

# Package ‘Spbsampling’

August 4, 2019

**Title** Spatially Balanced Sampling

**Version** 1.3.0

**Description** Selection of spatially balanced samples. In particular, the implemented sampling designs allow to select probability samples well spread over the population of interest, in any dimension and using any distance function (e.g. Euclidean distance, Manhattan distance). For more details, Benedetti R and Piersimoni F (2017) <doi:10.1002/bimj.201600194> and Benedetti R and Piersimoni F (2017) <arXiv:1710.09116>. The implementation has been done in C++ through the use of 'Rcpp' and 'RcppArmadillo'.

**Depends** R (>= 3.1)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**LinkingTo** Rcpp, RcppArmadillo

**Imports** Rcpp

**RoxygenNote** 6.1.1

**NeedsCompilation** yes

**Author** Francesco Pantalone [aut, cre],  
Roberto Benedetti [aut],  
Federica Piersimoni [aut]

**Maintainer** Francesco Pantalone <pantalone.fra@gmail.com>

**Repository** CRAN

**Date/Publication** 2019-08-04 12:10:02 UTC

## R topics documented:

hpwd . . . . .	2
income_emilia . . . . .	3
lucas_abruzzo . . . . .	4
pwd . . . . .	5
sbi . . . . .	6

simul1 . . . . .	7
simul2 . . . . .	8
simul3 . . . . .	9
Spbsampling . . . . .	10
stprod . . . . .	10
stsum . . . . .	11
swd . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

hpwd	<i>Heuristic Product Within Distance (Spatially Balanced Sampling Design)</i>
------	---

---

### Description

Selects spatially balanced samples through the use of Heuristic Product Within Distance design (HPWD). To have constant inclusion probabilities  $\pi_i = nsamp/N$ , where *nsamp* is sample size and *N* is population size, the distance matrix has to be standardized with function [stprod](#).

### Usage

```
hpwd(dis, nsamp, bexp = 10, nrepl = 1L)
```

### Arguments

<code>dis</code>	A distance matrix NxN that specifies how far all the pairs of units in the population are.
<code>nsamp</code>	Sample size.
<code>bexp</code>	Parameter $\beta$ for the algorithm. The higher $\beta$ is, the more the sample is going to be spread (default = 10).
<code>nrepl</code>	Number of samples to draw (default = 1).

### Details

The HPWD design generates samples approximately with the same probabilities of the [pwd](#) but with a significantly smaller number of steps. In fact, this algorithm randomly selects a sample of size *n* exactly with *n* steps, updating at each step the selection probability of not-selected units, depending on their distance from the units that were already selected in the previous steps.

### Value

Returns a matrix `nrepl x nsamp`, which contains the `nrepl` selected samples, each of them stored in a row. In particular, the *i*-th row contains all labels of units selected in the *i*-th sample.

## References

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084. <https://doi.org/10.1002/bimj.201600194>

Benedetti R, Piersimoni F (2017). Fast Selection of Spatially Balanced Samples. *arXiv*. <https://arxiv.org/abs/1710.09116>

## Examples

```
# Example 1
# Draw 1 sample of dimension 10 without constant inclusion probabilities
dis <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y))) # distance matrix
s <- hpwd(dis = dis, nsamp = 10) # drawn sample

# Example 2
# Draw 1 sample of dimension 15 with constant inclusion probabilities
# equal to nsamp/N, with N = population size
dis <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y))) # distance matrix
con <- rep(1, nrow(dis)) # vector of constraints
stand_dist <- stprod(mat = dis, vec = con) # standardized matrix
s <- hpwd(dis = stand_dist, nsamp = 15) # drawn sample

# Example 3
# Draw 2 samples of dimension 15 with constant inclusion probabilities
# equal to nsamp/N, with N = population size, and an increased level of spread, bexp = 20
dis <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y))) # distance matrix
con <- rep(0, nrow(dis)) # vector of constraints
stand_dist <- stprod(mat = dis, vec = con) # standardized matrix
s <- hpwd(dis = stand_dist, nsamp = 15, bexp = 20, nrepl = 2) # drawn samples
```

---

income\_emilia

*The income of municipalities of "Emilia Romagna".*

---

## Description

The dataset contains the total income of the municipalities in the region "Emilia Romagna", in Italy, for the year 2015. Each municipality is defined by their own ISTAT (Istituto nazionale di statistica, Italy) code and a name. For each municipality there are the following auxiliary variables: province, number of taxpayers and spatial coordinates (geographical position).

