

Package ‘BalancedSampling’

January 27, 2016

Type Package

Title Balanced and Spatially Balanced Sampling

Version 1.5.1

Date 2016-01-25

Author Anton Grafström, Jonathan Lisic

Maintainer Anton Grafström <anton.grafstrom@gmail.com>

Description Select balanced and spatially balanced probability samples in multi-dimensional spaces with any prescribed inclusion probabilities. It contains fast (C++ via Rcpp) implementations of the included sampling methods. The local pivotal method and spatially correlated Poisson sampling (for spatially balanced sampling) are included. Also the cube method (for balanced sampling) and the local cube method (for doubly balanced sampling) are included.

License GPL (>=2)

Imports Rcpp (>= 0.11.1)

LinkingTo Rcpp

Encoding UTF-8

URL <http://www.antongrafstrom.se/balancedsampling>

NeedsCompilation yes

R topics documented:

BalancedSampling-package	2
cube	4
cubestratified	6
flightphase	7
landingphase	8
lcube	9
lcubeflightphase	10
lcubelandingphase	11
lcubestratified	12
lpm	13
lpm1	14
lpm2	15
lpm2_kdtree	16
probabilities	17

rpm	18
sb	19
scps	20
spm	21
vsb	22
Index	23

BalancedSampling-package

Balanced and Spatially Balanced Sampling

Description

Select balanced and spatially balanced probability samples in multi-dimensional spaces with any prescribed inclusion probabilities. It contains fast (C++ via Rcpp) implementations of the included sampling methods. The local pivotal method and spatially correlated Poisson sampling (for spatially balanced sampling) are included. Also the cube method (for balanced sampling) and the local cube method (for doubly balanced sampling) are included.

Author(s)

Anton Grafström, Jonathan Lisic

Maintainer: Anton Grafström <anton.grafstrom@gmail.com>

Webpage: <http://www.antongrafstrom.se/balancedsampling>

References

Deville, J. C. and Tillé, Y. (2004). Efficient balanced sampling: the cube method. *Biometrika*, 91(4), 893-912.

Deville, J.-C. and Tillé, Y. (1998). Unequal probability sampling without replacement through a splitting method. *Biometrika* 85, 89-101.

Grafström, A. (2012). Spatially correlated Poisson sampling. *Journal of Statistical Planning and Inference*, 142(1), 139-147.

Grafström, A. and Lundström, N.L.P. (2013). Why well spread probability samples are balanced. *Open Journal of Statistics*, 3(1).

Grafström, A. and Schelin, L. (2014). How to select representative samples. *Scandinavian Journal of Statistics*.

Grafström, A., Lundström, N.L.P. and Schelin, L. (2012). Spatially balanced sampling through the Pivotal method. *Biometrics* 68(2), 514-520.

Grafström, A. and Tillé, Y. (2013). Doubly balanced spatial sampling with spreading and restitution of auxiliary totals. *Environmetrics*, 24(2), 120-131.

Examples

```

# *****
# check inclusion probabilities
# *****
set.seed(1234567);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9);
N = length(p);
X = cbind(runif(N),runif(N));
p1 = p2 = p3 = p4 = rep(0,N);
nrs = 1000; # increase for more precision
for(i in 1:nrs){
  # lpm1
  s = lpm1(p,X);
  p1[s]=p1[s]+1;

  # lpm2
  s = lpm2(p,X);
  p2[s]=p2[s]+1;

  # scps
  s = scps(p,X);
  p3[s]=p3[s]+1;

  # lcube
  s = lcube(p,X,cbind(p));
  p4[s]=p4[s]+1;
}
print(p);
print(p1/nrs);
print(p2/nrs);
print(p3/nrs);
print(p4/nrs);

# *****
# check spatial balance
# *****
set.seed(1234567);
N = 500;
n = 70;
p = rep(n/N,N);
X = cbind(runif(N),runif(N));
nrs = 10; # increase for more precision
b1 = b2 = b3 = b4 = b5 = rep(0,nrs);

for(i in 1:nrs){
  # lpm1
  s = lpm1(p,X);
  b1[i] = sb(p,X,s);

  # lpm2
  s = lpm2(p,X);
  b2[i] = sb(p,X,s);

  # scps
  s = scps(p,X);
  b3[i] = sb(p,X,s);
}

