored from various calculations carried out by other key functions

**Value**

returns the value of the objective function
Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

distr.alloc

To calculate the stratum sample sizes (nh) for a fixed sample size (n) based on the hypothetical distribution of the data

Description

This function is called towards the final stages of the stratification process after OSB have been determined. It uses the boundaries to calculate the stratum sample allocations using Neyman allocation for all individual strata using the underlying distribution of the population.

Usage

distr.alloc(my_env)

Arguments

my_env

The environment my_env which has various constants and outputs stored from earlier operations through various other functions

Value

calculates and stores quantities such as nh, Nh, Vh, etc. in the my_env to be accessed and printed as outputs

Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

distr.optim

To implement the Dynamic Programming (DP) solution procedure on the stratification problem presented in the form of a Mathematical Programming Problem (MPP)

Description

This function uses the Dynamic Programming (DP) solution procedure in solving the objective function for the univariate stratification problem. It calculates the objective function values using the brute-force algorithm and stores those values in the matrices and keeps a copy in my_env so that a global minimum could be obtained.
Usage

distr.optim(k, n, incf, minYk, maxYk, isFirstRun = TRUE, my_env)

Arguments

k A numeric: number of strata
n A numeric: is the distance*1000
incf A numeric: 10e-3 when k=1 and 10e-5 for k>=2
minYk A numeric: index to access minimum elements in the matrix
maxYk A numeric: index to access maximum elements in the matrix
isFirstRun A boolean: TRUE/FALSE parameter
my_env My environment my_env has various constants and calculations stored from earlier operations through various other functions

Value

returns the array filled with calculations of objective function values

Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

---

distr.root Calculate the objective function values

Description

This function is called within other important functions in the package to calculate the objective function values at systematic incremental progressions of stratum width and range of the data

Usage

distr.root(d, y, c, my_env)

Arguments

d A numeric: distance or range of data
y A numeric: stratum width
c A numeric: stratum cost
my_env My environment my_env contains the constants and outputs from various calculations carried out by other key functions
Value
returns the value of the objective function

Author(s)
Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

erf
To calculate the error for a normal variable

Description
This function calculates the value of the error according to the normally distributed variable using
the idea presented in Abramowitz and Stegun (2011)

Usage
erf(x)

Arguments
x The data that is provided

Value
Gives the error for a normal variable

Author(s)
Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

get.dist
To identify the best-fit distribution of a univariate data

Description
This function is called at the start of the stratification process where the best-fit distribution and its
parameters are estimated and returned for further processing towards the computation of stratum
boundaries. It basically takes in the data and fits it with a list of 10 possible distributions and
computes the parameters for all given distributions. It selects the best-fit distribution to be the one
with the lowest AIC
Usage

get.dist(data, my_env)

Arguments

data A vector: usually a column in a given data frame
my_env My environment my_env has various constants and data that are used by the
get.dist() function

Value

returns a list which contains the best-fit distribution and its estimated parameters

Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

Description

The hies data comes from the HIES survey conducted in Fiji in the year 2010. The data contains only two aspects of the survey.

Usage

data(hies)

Format

A data frame with 3566 observations on two of the major quantities collected in the survey. The variables are:

- Expenditure Level of expenditure (FJD)
- Income Level of income (FJD)

Source

This survey was conducted in 2010 by the Bureau of Statistics (FIBoS) - Fiji Government.

Examples

data(hies$Income)
min(hies$Income); max(hies$Income)
hist(hies$Income)
boxplot(hies$Income)
### Description
The data contains the mathematics coursework marks, final examination marks and grades obtained by students in a first year mathematics course at The University level in the year 2010 in Fiji.

### Usage
```r
data(math)
```

### Format
A data frame with 353 observations which represent mathematics marks and grades for first year math students at university level. The variable is as follows:

- **cw**: Coursework marks in 1st year mathematics (0-50)
- **end_exam**: The end of semester examination marks maths (0-50)
- **final_marks**: Final examination marks in maths, which is an addition of the cw and end_exam (0-100)
- **grade**: The grade obtained by the student based on the final marks

### Source
The data was obtained by a masters students at USP, Fiji.

### Examples
```r
data(math)
min(math$final_marks); max(math$final_marks)
hist(math$final_marks)
boxplot(math$final_marks)
```

---

### minim.val

**To identify the minimum value out of two given sets of values**

### Description
This function is called in data.optim() or distr.optim() which basically compares and returns the smaller value out of two given sets of values.

### Usage
```r
minim.val(val1, val2)
```
**mode.val**

**Arguments**

- **val1** A numeric: the first value
- **val2** A numeric: the second value

**Value**

returns the minimum value

**Author(s)**

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

---

**Description**

This function calculates the value of the mode of the data that is provided

**Usage**

`mode.val(x)`

**Arguments**

- **x** The data that is provided

**Value**

Gives the mode

**Author(s)**

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>
To re-allocate the stratum sample sizes (nh)

Description

This function re-calculates or re-allocate the stratum sample sizes (nh) after it has already been initially allocated via Neyman allocation. This is applied to resolve the problem of oversampling in one or more of the strata.

