Package 'sms'

November 15, 2015

Type Package

Title Spatial Microsimulation

Version 2.3.1

Date 2015-11-10

Author Dimitris Kavroudakis <dimitris123@gmail.com>

Maintainer Dimitris Kavroudakis <dimitris123@gmail.com>

Description Produce small area population estimates by fitting census data to survey data.

License GPL-3

LazyLoad yes

Depends doParallel, foreach, parallel, methods, iterators, stats

RoxygenNote 5.0.0

NeedsCompilation no

Repository CRAN

Date/Publication 2015-11-15 10:09:12

R topics documented:

`sms-package` *Spatial Microsimulation Library*

Description

Generate small area population microdata from census and survey datasets. Fit the survey data to census area descriptions and export the population of small areas (microdata).

Details

Generate small area population microdata from census and panel datasets. Fit the survey data to census area descriptions and export the popultion of small areas.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

References

Dimitris Kavroudakis D (2015). **sms: An R Package for the Construction of Microdata for Geographical Analysis.** *Journal of Statistical Software*, **68**(2), pp. 1-23. http://10.18637/jss.v068.i02

`addDataAssociation` *addDataAssociation*

Description

Create a data lexicon for holding the associated column names

Usage

```
addDataAssociation(indf, data_names)
```

Arguments

indf	A data Lexicon (data.frame) created from the function: createLexicon
data_names	A vector vith two elements. The first element should be the name of the census data column, and the second element should be the name of the survey data column

Value

indf The imported data lexicon with one extra column.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey)
data(census)
in.lexicon=createLexicon()
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))
print(in.lexicon)
```

---

calculate_error *Calculate error of a selection*

Description

Calculate the error of a selection.

Usage

```
calculate_error(selection, area_census, lexicon)
```

Arguments

selection	A population selection, to evaluate its error
area_census	An area from census (a row)
lexicon	A data.frame with details about data connections

Details

Calculates the Total Absolute Error (TAE) of a selection for a census area.

Value

TAE Total Absolute Error of this selection against the census description of this area.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey) #load the data
data(census)
in.lexicon=createLexicon() # Create a data lexicon for holding the associated column names.
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

#Select the first area from the census table
this_area=as.data.frame(census[1,])

#make a random selection of individuals for this area.
selection=random_panel_selection( survey, this_area$population )

#evaluate the Total Absolute Error (TAE) for this selection
error=calculate_error( selection, this_area, in.lexicon )
print( error )    # print the error of the selection
```

---

census *A census dataset of 10 areas*

Description

A sample census dataset containing descriptive information about 10 geographical areas. The variables in the dataset are as follows:

- areaid: The unique indentifier of the area
- population: The number of indivisuals in the area.
- he: Number of individuals in the area, with at least Higher Education degree
- females: Number of female individuals in the area

Usage

```
data(census)
```

Format

A data frame with 10 rows and 4 variables

checkIfNamesInDataColumns *checkIfNamesInDataColumns*

Description

Check the integrisy of the data Lexicon

Usage

```
checkIfNamesInDataColumns(names, incensus, insurvey)
```

Arguments

names	A vector with names to check if they exist as column names in the data (census and survey)
incensus	The census data
insurvey	The survey data

Value

anumber If both names are valid then it return '1' else if the names are not valid data column names, it returns '0'.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

check_lexicon *check_lexicon*

Description

Check the lexicon data.frame

Usage

```
check_lexicon(inlex)
```

Arguments

inlex	A data.frame which will be used a data lexicon for listing the associated data columns.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
df=createLexicon()
df=addDataAssociation(df, c("ena","duo"))
check_lexicon(df)
```

---

createLexicon *createLexicon*

Description

Create a data lexicon for holding the associated column names

Usage

```
createLexicon()
```

Value

dataLexicon A data.frame holding the associated column names.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey)
data(census)
in.lexicon=createLexicon()
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))
print(in.lexicon)
```

find_best_selection *find_best_selection*

Description

Find the best selection of individual records for a census area.