```

```

# lcube
s = lcube(p,X,cbind(p));
b4[i] = sb(p,X,s);

# srs
s = sample(N,n);
b5[i] = sb(p,X,s);
}
print(mean(b1));
print(mean(b2));
print(mean(b3));
print(mean(b4));
print(mean(b5));

# *****
# stratification
# *****
set.seed(1234567);
N = 10;
n = 4;
p = rep(n/N,N);
stratum1 = c(1,1,1,1,1,0,0,0,0,0); # stratum 1 indicator
stratum2 = c(0,0,0,0,0,1,1,1,1,1); # stratum 2 indicator
stratum3 = c(0,0,1,1,1,1,1,0,0,0); # overlapping 1 and 2
s = lpm1(p,cbind(stratum1,stratum2,stratum3));

# *****
# plot spatially balanced sample
# *****
set.seed(1234567);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(runif(N),runif(N)); # matrix of auxiliary variables
s = lpm1(p,X); # select sample
plot(X[,1],X[,2]); # plot population
points(X[s,1],X[s,2], pch=19); # plot sample

# *****
# check cpu time (for simulation)
# *****
set.seed(1234567);
N = 2000;
n = 100;
X = cbind(runif(N),runif(N));
p = rep(n/N,N);
system.time(for(i in 1:10){lpm1(p,X)});
system.time(for(i in 1:10){lpm2(p,X)});

```

Description

This is a fast implementation of the cube method. To have a fixed sample size, include the inclusion probabilities as a balancing variable in `Xbal` and make sure the inclusion probabilities sum to a positive integer. Landing is done by dropping balancing variables (from rightmost column, so keep inclusion probabilities in first column to guarantee fixed size).

Usage

```
cube(prob, Xbal)
```

Arguments

<code>prob</code>	vector of length <code>N</code> with inclusion probabilities
<code>Xbal</code>	matrix of balancing auxiliary variables of <code>N</code> rows and <code>r</code> columns

Value

Returns a vector of selected indexes in `1,2,...,N`.

References

Deville, J. C. and Tillé, Y. (2004). Efficient balanced sampling: the cube method. *Biometrika*, 91(4), 893-912.

Chauvet, G. and Tillé, Y. (2006). A fast algorithm for balanced sampling. *Computational Statistics*, 21(1), 53-62.

Examples

```
## Not run:
# Example 1
# Select sample
set.seed(12345);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(p,runif(N),runif(N)); # matrix of auxiliary variables
s = cube(p,X); # select sample

# Example 2
# Check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = cube(p,cbind(p));
  ep[s]=ep[s]+1;
}
print(ep/nrs);

# Example 3
# How fast is it?
# Lets check with N = 100 000 and 5 balancing variables
```

```

set.seed(12345);
N = 100000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
# matrix of 5 auxiliary variables
X = cbind(p,runif(N),runif(N),runif(N),runif(N));
system.time(cube(p,X));

## End(Not run)

```

cubestratified

Stratified balanced sampling with pooling of landing phases

Description

This is a fast implementation of stratified balanced sampling. To have a fixed sample size, include the inclusion probabilities as a balancing variable in `Xbal` and make sure the inclusion probabilities sum to a positive integer (within each stratum).

Usage

```
cubestratified(prob,Xbal, integerStrata)
```

Arguments

<code>prob</code>	vector of length <code>N</code> with inclusion probabilities
<code>Xbal</code>	matrix of balancing auxiliary variables of <code>N</code> rows and <code>r</code> columns
<code>integerStrata</code>	vector of length <code>N</code> with stratum number

Value

Returns a vector of length `N` with sampling indicators.

