Journal of Statistical Software
*August 2022, Volume 103, Code Snippet 2.* *doi: 10.18637/jss.v103.c02*

# Spbsampling: An R Package for Spatially Balanced Sampling

**Francesco Pantalone**
University of Perugia

**Roberto Benedetti**
"G. d'Annunzio" University

**Federica Piersimoni**
Istat

**Abstract**

The basic idea underpinning the theory of *spatially balanced sampling* is that units closer to each other provide less information about a target of inference than units farther apart. Therefore, it should be desirable to select a sample well spread over the population of interest, or a *spatially balanced sample*. This situation is easily understood in, among many others, environmental, geological, biological, and agricultural surveys, where usually the main feature of the population is to be geo-referenced. Since traditional sampling designs generally do not exploit the spatial features and since it is desirable to take into account the information regarding spatial dependence, several sampling designs have been developed in order to achieve this objective. In this paper, we present the R package **Spbsampling**, which provides functions in order to perform three specific sampling designs that pursue the aforementioned purpose. In particular, these sampling designs achieve spatially balanced samples using a summary index of the distance matrix. In this sense, the applicability of the package is much wider, as a distance matrix can be defined for units according to variables different than geographical coordinates.

*Keywords*: finite population, spatial dependence, spatial balance index, MCMC.

## 1. Introduction

Recently, the availability of spatial data has strongly increased, e.g., GIS technologies and satellites orbiting around the Earth provide a huge amount of geo-referenced information. This type of data is, among others, the basis of environmental, geological, biological, and agricultural survey-based statistics. For example, we can be interested in estimating the average level of a pollutant in a specific area, the acidity of a population of lakes, the number of trees in a region for forest inventory purposes, crop average and yield estimation (Müller 2007; Benedetti, Piersimoni, and Postiglione 2015). Moreover, nowadays, National Statistical Offices can obtain the exact or estimated location of statistical units, such as businesses and households, and this information provides new possibilities for enhancing the design of surveys focused on such units. For instance, if we are interested in the average price of residential houses in a city, we can think that the spatial location of the houses has auxiliary information for the target variable. This kind of data usually implies a significant amount of spatial dependence or the presence of a spatial trend or both. Therefore, taking into account this spatial information when designing a survey could improve the efficiency of our estimates. Here, we focus on the use of this information in the *design phase* of a survey, which is the phase where we select a sample in order to collect the data. In the literature, several sampling designs have been developed to fulfill this need, which we refer to as *spatial sampling designs* (Benedetti *et al.* 2015). Among them, *spatially balanced sampling designs* achieve samples well spread over the population of interest, or *spatially balanced samples*. The first example of a spatially balanced sample is the one obtained using the generalized randomized tessellation stratified method (Stevens and Olsen 2004), which is based on a function that maps the two-dimensional space onto a one-dimensional space. More recently, spatially correlated Poisson sampling (Grafström 2012) and the local pivotal method (Grafström, , Lundström, and Schelin 2012) have been introduced, that are adaptations of correlated Poisson sampling (Bondesson and Thorburn 2008) and the pivotal method (Deville and Tillè 2000), respectively. Balanced acceptance sampling (Robertson, Brown, Mcdonald, and Jaksons 2013) uses a random-start Halton sequence while the product within distance method (Benedetti and Piersimoni 2017b) is based on a summary index of a distance matrix. For a detailed review, see Benedetti *et al.* (2015) and Benedetti, Piersimoni, and Postiglione (2017b). Spatially balanced samples are desirable because they yield more efficient estimates than estimates obtained from samples lacking spatial balance (Benedetti *et al.* 2017b). A problem that usually arises in real surveys with spatially unbalanced data in the presence of spatial autocorrelation is the problem of "pseudo-replicates" in the samples (Hurlbert 1984; Heffner, Butler, and Reilly 1996).