Usage

```
find_best_selection(area, insms, inseed = -1)
```

Arguments

area	A census area
insms	A microsimulation object which holds the data and details of the simulation such as iterations, lexicon.
inseed	test

Details

Calculate the best area representation, after a series of selection tries.

Value

list A list with results (#areaid, #selection, #tae, #tries, #error_states).

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey) #load the data
data(census)
in.lexicon=createLexicon() # Create a data lexicon for holding the associated column names.
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

this_area=as.data.frame(census[1,]) #Select the first area from the census table
insms= new("microsimulation",census=census,panel=survey, lexicon=in.lexicon, iterations=10)
best=find_best_selection(this_area, insms)
print(best)
```

find_best_selection_SA *find_best_selection_SA*

Description

Run a simulation in parallel mode with Simulated Annealing

Usage

```
find_best_selection_SA(area_census, insms, inseed = -1)
```

Arguments

area_census	A census dataset consisting of various areas rows.
insms	A microsimulation object which holds the data and details of the simulation such as iterations, lexicon.
inseed	A number to be used for random seed.

Value

msm_results An object with the results of the simulation, of this area.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey)
data(census)
in.lexicon=createLexicon()
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

this_area=as.data.frame(census[1,]) #Select the first area from the census table
insms= new("microsimulation",census=census, panel=survey, lexicon=in.lexicon, iterations=5)
myselection= find_best_selection_SA( this_area, insms, inseed=1900)
print(myselection)
```

getInfo *getInfo Generic*

Description

getInfo Generic

Usage

```
getInfo(object)
```

Arguments

object	A microsimulation object to get its information.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

getInfo,microsimulation-method *getInfo Method*

Description

Get information from a microsimulation object

Usage

```
## S4 method for signature 'microsimulation'
getInfo(object)
```

Arguments

object	A microsimulation object to get its information.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

`getTAEs` *getTAEs Generic*

Description

Get the TAE from a microsimulation object.

Usage

```
getTAEs(object)
```

Arguments

`object`	A microsimulation object to get its information.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

`getTAEs,microsimulation-method` *getTAEs Method*

Description

getTAEs Method

Usage

```
## S4 method for signature 'microsimulation'
getTAEs(object)
```

Arguments

`object`	A microsimulation object to get its information.

Value

taes A list of numbers indicating the Total Absolute Error of the fitting process for each of the census areas.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

microsimulation-class *A microsimulation object*

Description

It holds all microsimulation details and objects such as data, results etc.

Arguments

census:	A census data.frame where each row contains census information about a geographical area
panel:	A data.frame containing the individual based records from a panel survey. Those data will be fitted to small area contrains and will populate each vrtual area.
lexicon:	A data.frame containing the association of columns between census data and panel data. Each row contain a conection between census and panel data.frame.
resuls:	A list of results from the fitting process.
iterations:	The number of itertions until th end of the fitting process.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

mysetSeed *mysetSeed*

Description

mysetSeed

Usage

```
mysetSeed(inseed)
```

Arguments

inseed	A number to set as a random seed.

Details

mysetSeed

Examples

```
library(sms)
sms::mysetSeed(1900)
```

plotTries *Plot selection results*

Description

Plot the selection process of an area from a microsimulation object.

Usage

```
plotTries(insms, number)
```

Arguments

insms	The input results
number	the number of the area to plot

Details

Plot errors during selection process for an area.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey) #load the data
data(census)
in.lexicon=createLexicon() # Create a data lexicon for holding the associated column names.
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

ansms = new("microsimulation", census=census, panel=survey, lexicon=in.lexicon, iterations=5)
sa = run_parallel_SA(ansms, inseed=1900)
plotTries( sa, 1 )
```

---

random_panel_selection *random_panel_selection*

Description

Select n random rows from a dataframe

Usage