References

Chauvet, G. (2009). Stratified balanced sampling. *Survey Methodology*, 35, 115-119.

Examples

```

## Not run:
# Example 1
N = 10;
n = 5;
p = rep(n/N,N);
strata = c(1,1,2,2,3,3,4,4,5,5);
indicators = cubestratified(p,cbind(p),strata);
s = (1:N)[indicators==1];

## End(Not run)

```

`flightphase`*Flight phase of the cube method*

Description

This is a fast implementation of the flight phase of the cube method. To have a fixed sample size, include the inclusion probabilities as a balancing variable in `Xbal` and make sure the inclusion probabilities sum to a positive integer.

Usage

```
flightphase(prob, Xbal)
```

Arguments

<code>prob</code>	vector of length <code>N</code> with inclusion probabilities
<code>Xbal</code>	matrix of balancing auxiliary variables of <code>N</code> rows and <code>q</code> columns

Value

Returns a vector of length `N` with new probabilities, where at most `q` are non-integer.

References

Deville, J. C. and Tillé, Y. (2004). Efficient balanced sampling: the cube method. *Biometrika*, 91(4), 893-912.

Chauvet, G. and Tillé, Y. (2006). A fast algorithm for balanced sampling. *Computational Statistics*, 21(1), 53-62.

Examples

```
## Not run:
# Example 1
# Select sample and check balance
set.seed(12345);
N = 100; # population size
n = 10; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(p,runif(N),runif(N)); # matrix of auxiliary variables

pflight = flightphase(p,X);

# check balance after flight
colSums(X)
colSums(X/p*pflight)

# select final sample as indicators
indicators = landingphase(p,pflight,X);

# check final balance
colSums(X)
colSums(X/p*indicators)
```

```
# final sample as indexes
s = (1:N)[indicators==1];

## End(Not run)
```

landingphase

Landing phase of the cube method

Description

Landing is done by dropping balancing variables (from rightmost column).

Usage

```
landingphase(prob,probflight,Xbal)
```

Arguments

prob	vector of length N with inclusion probabilities
probflight	vector of length N obtained from the flightphase
Xbal	matrix of balancing auxiliary variables of N rows and q columns

Value

Returns a vector of length N with inclusion indicators.

References

Deville, J. C. and Tillé, Y. (2004). Efficient balanced sampling: the cube method. *Biometrika*, 91(4), 893-912.

Chauvet, G. and Tillé, Y. (2006). A fast algorithm for balanced sampling. *Computational Statistics*, 21(1), 53-62.

Examples

```
## Not run:
# Example 1
# Select sample
set.seed(12345);
N = 100; # population size
n = 10; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(p,runif(N),runif(N)); # matrix of auxiliary variables
pflight = flightphase(p,X); # flight
indicators = landingphase(p,pflight,X); # landing
# final sample
s = (1:N)[indicators==1];

## End(Not run)
```


lcube

*Local cube method (Doubly balanced sampling)***Description**

Select doubly balanced samples with prescribed inclusion probabilities from a finite population. To have a fixed sample size, include the inclusion probabilities as a balancing variable in Xbal and make sure the inclusion probabilities sum to a positive integer. This is a simplified (optimized for speed) implementation of the local cube method (doubly balanced sampling). Landing is done by dropping balancing variables (from rightmost column, so keep inclusion probabilities in first column to guarantee fixed size). Euclidean distance is used in the Xspread space.

Usage

```
lcube(prob, Xspread, Xbal)
```

Arguments

prob	vector of length N with inclusion probabilities
Xspread	matrix of (standardized) auxiliary variables of N rows and q columns
Xbal	matrix of balancing auxiliary variables of N rows and r columns

Value

Returns a vector of selected indexes in 1,2,...,N.