## Usage

```
income_emilia
```

**Format**

A data frame with 334 rows and 7 variables:

**municipality\_code** identification municipality code  
**municipality** name of the municipality  
**province** province of the municipality  
**numtaxpay** number of taxpayers in the municipality  
**tot\_inc** average income of the municipality  
**x\_coord** coordinate x of the municipality  
**y\_coord** coordinate y of the municipality

**Source**

The dataset is a rearrangement from the data released by the Italian Finance Department, MEF - Dipartimento delle Finanze (Italy).

---

lucas\_abruzzo

*LUCAS data for the region "Abruzzo", Italy.*

---

**Description**

The land use/cover area frame statistical survey, abbreviated as LUCAS, is a European field survey program funded and executed by Eurostat. Its objective is to set up area frame surveys for the provision of coherent and harmonised statistics on land use and land cover in the European Union (EU). Note that in LUCAS survey the concept of land is extended to inland water areas (lakes, river, coastal areas, etc.) and does not embrace uses below the earth's surface (mine deposits, subways, etc.). The LUCAS survey is a point survey, in particular the basic unit of observation is a circle with a radius of 1.5m (corresponding to an identifiable point on an orthophoto). In the classification there is a clear distinction between land cover and land use: land cover means physical cover ("material") observed at the earth's surface; land use means socio-economic function of the observed earth's surface. For each of both we assign a code to identified which type the point is. Land cover has 8 main categories, which are indicated by letter:

- A** artificial land
- B** cropland
- C** woodland
- D** shrubland
- E** grassland
- F** bareland
- G** water
- H** wetlands

Every main category has subclasses, which are indicated by the combination of the letter of the category and digits. Altogether there are 84 classes. Land use has 14 main categories. It has altogether 33 classes, which are indicated by the combination of the letter U and three digits.

**Usage**

```
lucas_abruzzo
```

**Format**

A data frame with 2699 rows and 7 variables:

**id** identified code for the unit spatial point

**prov** province

**elev** elevation of the unit spatial point, meant as the height above or below sea level

**x** coordinate x

**y** coordinate y

**lc** land cover code

**lu** land use code

**Source**

The dataset is a rearrangement of the data from LUCAS 2012 for the region "Abruzzo", Italy.  
<https://ec.europa.eu/eurostat/web/lucas/data/primary-data/2012>

---

pwd

*Product Within Distance (Spatially Balanced Sampling Design)*

---

**Description**

Selects spatially balanced samples through the use of the Product Within Distance design (PWD). To have constant inclusion probabilities  $\pi_i = nsamp/N$ , where  $nsamp$  is sample size and  $N$  is population size, the distance matrix has to be standardized with function `stprod`.

**Usage**

```
pwd(dis, nsamp, bexp = 10, nrepl = 1L, niter = 10L)
```

**Arguments**

<code>dis</code>	A distance matrix NxN that specifies how far all the pairs of units in the population are.
<code>nsamp</code>	Sample size.
<code>bexp</code>	Parameter $\beta$ for the algorithm. The higher $\beta$ is, the more the sample is going to be spread (default = 10).
<code>nrepl</code>	Number of samples to draw (default = 1).
<code>niter</code>	Number of iterations for the algorithm. More iterations are better but require more time. Usually 10 is very efficient (default = 10).

**Value**

Returns a matrix `nrepl x nsamp`, which contains the `nrepl` selected samples, each of them stored in a row. In particular, the *i*-th row contains all labels of units selected in the *i*-th sample.