Usage

realloc(h, x, nh, Nh, nume, my_env)

Arguments

- **h**: A numeric: the no. of strata
- **x**: A vector: the osb that has been calculated
- **nh**: A vector: the stratum sample sizes that have been initially calculated
- **Nh**: A vector: the stratum population sizes that have been initially calculated
- **nume**: A numeric: the numerator total
- **my_env**: The environment my_env has various constants and outputs stored from earlier operations through various other functions

Value

calculates and presents the new re-allocate stratum samples

Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

Stratification of Univariate Survey Population Using the Data

Description

This function takes in the univariate population data (argument data) and a fixed sample size (n) to compute the optimum stratum boundaries (OSB) for a given number of strata (L), optimum sample sizes (nh), etc. directly from the data. The main idea used is from Khan et al (2008) whereby the problem of stratification is formulated into a Mathematical Programming Problem (MPP) using the best-fit frequency distribution and its parameters estimated from the data. This MPP is then solved for the OSB using a Dynamic Programming (DP) solution procedure.
This function takes in the univariate population data (argument `data`) and a fixed sample size (`n`) to compute the optimum stratum boundaries (OSB) for a given number of strata (`L`), optimum sample sizes (`nh`), etc. directly from the data. The main idea used is from Khan et al (2008) whereby the problem of stratification is formulated into a Mathematical Programming Problem (MPP) using the best-fit frequency distribution and its parameters estimated from the data. This MPP is then solved for the OSB using a Dynamic Programming (DP) solution procedure.

**Usage**

```r
strata.data(data, h, n, cost = FALSE, ch = NULL)
```

**Arguments**

- `data`: A vector of values of the survey variable y for which the OSB are determined.
- `h`: A numeric: denotes the number of strata to be created.
- `n`: A numeric: denotes a fixed total sample size.
- `cost`: A logical: has default cost=FALSE. If it is a stratum-cost problem, cost=TRUE, with which, one must provide the Ch parameter.
- `ch`: A numeric: denotes a vector of stratum costs. When cost=FALSE, it has a default of NULL.

**Value**

The function `strata.data` returns Optimum Strata Boundaries (OSB), stratum weights (Wh), stratum variances (Vh), Optimum Sample Sizes (nh), stratum population sizes (Nh) and sampling fraction (fh).

**Author(s)**

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

**See Also**

- `strata.distr`
- `strata.distr`

**Examples**

```r
## Not run:
data <- rweibull(1000, shape=2, scale = 1.5)
hist(data)
obj <- strata.data(data, h = 2, n=300)
```
summary(obj)
#-------------------------------------------------------------
data(anaemia)
Iron <- anaemia$Iron
res <- strata.data(Iron, h = 2, n=350)
summary(res)
#-------------------------------------------------------------
data(SHS) #Household Spending data from stratification package
weight <- SHS$WEIGHT
hist(weight); length(weight)
res <- strata.data(weight, h = 2, n=500)
summary(res)
#-------------------------------------------------------------
data(sugarcane)
Production <- sugarcane$Production
hist(Production)
res <- strata.data(Production, h = 2, n=1000)
summary(res)
#-------------------------------------------------------------

# The function be dynamically used to visualize the the strata boundaries,
# for 2 strata, over the density (or observations) of the "mag" variable
# from the quakes data (with purrr and ggplot2 packages loaded).
output <- quakes %>%
  pluck("mag") %>%
  strata.data(h = 2, n = 300)
quakes %>%
ggplot(aes(x = mag)) +
  geom_density(fill = "blue", colour = "black", alpha = 0.3) +
  geom_vline(xintercept = output$OSB, linetype = "dotted", color = "red")
#-------------------------------------------------------------

## End(Not run)

## Not run:
data <- rweibull(1000, shape=2, scale = 1.5)
hist(data)
obj <- strata.data(data, h = 2, n=300)
summary(obj)
#-------------------------------------------------------------
data(anaemia)
Iron <- anaemia$Iron
res <- strata.data(Iron, h = 2, n=350)
summary(res)
#-------------------------------------------------------------
data(SHS) #Household Spending data from stratification package
weight <- SHS$WEIGHT
hist(weight); length(weight)
res <- strata.data(weight, h = 2, n=500)
summary(res)
#-------------------------------------------------------------
data(sugarcane)
Production <- sugarcane$Production
hist(Production)
strata.distr

Stratification of Univariate Survey Population Using the Distribution

Description

This function takes in the underlying hypothetical distribution and its parameter(s) of the survey variable, the initial value and the range of the population, the fixed sample size (n) and the fixed population size (N) to compute the optimum stratum boundaries (OSB) for a given number of strata (L), optimum sample sizes (nh), etc. The main idea used is from Khan et al. (2008) whereby the problem of stratification is formulated into a Mathematical Programming Problem (MPP) using the best-fit frequency distribution and its parameter estimates of the data. This MPP is then solved for the optimal solutions using the Dynamic Programming (DP) solution procedure.