References

Grafström, A. and Tillé, Y. (2013). Doubly balanced spatial sampling with spreading and restitution of auxiliary totals. *Environmetrics*, 24(2), 120-131.

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(runif(N),runif(N)); # matrix of auxiliary variables
s = lcube(p,X,cbind(p)); # select sample
plot(X[,1],X[,2]); # plot population
points(X[s,1],X[s,2], pch=19); # plot sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
X = cbind(runif(N),runif(N)); # some artificial auxiliary variables
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
```

```

    s = lcube(p,X,cbind(p));
    ep[s]=ep[s]+1;
  }
  print(ep/nrs);

## End(Not run)

```

 lcubeflightphase

Flight phase for the local cube method

Description

Flight phase for the local cube method. To have a fixed sample size, include the inclusion probabilities as a balancing variable in Xbal and make sure the inclusion probabilities sum to a positive integer. This is a simplified (optimized for speed) implementation of the flight phase of the local cube method (doubly balanced sampling). Euclidean distance is used in the Xspread space.

Usage

```
lcubeflightphase(prob,Xspread,Xbal)
```

Arguments

prob	vector of length N with inclusion probabilities
Xspread	matrix of (standardized) auxiliary variables of N rows and q columns
Xbal	matrix of balancing auxiliary variables of N rows and r columns

Value

Returns a vector of length N with new probabilities, where at most r are non-integer.

References

Grafström, A. and Tillé, Y. (2013). Doubly balanced spatial sampling with spreading and restitution of auxiliary totals. *Environmetrics*, 24(2), 120-131.

Examples

```

## Not run:
# Example 1
# Select sample
set.seed(12345);
N = 100; # population size
n = 10; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(runif(N),runif(N)); # matrix of auxiliary variables
pflight = lcubeflightphase(p,X,cbind(p,X));
# check balance
colSums(X)
colSums(X/p*pflight)

## End(Not run)

```

lcubelandingphase	<i>Landing phase for the local cube method</i>
-------------------	--

Description

Landing is done by dropping balancing variables (from rightmost column). Euclidean distance is used in the Xspread space.

Usage

```
lcubelandingphase(prob, probflight, Xspread, Xbal)
```

Arguments

prob	vector of length N with inclusion probabilities
probflight	vector of length N with probabilities from flightphase
Xspread	matrix of (standardized) auxiliary variables of N rows and q columns
Xbal	matrix of balancing auxiliary variables of N rows and r columns

Value

Returns a vector of length N with indicators.

References

Grafström, A. and Tillé, Y. (2013). Doubly balanced spatial sampling with spreading and restitution of auxiliary totals. *Environmetrics*, 24(2), 120-131.

Examples

```
## Not run:  
# Example 1  
# Select sample  
set.seed(12345);  
N = 100; # population size  
n = 10; # sample size  
p = rep(n/N, N); # inclusion probabilities  
X = cbind(runif(N), runif(N)); # matrix of auxiliary variables  
pflight = lcubeflightphase(p, X, cbind(p, X)); # flight  
indicators = lcubelandingphase(p, pflight, X, cbind(p, X)); # landing  
# final sample  
s = (1:N)[indicators==1];  
  
## End(Not run)
```

lcubestratified

*Stratified doubly balanced sampling with pooling of landing phases***Description**

This is a fast implementation of stratified doubly balanced sampling. To have a fixed sample size, include the inclusion probabilities as a balancing variable in `Xbal` and make sure the inclusion probabilities sum to a positive integer (within each stratum). Euclidean distance is used in the `Xspread` space.

Usage