In this paper we present package **Spbsampling** (Pantalone, Benedetti, and Piersimoni 2022) for the R environment for statistical computing and graphics (R Core Team 2022) which is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=Spbsampling`. The package implements three spatially balanced sampling designs and some related functions. The package is based on Benedetti and Piersimoni (2017a) and Benedetti and Piersimoni (2017b), where these sampling designs are introduced and illustrated in more detail. These methods use a distance matrix of the population as synthesis of the spatial information. Such a matrix contains the distances between all pairs of units and can be computed by any distance function. The target of these methods is to achieve a sample that is spread according to these distances. In order to do that, an iterative procedure is put in place, where at each step a candidate sample is updated according to an acceptance rule that is based on a summary index of the distance matrix and promotes spread samples over non-spread samples. Since these methods could be very computationally intensive, particularly if the target population is very large, in order to boost the performance of the algorithms the implementation has been done in C++, along with the **Armadillo** library (Sanderson and Curtin 2016), and integrated in the R environment through the use of the R packages **Rcpp** (Eddelbuettel and Balamuta 2017) and **RcppArmadillo** (Eddelbuettel and Sanderson 2014). Nevertheless, in case of large grid data the computational burden could be significant. One possible solution could be to resample the raster data to a coarser scale in order to reduce the population size, which in turn eases the computation.

The paper is organized as follows. Section 2 introduces spatially balanced sampling designs; Section 3 introduces the methods implemented in the **Spbsampling** package; Section 4 provides an overview of the package, and Section 5 illustrates the usage of the package through examples based on real data. Finally, Section 6 provides conclusions and future developments.

## 2. Spatially balanced sampling

Suppose we have a target finite population, $U = \{1, \ldots, N\}$, with a set of $k$ auxiliary variables known for every unit in the population, $\mathbf{x}_i = \{x_{i1}, \ldots, x_{ik}\}$, and a set of coordinates $\mathbf{h}_i = \{h_{i1}, \ldots, h_{ih}\}$, where usually $h = 2$. Denote with $\mathcal{S}$ the set of all possible subsets of $U$, which has cardinality $2^N$. A sample without replacement is an element $s \in \mathcal{S}$. A sampling design is a probability distribution on $\mathcal{S}$ such that

$$p(s) \geq 0 \text{ and } \sum_{s \in \mathcal{S}} p(s) = 1.$$

The support of the sampling design is the set $\{s \in \mathcal{S} : p(s) > 0\} \subset \mathcal{S}$. The probability of selecting unit $i$ is called first-order inclusion probability, denoted by $\pi_i$, and given by

$$\pi_i = \sum_{s \ni i} p(s),$$

while the probability of selecting unit $i$ and $j$ in the same sample is called second-order inclusion probability, denoted by $\pi_{ij}$, and given by

$$\pi_{ij} = \sum_{s \supset \{i,j\}} p(s).$$

In the *design-based* approach the variable of interest $y$ is considered deterministic (and not as a realization of a stochastic process as in the *model-based* approach; Brus and DeGruijter 1993). The only source of randomness comes from the selection of the sample $s$ from all possible samples. In this framework, we can estimate the total of $y$, $t_y = \sum_{i=1}^{N} y_i$, through the Horvitz-Thompson estimator (Horvitz and Thompson 1952)

$$t_{y,HT} = \sum_{i \in s} \frac{y_i}{\pi_i},$$

which has the property to be unbiased with respect to the design when $\pi_i > 0$ for all $i \in U$. The variance of $t_{y,HT}$ is given by

$$V(t_{y,HT}) = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{y_i}{\pi_i} \frac{y_j}{\pi_j} (\pi_{ij} - \pi_i \pi_j),$$

where $\pi_{ij} = \pi_{ii}$ if $i = j$, and it can be estimated by

$$\hat{V}(t_{y,HT}) = \sum_{i \in s} \sum_{j \in s} \frac{y_i}{\pi_i} \frac{y_j}{\pi_j} \frac{(\pi_{ij} - \pi_i \pi_j)}{\pi_{ij}}. \quad (1)$$

The idea behind spatially balanced sampling is that the population units that are closer provide less information about a population parameter than units that are farther apart.