Usage

strata.distr(
  h,
  initval,
  dist,
  distr = c("pareto", "triangle", "rtriangle", "weibull", "gamma", "exp", "unif",
            "norm", "lnorm", "cauchy"),
  params = c(shape = 0, scale = 0, rate = 0, gamma = 0, location = 0, mean = 0, sd = 0,
             meanlog = 0, sdlog = 0, min = 0, max = 0, mode = 0),
  n,
  N,
  cost = FALSE,
  ch = NULL
)
Arguments

- **h** A numeric: denotes the number of strata to be created.
- **initval** A numeric: denotes the initial value of the population
- **dist** A numeric: denotes distance (or range) of the population
- **distr** A character: denotes the name of the distribution that characterizes the population
- **params** A list: contains the values of all parameters of the distribution
- **n** A numeric: denotes the fixed total sample size.
- **N** A numeric: denotes the fixed total population size.
- **cost** A logical: has default cost=FALSE. If it is a stratum-cost problem, cost=TRUE, with which one must provide the Ch parameter.
- **ch** A numeric: denotes a vector of stratum costs.

Value

strata.distr returns Optimum Strata Boundaries (OSB), stratum weights (Wh), stratum costs (Ch), stratum variances (Vh), Optimum Sample Sizes (nh), stratum population sizes (Nh).

Author(s)

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

See Also

strata.data

Examples

```r
## Not run:
#Assume data has initial value of 1.5, distance of 33 and follows
#weibull distribution with estimated parameters as shape=2.15 and scale=13.5
#To compute the OSB, OSS, etc. with fixed sample n=500, we use:
res <- strata.distr(h=2, initval=1.5, dist=33, distr = "weibull",
params = c(shape=2.15, scale=13.5), n=500, N=2000, cost=FALSE)
summary(res)
#-------------------------------------------------------------
#Assume data has initial value of 1, distance of 10415 and follows
#lnorm distribution with estimated parameters as meanlog=5.5 and sdlog=1.5
#To compute the OSB, OSS, etc. with fixed sample n=500, we use:
res <- strata.distr(h=2, initval=1, dist=10415, distr = "lnorm",
params = c(meanlog=5.5, sdlog=1.5), n=500, N=12000)
summary(res)
#-------------------------------------------------------------
#Assume data has initial value of 2, distance of 68 and follows
#gamma distribution with estimated parameters as shape=3.8 and rate=0.55
#To compute the OSB, OSS, etc. with fixed sample n=500, we use:
res <- strata.distr(h=2, initval=0.65, dist=68, distr = "gamma",
```
sugarcane

Description

The sugarcane data shows the disposition area (land area under cane) for individual sugarcane farms and their cane productions with the incomes/earnings for the year 2010 in Fiji.

Usage

data(sugarcane)

Format

A data frame with 13894 observations corresponding to individual farms. The following are the variables:

- `DispArea` Disposition area (or land area under cane) (hectares)
- `Production` The amount of sugarcane produced in the farm (tonnes)
- `Income` Net income or money paid to farmers (in FJD)

Source

This data was obtained from the Fiji Sugar Corporation in Fiji.

Examples

data(sugarcane$Production)
head(sugarcane$Production)
Production <- sugarcane$Production
min(Production); max(Production)
hist(Production)
boxplot(Production)
This method formats and outputs the final results to the R console

**Summary**

This function defines the method for the "strata" class that has been created in the constructor function (strata.data() or strata.distr()) where all computed objects, via other relevant functions, are collated and passed as a list. The function extracts all individual objects from the "strata" class object and combines them into dataframes before writing the formatted outputs to the console. This is used for both cases, depending on either the data or a hypothetical distribution.

**Usage**

```r
## S3 method for class 'strata'
summary(object, ...)
```

**Arguments**

- `object`: A list: An object of class "strata".
- `...`: Any data type: This argument can be any particular argument.

**Value**

returns the formatted output

**Author(s)**

Karuna Reddy <karuna.reddy@usp.ac.fj>
MGM Khan <khan_mg@usp.ac.fj>

**Examples**

```r
## Not run:
data <- rweibull(1000, shape=2, scale = 1.5)
res <- strata.data(data, h = 2, n=300)
summary(res)

## End(Not run)
```
Index

* datasets
  anaemia, 2
  hies, 9
  math, 10
  sugarcane, 17

anaemia, 2
create.mat, 3
data.alloc, 4
data.optim, 4
data.root, 5
distr.alloc, 6
distr.optim, 6
distr.root, 7

erf, 8
get.dist, 8
hies, 9
math, 10
minim.val, 10
mode.val, 11
reallocate, 12
strata.data, 12
strata.distr, 15
sugarcane, 17
summary.strata, 18