```
lcubestratified(prob,Xspread,Xbal, integerStrata)
```

Arguments

<code>prob</code>	vector of length <code>N</code> with inclusion probabilities
<code>Xspread</code>	matrix of (standardized) auxiliary variables of <code>N</code> rows and <code>q</code> columns
<code>Xbal</code>	matrix of balancing auxiliary variables of <code>N</code> rows and <code>r</code> columns
<code>integerStrata</code>	vector of length <code>N</code> with stratum number

Value

Returns a vector of length `N` with sampling indicators.

References

Chauvet, G. (2009). Stratified balanced sampling. *Survey Methodology*, 35, 115-119.

Grafström, A. and Tillé, Y. (2013). Doubly balanced spatial sampling with spreading and restitution of auxiliary totals. *Environmetrics*, 24(2), 120-131.

Examples

```
## Not run:
# Example 1
N = 10;
n = 5;
p = rep(n/N,N);
Xspread = cbind(1:N);
strata = c(1,1,1,1,1,1,2,2,2,2);
indicators = lcubestratified(p,Xspread,cbind(p),strata);
s = (1:N)[indicators==1];

## End(Not run)
```

lpm	<i>Local pivotal method (sub-optimal)</i>
-----	---

Description

Select spatially balanced samples with prescribed inclusion probabilities from a finite (large) population using a sub-optimal implementation of the local pivotal method. Euclidean distance is used in the x space.

Usage

```
lpm(prob, x, h)
```

Arguments

prob	vector of length N with inclusion probabilities
x	matrix of (standardized) auxiliary variables of N rows and q columns
h	positive integer, size of window in the list to search for nearest neighbor

Value

Returns a vector of selected indexes in 1,2,...,N. If the inclusion probabilities sum to n, where n is integer, then the sample size is fixed (n).

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(runif(N),runif(N)); # matrix of auxiliary variables
h = 100; # size of search window (for finding nearest neighbor)
s = lpm(p,X,h); # select sample
plot(X[,1],X[,2]); # plot population
points(X[s,1],X[s,2], pch=19); # plot sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
X = cbind(runif(N),runif(N)); # some artificial auxiliary variables
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = lpm(p,X,10);
  ep[s]=ep[s]+1;
}
print(ep/nrs);

## End(Not run)
```

lpm1

*Local pivotal method 1***Description**

Select spatially balanced samples with prescribed inclusion probabilities from a finite population. Euclidean distance is used in the x space.

Usage

```
lpm1(prob, x)
```

Arguments

prob	vector of length N with inclusion probabilities
x	matrix of (standardized) auxiliary variables of N rows and q columns

Value

Returns a vector of selected indexes in 1,2,...,N. If the inclusion probabilities sum to n, where n is integer, then the sample size is fixed (n).

References

Grafström, A., Lundström, N.L.P. and Schelin, L. (2012). Spatially balanced sampling through the Pivotal method. *Biometrics* 68(2), 514-520.

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(runif(N),runif(N)); # matrix of auxiliary variables
s = lpm1(p,X); # select sample
plot(X[,1],X[,2]); # plot population
points(X[s,1],X[s,2], pch=19); # plot sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
X = cbind(runif(N),runif(N)); # some artificial auxiliary variables
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = lpm1(p,X);
  ep[s]=ep[s]+1;
}
print(ep/nrs);
```

```
## End(Not run)
```

```
lpm2
```

```
Local pivotal method 2
```

Description

Select spatially balanced samples with prescribed inclusion probabilities from a finite population. Euclidean distance is used in the x space.

Usage

```
lpm2(prob, x)
```

Arguments

prob	vector of length N with inclusion probabilities
x	matrix of (standardized) auxiliary variables of N rows and q columns

Value

Returns a vector of selected indexes in $1, 2, \dots, N$. If the inclusion probabilities sum to n , where n is integer, then the sample size is fixed (n).

References

Grafström, A., Lundström, N.L.P. and Schelin, L. (2012). Spatially balanced sampling through the Pivotal method. *Biometrics* 68(2), 514-520.

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N, N); # inclusion probabilities
X = cbind(runif(N), runif(N)); # matrix of auxiliary variables
s = lpm2(p, X); # select sample
plot(X[,1], X[,2]); # plot population
points(X[s,1], X[s,2], pch=19); # plot sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
X = cbind(runif(N), runif(N)); # some artificial auxiliary variables
ep = rep(0, N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = lpm2(p, X);
```

```
    ep[s]=ep[s]+1;
  }
  print(ep/nrs);

## End(Not run)
```