**References**

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084. <https://doi.org/10.1002/bimj.201600194>

**Examples**

```
# Example 1
# Draw 1 sample of dimension 15 without constant inclusion probabilities
dis <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y))) # distance matrix
s <- pwd(dis = dis, nsamp = 15) # drawn sample

# Example 2
# Draw 1 sample of dimension 15 with constant inclusion probabilities
# equal to nsamp/N, with N = population size
dis <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y))) # distance matrix
con <- rep(0, nrow(dis)) # vector of constraints
stand_dist <- stprod(mat = dis, vec = con) # standardized matrix
s <- pwd(dis = stand_dist, nsamp = 15) # drawn sample

# Example 3
# Draw 2 samples of dimension 15 with constant inclusion probabilities
# equal to nsamp/N, with N = population size, and an increased level of spread, bexp = 20
dis <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y))) # distance matrix
con <- rep(0, nrow(dis)) # vector of constraints
stand_dist <- stprod(mat = dis, vec = con) # standardized matrix
s <- pwd(dis = stand_dist, nsamp = 15, bexp = 20, nrepl = 2) # drawn samples
```

---

sbi

*Spatial Balance Index*


---

**Description**

Computes the Spatial Balance Index (SBI), which is a measure of spatial balance of a sample. The lower it is, the better the spread.

**Usage**

```
sbi(dis, pi, s)
```

**Arguments**

dis	A distance matrix NxN that specifies how far all the pairs of units in the population are.
pi	A vector of first order inclusion probabilities of the units of the population.
s	A vector of labels of the sample.

**Details**

The SBI is based on Voronoi polygons. Given a sample  $s$ , each unit  $i$  in the sample has its own Voronoi polygon, which is composed by all population units closer to  $i$  than to any other sample unit  $j$ . Then, per each Voronoi polygon, define  $v_i$  as the sum of the inclusion probabilities of all units in the  $i$ -th Voronoi polygon. Finally, the variance of  $v_i$  is the SBI.

**Value**

Returns the Spatial Balance Index.

**References**

Stevens DL, Olsen AR (2004). Spatially Balanced Sampling of Natural Resources. *Journal of the American Statistical Association*, **99**(465), 262-278. <https://doi.org/10.1198/016214504000000250>

**Examples**

```
dis <- as.matrix(dist(cbind(simul1$x, simul1$y))) # distance matrix
con <- rep(0, nrow(dis)) # vector of constraints
stand_dist <- stprod(mat = dis, vec = con) # standardized matrix
pi <- rep(100 / nrow(dis), nrow(dis)) # vector of probabilities inclusion
s <- pwd(dis = stand_dist, nsamp = 100) # sample
sbi(dis = dis, pi = pi, s = s)
```

---

 simul1

*Simulated Population 1.*


---

**Description**

The dataset contains a simulated georeferenced population of dimension  $N = 1000$ . The coordinates are generated in the range  $[0, 1]$  as a simulated realization of a particular random point pattern: the Neyman-Scott process with Cauchy cluster kernel. The nine values for each unit are generated according to the outcome of a Gaussian stochastic process, with an intensity dependence parameter  $\rho = 0.001$  (that means low dependence) and with no spatial trend.

**Usage**

```
simul1
```

**Format**

A data frame with 1000 rows and 11 variables:

**x** coordinate x

**y** coordinate y

**z11** first value of the unit

**z12** second value of the unit

**z13** third value of the unit

**z14** fourth value of the unit

**z15** fifth value of the unit

**z16** sixth value of the unit

**z17** seventh value of the unit

**z18** eighth value of the unit

**z19** ninth value of the unit

**Source**

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084.

---

simul2

*Simulated Population 2.*

---

**Description**

The dataset contains a simulated georeferenced population of dimension  $N = 1000$ . The coordinates are generated in the range  $[0, 1]$  as a simulated realization of a particular random point pattern: the Neyman-Scott process with Cauchy cluster kernel. The nine values for each unit are generated according to the outcome of a Gaussian stochastic process, with an intensity dependence parameter  $\rho = 0.01$  (that means medium dependence) and with a spatial trend  $x_1 + x_2 + \epsilon$ .