Therefore, during the design of a sample, a good strategy would be to select a sample well spread over the population of interest, which is referred to in the literature as *spatially balanced sample*. More technically, a well spread sample has a number of selected units on every part of the study region close to what it is expected on average. For example, if we divide our region of interest in $h$ strata and assign equal inclusion probabilities to the units, then we expect that a well spread sample has $1/h$ units from each stratum (Grafström and Lundström 2013). In this way, it is possible to capture the spatial heterogeneity and exploit the spatial information in the population. Therefore, we are interested in selecting from the given population $U$ a probability sample that is well spread.

In order to analyze this scenario, we can use a model for spatial dependence and take insights from it. In particular, we can assume that a linear model holds for $y_i$, given the known auxiliaries $\mathbf{x}_i$, for each unit of $U$, that is:

$$\begin{cases} y_i = \mathbf{x}_i^\top \boldsymbol{\beta} + \epsilon_i \\ E_m\left(\epsilon_i\right) = 0 \\ C_m\left(\epsilon_i, \epsilon_j\right) = \sigma_i \sigma_j \rho_{ij} \end{cases} \tag{2}$$

where $E_m$ and $C_m$ denote expectation and covariance with respect to the model, respectively, $\boldsymbol{\beta}$ is a vector of regression coefficients, $\epsilon_i$ is a zero-mean random variable with variance $\sigma_i^2$, and $\rho_{ij}$ is an autocorrelation parameter for $i \neq j$ and such that $\rho_{ij} = 1$ if $i = j$. This model, through the use of the *anticipated variance* (AV; Isaki and Fuller 1982), can help us understand how to take the spatial information into account at the design phase. The AV for a generic parameter of interest $\theta$ and a generic estimator $\hat{\theta}$ is given by

$$AV(\hat{\theta} - \theta) = E_m[E[(\hat{\theta} - \theta)^2]] - [E_m[E[\hat{\theta} - \theta]]]^2,$$

where $E$ denotes expectation with respect to the sampling design. Note that this is equivalent to an expected mean squared error, but since the HT estimator is unbiased with respect to the design it is usually called *anticipated variance* in the survey literature. Under the working model (2), the AV of the HT estimator of the total $t_y$ is given by:

$$AV(t_{y,HT}) = EE_m(t_{y,HT} - t_y)^2 = E\left[\left(\sum_{i \in s} \frac{\mathbf{x}_i}{\pi_i} - \sum_{i=1}^{N} \mathbf{x}_i\right)^\top \boldsymbol{\beta}\right]^2 + \sum_{i=1}^{N}\sum_{j=1}^{N} \sigma_i \sigma_j \rho_{ij} \frac{\pi_{ij} - \pi_i \pi_j}{\pi_i \pi_j}, \tag{3}$$

see Grafström and Tillè (2013). From Equation 3 we can see that the uncertainty about the estimate can be split into two components: the estimation error on the total of the auxiliary variables and the dependence of observed units. We can reduce the first term by selecting units in such a way that the HT estimates of the total of the auxiliary variables will be equal (or approximately equal) to the known totals $\mathbf{t}_x = \sum_{i=1}^{N} \mathbf{x}_i$, that is $\sum_{i \in s} \frac{\mathbf{x}_i}{\pi_i} \approx \mathbf{t}_x$. Such a sample is called *balanced* (on the variable $\mathbf{x}$) and can be obtained by a *balanced sampling design*, such as the cube method (Deville and Tillè 2004). The second term requires an assumption about $\rho_{ij}$. If $\rho_{ij}$ decreases as the distance $d_{ij}$ increases (an assumption that reflects our previous discussion), then selecting units far apart (therefore altering the second-order inclusion probabilities $\pi_{ij}$) reduces the second component. Therefore the use of spatially balanced samples can reduce the second term. Grafström and Lundström (2013) provide some conditions on which a spatially balanced sample is also a balanced sample. In this case, the former decreases both terms of (3).

Spatially balanced sampling designs necessarily set the second order inclusion probabilities of nearby units close, or equal, to zero. Moreover, the computation of $\pi_{ij}$s is usually prohibitive. Therefore, we are not able to compute the general variance estimator (1), which requires the knowledge of the $\pi_{ij}$s. Some solutions to this problem have been proposed in the literature. Stevens and Olsen (2003) propose a local mean variance estimator, while Grafström and Schelin (2014) use a modification of that proposal. Both of these estimators do not require the knowledge of the $\pi_{ij}$s.