lpm2_kdtree

Local Pivotal Method

Description

The local pivotal method provides a way to perform balanced sampling. This implementation replace linear searches in lpm2, with k-d trees. K-d trees are binary trees used to effectively search high dimensional spaces, and reduce the average computational complexity of lpm2 from $O(N^2)$ to $O(N \log(N))$.

Usage

```
lpm2_kdtree(  
  prob,  
  x,  
  m=40  
)
```

Arguments

prob	An array of length N such that the sum of prob is equal to the sample size, where the N is the number of rows of x.
x	A matrix of N rows and p columns, each row is assumed to be a sampling unit.
m	Max leaf size used as a terminal condition for building the k-d tree.

Value

Returns a vector of selected indexes from the matrix x. The results are identical to the lpm2 function when no ties in distance occur.

Author(s)

Jonathan Lisic

References

Lisic, J., Jonathan (2015) Parcel Level Agricultural Land Cover Prediction. (Unpublished doctoral dissertation). George Mason University, Fairfax, Virginia.

Examples

```
N <- 1000
n <- 100
x <- cbind( runif(N), runif(N))

set.seed(100)
Cprog <- proc.time()
sampled <- lpm2_kdtree( rep(n/N,N), x)
print("lpm2_kdtree running time")
print(proc.time() - Cprog)
```

probabilities

Inclusion probabilities

Description

Computes the first-order inclusion probabilities from a vector of positive numbers (for a probability proportional-to-size sampling design). This function is borrowed from the package "sampling" by Alina Matei and Yves Tillé. Licence (GPL >=2).

Usage

```
probabilities(a,n)
```

Arguments

a	vector of positive numbers
n	sample size

Examples

```
## Not run:
#####
## Example
#####
# a vector of positive numbers
a=1:20
# computation of the inclusion probabilities for a sample size n=12
pik=probabilities(a,12)
pik

## End(Not run)
```

rpm	<i>Random pivotal method</i>
-----	------------------------------

Description

Select samples with prescribed inclusion probabilities from a finite population. This design has high entropy. In each of the (at most) N steps, two undecided units are selected at random to compete.

Usage

```
rpm(prob)
```

Arguments

prob vector of length N with inclusion probabilities

Value

Returns a vector of selected indexes in $1,2,\dots,N$. If the inclusion probabilities sum to n , where n is integer, then the sample size is fixed (n).

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 100; # population size
n = 10; # sample size
p = rep(n/N,N); # inclusion probabilities
s = rpm(p); # select sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = rpm(p);
  ep[s]=ep[s]+1;
}
print(ep/nrs);

## End(Not run)
```

sb	<i>Spatial balance</i>
----	------------------------

Description

Calculates spatial balance of a sample subject to inclusion probabilities and auxiliary space

Usage

```
sb(p, x, s)
```

Arguments

p	vector of length N with inclusion probabilities
x	matrix of (standardized) auxiliary variables of N rows and q columns
s	the sample, vector of length n

Value

Number, the spatial balance

References

Grafström, A., Lundström, N.L.P. and Schelin, L. (2012). Spatially balanced sampling through the Pivotal method. *Biometrics* 68(2), 514-520.

Examples

```
## Not run:
# check spatial balance
set.seed(1234567);
N = 500;
n = 70;
p = rep(n/N,N);
X = cbind(runif(N),runif(N));

# select lpm1 sample
s = lpm1(p,X);
# calculate balance
B = sb(p,X,s);

## End(Not run)
```

 scps

Spatially correlated Poisson sampling

Description

Select spatially balanced samples with prescribed inclusion probabilities from a finite population. This implementation uses the maximal weight strategy and Euclidean distance.

Usage