```
random_panel_selection(indf, n)
```

Arguments

indf	The initial dataframe from wich a selection will be made.
n	The number of random rows

Details

Select n random rows from a dataframe

Value

a selection of rows as a dataframe

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey) #load the data
data(census)

some.individuals=random_panel_selection(survey,4)
print(some.individuals)     # Print the selection of individuals
```

---

run_parallel_HC *run_parallel_HC*

Description

Run a simulation in serial mode with Hill Climbing

Usage

```
run_parallel_HC(insms, inseed = -1)
```

Arguments

insms	A microsimulation object which holds the data and details of the simulation such as iterations, lexicon.
inseed	A number to be used for random seed.

Details

Run a simulation in serial mode with Hill Climbing

Value

msm_results An object with the results of the simulation, for each area.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey) #load the data
data(census)
in.lexicon=createLexicon() # Create a data lexicon for holding the associated column names.
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

insms= new("microsimulation",census=census,panel=survey, lexicon=in.lexicon, iterations=10)
re=run_parallel_HC(insms, inseed=1900)
print(re)
```

`run_parallel_SA` *run_parallel_SA*

Description

Run a simulation in parallel mode with Simulated Annealing

Usage

```
run_parallel_SA(insms, inseed = -1)
```

Arguments

`insms`	A microsimulation object which holds the data and details of the simulation such as iterations, lexicon.
`inseed`	A random number to be used for random seed.

Value

msm_results An object with the results of the simulation, for each area.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey)
data(census)
in.lexicon=createLexicon()
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

insms= new("microsimulation",census=census, panel=survey, lexicon=in.lexicon, iterations=5)
results= run_parallel_SA(insms, inseed=1900)
print(results)
```

`run_serial` *Run_serial*

Description

Run a simulation in serial mode

Usage

```
run_serial(insms)
```

Arguments

`insms`	A microsimulation object which holds the data and details of the simulation such as iterations, lexicon.

Details

Run a simulation in serial mode.

Value

msm_results An object with the results of the simulation, for each area.

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey)
data(census)
in.lexicon=createLexicon()
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))
```

```
insms= new("microsimulation",census=census, panel=survey, lexicon=in.lexicon, iterations=5)
results= run_serial( insms)
print(results)
```

`selection_for_area` *selection_for_area*

Description

Make a single selection of individual records for a census area.

Usage

```
selection_for_area(inpanel, area_census, inlexicon)
```

Arguments

`inpanel`	The panel dataset
`area_census`	A census area
`inlexicon`	A data lexicon showing the variable associations.

Details

Select a number of individual records from panel dataset, to represent a census description of an area.

Value

list A list of results (#areaid, #selection, #error)

Author(s)

Dimitris Kavroudakis <dimitris123@gmail.com>

Examples

```
library(sms)
data(survey) #load the data
data(census)
in.lexicon=createLexicon() # Create a data lexicon for holding the associated column names.
in.lexicon=addDataAssociation(in.lexicon, c("he","he"))
in.lexicon=addDataAssociation(in.lexicon, c("females","female"))

# Select the first area from the census table
this_area=as.data.frame(census[1,])

#make a representation for this area.
sel=selection_for_area(survey, this_area, in.lexicon)

print(sel) #print the representation
```

survey *A survey dataset of 200 individuals*

Description

A sample survey dataset containing binary (0 or 1) information about 200 individuals. Those individuals will be used to populate the simulated areas. The variables in the dataset are as follows:

- pid: The unique indentifier of the individual
- female: Binary value of the sex of the individual. 1-Female, 0-Male
- agemature: Binary value indicating if the individual belongs to the mature age group. 0-No, 1-Yes
- car_owner: Binary value indicating if the individual owns a car. 0-No, 1-Yes
- house_owner: Binary value indicating if the individual owns a house. 0-No, 1-Yes
- working: Binary value indicating if the individual is working. 0-No, 1-Yes

Usage

```
data(survey)
```

Format

A data frame with 200 rows and 7 variables

Index