**Usage**

simul2

**Format**

A data frame with 1000 rows and 11 variables:

**x** coordinate x

**y** coordinate y

**z21** first value of the unit

**z22** second value of the unit



- z23** third value of the unit
- z24** fourth value of the unit
- z25** fifth value of the unit
- z26** sixth value of the unit
- z27** seventh value of the unit
- z28** eighth value of the unit
- z29** ninth value of the unit

### Source

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084.

---

simul3

*Simulated Population 3.*

---

### Description

The dataset contains a simulated georeferenced population of dimension  $N = 1000$ . The coordinates are generated in the range  $[0, 1]$  as a simulated realization of a particular random point pattern: the Neyman-Scott process with Cauchy cluster kernel. The nine values for each unit are generated according to the outcome of a Gaussian stochastic process, with an intensity dependence parameter  $\rho = 0.1$  (that means high dependence) and with a spatial trend  $x_1 + x_2 + \epsilon$ .

### Usage

simul3

### Format

A data frame with 1000 rows and 11 variables:

- x** coordinate x
- y** coordinate y
- z31** first value of the unit
- z32** second value of the unit
- z33** third value of the unit
- z34** fourth value of the unit
- z35** fifth value of the unit
- z36** sixth value of the unit
- z37** seventh value of the unit
- z38** eighth value of the unit
- z39** ninth value of the unit

**Source**

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084.

---

 Spbsampling

*Spatially balanced sampling designs*


---

**Description**

Selection of spatially balanced samples. In particular, the implemented sampling designs allow to select probability samples well spread over the population of interest, in any dimension and using any distance function (e.g. Euclidean distance, Manhattan distance). The implementation has been done in C++ through the use of Rcpp and RcppArmadillo.

**Author(s)**

Francesco Pantalone, Roberto Benedetti, Federica Piersimoni

Maintainer: Francesco Pantalone <pantalone.fra@gmail.com>

**References**

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084. <https://doi.org/10.1002/bimj.201600194>

Benedetti R, Piersimoni F (2017). Fast Selection of Spatially Balanced Samples. *arXiv*. <https://arxiv.org/abs/1710.09116>

---

 stprod

*Standardize a symmetric matrix (distances) to fixed row (column) products*


---

**Description**

Standardizes a distance matrix to fixed rows and columns products. The function iteratively constrains a logarithmic transformed matrix to known products, and in order to keep the symmetry of the matrix, at each iteration performs an average with its transpose. When the known products are all equal to a constant (e.g. 0), this method provides a simple and accurate way to scale a distance matrix to a doubly stochastic matrix.

**Usage**

```
stprod(mat, vec, differ = 1e-15, niter = 1000L)
```

**Arguments**

mat	A distance matrix size NxN.
vec	A vector of row (column) constraints.
differ	A scalar with the maximum accepted difference with the constraint (default = 1e-15).
niter	An integer with the maximum number of iterations (default = 1000).

**Details**

The standardized matrix will not be affected by problems arising from units with different inclusion probabilities caused by undesired features of the spatial distribution of the population, as edge effects and/or isolated points.

**Value**

Returns a standardized distance matrix of size NxN.

**References**

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084. <https://doi.org/10.1002/bimj.201600194>

**Examples**

```
dis <- as.matrix(dist(cbind(simul1$x, simul1$y))) # distance matrix
con <- rep(0, nrow(dis)) # vector of constraints
stand_dist <- stprod(mat = dis, vec = con) # standardized matrix
```

---

stsum	<i>Standardize a symmetric matrix (distances) to fixed row (column) totals</i>
-------	--

---

**Description**

Standardizes a distance matrix to fixed rows and columns totals. The function iteratively constrains the rows sums of the matrix to known totals, and in order to keep the symmetry of the matrix, at each iteration performs an average with its transpose. When the known totals are all equal to a constant (e.g. 1), this method provides a simple and accurate way to scale a distance matrix to a doubly stochastic matrix.