```
scps(prob, x)
```

Arguments

prob	vector of length N with inclusion probabilities
x	matrix of (standardized) auxiliary variables of N rows and q columns

Value

Returns a vector of selected indexes in 1,2,...,N. If the inclusion probabilities sum to n, where n is integer, then the sample size is fixed (n).

References

Grafström, A. (2012). Spatially correlated Poisson sampling. *Journal of Statistical Planning and Inference*, 142(1), 139-147.

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 1000; # population size
n = 100; # sample size
p = rep(n/N,N); # inclusion probabilities
X = cbind(runif(N),runif(N)); # matrix of auxiliary variables
s = scps(p,X); # select sample
plot(X[,1],X[,2]); # plot population
points(X[s,1],X[s,2], pch=19); # plot sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
X = cbind(runif(N),runif(N)); # some artificial auxiliary variables
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = scps(p,X);
  ep[s]=ep[s]+1;
}
print(ep/nrs);
```

```
## End(Not run)
```

```
spm          Sequential pivotal method (also known as ordered pivotal sampling
and Deville's systematic sampling)
```

Description

Select samples with prescribed inclusion probabilities from a finite population. The resulting samples are well spread in the list (similar to systematic sampling). In each of the (at most) N steps, two undecided units with smallest index are selected to compete.

Usage

```
spm(prob)
```

Arguments

```
prob          vector of length  $N$  with inclusion probabilities
```

Value

Returns a vector of selected indexes in $1,2,\dots,N$. If the inclusion probabilities sum to n , where n is integer, then the sample size is fixed (n).

References

Deville, J.-C. and Tillé, Y. (1998). Unequal probability sampling without replacement through a splitting method. *Biometrika* 85, 89-101.

Chauvet, G. (2012). On a characterization of ordered pivotal sampling. *Bernoulli*, 18(4), 1320-1340.

Examples

```
## Not run:
# Example 1
set.seed(12345);
N = 100; # population size
n = 10; # sample size
p = rep(n/N,N); # inclusion probabilities
s = spm(p); # select sample

# Example 2
# check inclusion probabilities
set.seed(12345);
p = c(0.2, 0.25, 0.35, 0.4, 0.5, 0.5, 0.55, 0.65, 0.7, 0.9); # prescribed inclusion probabilities
N = length(p); # population size
ep = rep(0,N); # empirical inclusion probabilities
nrs = 10000; # repetitions
for(i in 1:nrs){
  s = spm(p);
  ep[s]=ep[s]+1;
}
```

```
}  
print(ep/nrs);  
  
## End(Not run)
```

vsb

Variance estimator for spatially balanced sample

Description

Variance estimator of HT estimator of population total of target variable y

Usage

```
vsb(probs,ys,xs)
```

Arguments

probs	vector of length n (sample) with inclusion probabilities
ys	vector of target variable y of length n (sample)
xs	matrix of (standardized) auxiliary variables of n rows (sample) and q columns

Value

Number, the estimated variance

References

Grafström, A., and Schelin, L. (2014). How to select representative samples. *Scandinavian Journal of Statistics*.

Index

*Topic **sampling**

BalancedSampling-package, [2](#)

BalancedSampling

(BalancedSampling-package), [2](#)

BalancedSampling-package, [2](#)

cube, [4](#)

cubestratified, [6](#)

flightphase, [7](#)

landingphase, [8](#)

lcube, [9](#)

lcubeflightphase, [10](#)

lcubelandingphase, [11](#)

lcubestratified, [12](#)

lpm, [13](#)

lpm1, [14](#)

lpm2, [15](#)

lpm2_kdtree, [16](#)

probabilities, [17](#)

rpm, [18](#)

sb, [19](#)

scps, [20](#)

spm, [21](#)

vsb, [22](#)