**Usage**

```
stsum(mat, vec, differ = 1e-15, niter = 1000L)
```

**Arguments**

mat	A distance matrix size NxN.
vec	A vector of row (column) constraints.
differ	A scalar with the maximum accepted difference with the constraint (default = 1e-15).
niter	An integer with the maximum number of iterations (default = 1000).

**Details**

The standardized matrix will not be affected by problems arising from units with different inclusion probabilities caused by undesired features of the spatial distribution of the population, as edge effects and/or isolated points.

**Value**

Returns a standardized distance matrix of size NxN.

**References**

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084. <https://doi.org/10.1002/bimj.201600194>

**Examples**

```
dis <- as.matrix(dist(cbind(simul2$x, simul2$y))) # distance matrix
con <- rep(1, nrow(dis)) # vector of constraints
stand_dist <- stsum(mat = dis, vec = con) # standardized matrix
```

---

swd

---

*Sum Within Distance (Spatially Balanced Sampling Design)*


---

**Description**

Selects spatially balanced samples through the use of the Sum Within Distance design (SWD). To have a constant inclusion probabilities  $\pi_i = nsamp/N$ , where *nsamp* is sample size and *N* is population size, the distance matrix has to be standardized with function `stsum`.

**Usage**

```
swd(dis, nsamp, bexp = 10, nrepl = 1L, niter = 10L)
```

## Arguments

<code>dis</code>	A distance matrix $N \times N$ that specifies how far all the pairs of units in the population are.
<code>nsamp</code>	Sample size.
<code>bexp</code>	Parameter $\beta$ for the algorithm. The higher $\beta$ is, the more the sample is going to be spread.
<code>nrepl</code>	Number of samples to draw (default = 1).
<code>niter</code>	Number of iterations for the algorithm. More iterations are better but require more time. Usually 10 is very efficient (default = 10).

## Value

Returns a matrix `nrepl x nsamp`, which contains the `nrepl` selected samples, each of them stored in a row. In particular, the *i*-th row contains all labels of units selected in the *i*-th sample.

## References

Benedetti R, Piersimoni F (2017). A spatially balanced design with probability function proportional to the within sample distance. *Biometrical Journal*, **59**(5), 1067-1084. <https://doi.org/10.1002/bimj.201600194>

## Examples

```
# Example 1
# Draw 1 sample of dimension 15 without constant inclusion probabilities
dis <- as.matrix(dist(cbind(income_emilia$x_coord, income_emilia$y_coord))) # distance matrix
s <- swd(dis = dis, nsamp = 15) # drawn sample

# Example 2
# Draw 1 sample of dimension 15 with constant inclusion probabilities
# equal to nsamp/N, with N = population size
dis <- as.matrix(dist(cbind(income_emilia$x_coord, income_emilia$y_coord))) # distance matrix
con <- rep(1, nrow(dis)) # vector of constraints
stand_dist <- stsum(mat = dis, con = vec) # standardized matrix
s <- swd(dis = stand_dist, nsamp = 15) # drawn sample

# Example 3
# Draw 2 samples of dimension 15 with constant inclusion probabilities
# equal to nsamp/N, with N = population size and an increased level of spread, i.e. bexp = 20
dis <- as.matrix(dist(cbind(income_emilia$x_coord, income_emilia$y_coord))) # distance matrix
con <- rep(1, nrow(dis)) # vector of constraints
stand_dist <- stsum(mat = dis, con = vec) # standardized matrix
s <- swd(dis = stand_dist, nsamp = 15, bexp = 20, nrepl = 2) # drawn samples
```

# Index

## \*Topic **datasets**

income\_emilia, [3](#)

lucas\_abruzzo, [4](#)

simul1, [7](#)

simul2, [8](#)

simul3, [9](#)

hpwd, [2](#)

income\_emilia, [3](#)

lucas\_abruzzo, [4](#)

pwd, [2](#), [5](#)

sbi, [6](#)

simul1, [7](#)

simul2, [8](#)

simul3, [9](#)

Spbsampling, [10](#)

Spbsampling-package (Spbsampling), [10](#)

stprod, [2](#), [5](#), [10](#)

stsum, [11](#), [12](#)

swd, [12](#)