In order to measure the spatial balance of a sample, we resort to the approach of Stevens and Olsen (2004) and use a *spatial balance index (SBI)* based on Voronoi polygons. Given a sample $s$, we can construct a Voronoi polygon for each unit $i$ in the sample such that it includes all population units closer to $i$ than to any other sample unit $j$. Then, the SBI takes the following form:

$$\mathrm{SBI}(s) = \frac{1}{n} \sum_{i \in s} (v_i - 1)^2, \tag{4}$$

where $v_i$ is the sum of the inclusion probabilities of all units in the $i$th Voronoi polygon. Since the union of Voronoi polygons makes a partition of the population and the sample size is equal to $n$, $\sum_{i \in s} v_i = \sum_{j=1}^{N} \pi_j = n$. Hence, for any sample unit $i$ we have $E(v_i) = 1$. For a spatially balanced sample, $v_i \approx 1$ for every $i$, so that values of SBI close to zero correspond to more spread samples. Moreover, if we select $B$ samples by means of a sampling design, then $\sum_{b=1}^{B} \mathrm{SBI}_b / B$ indicates how well the sampling design produces well spread samples.

Given the above discussion, we see that the lower the SBI, the higher the spread, the lower the second term in (3). Therefore, we could be tempted to select the most spread sample in order to minimize the second term as much as possible. The problem with this approach is that it does not take into account the randomization assumption of the design-based approach, which states that the only source of randomness is due to the selection of the sample. In particular, if we select every time the most spread sample, we would have no randomization, i.e., the support of the sampling design would be composed by just one sample, and we would select it with certainty (Benedetti and Palma 1995). Hence, the design-based inference would be no longer valid.

Usually, we measure the randomness of a sampling design through the use of an entropy index, such as

$$l(s) = -\sum_{s \in \mathcal{S}} p(s) \log p(s),$$

(Shannon 1948). Note that high entropy sampling designs are more robust because they produce highly randomized samples. By robust we mean that the sampling design performs well even if the assumption on the correlation between $y$ and $\mathbf{x}$ does not hold well. For more details on the relationship between entropy and robustness, see Grafström (2010). Moreover, regarding the HT estimator, the asymptotic convergence of the estimates towards the normal distribution depends on the entropy: the higher it is, the higher the rate of convergence (Berger 1998a,b). Therefore, the main objective of a spatially balanced sampling design is, on one hand, to obtain a sample well spread over the population of interest, and, on the other hand, to keep the selection random in order not to reduce substantially the entropy. In practice, we are facing a trade-off between spreadness and entropy.

Different spatially balanced sampling designs have been proposed in the last few years and then implemented in R. The generalized random tessellation stratified (GRTS) design (Stevens and Olsen 2004) selects samples with fixed size $n$, mapping the two-dimensional spatial population into one dimension, while preserving some spatial relationships between the units. This design is implemented in the R package **spsurvey** (Kincaid, Olsen, and Weber 2022). The package also implements the local mean variance estimator of Stevens and Olsen (2003). The spatially correlated Poisson sampling (SCPS) design (Grafström 2012) is a modification of correlated Poisson sampling (Bondesson and Thorburn 2008) and is based on a list sequential criterion of random decisions, i.e., starting from a list of units, the sampling outcome is first decided for the first unit in the list, then for the second, and so on. It is implemented in the R package **BalancedSampling** (Grafström and Lisic 2022). In this package, the variance estimator of Grafström and Schelin (2014) is implemented as well. The local pivotal method (LPM; Grafström *et al.* 2012) is a derivation of the pivotal method (Deville and Tillè 2000), which consists in updating at each step the inclusion probability for two units so that the sampling outcome is decided for at least one of the two units. Two versions of this design are implemented in the R package **BalancedSampling** (Grafström and Lisic 2022). Finally, it is worth noting that a common technique used in environmental surveys is the use of stratified random sampling, where one unit is selected for each stratum. Walvoort, Brus, and De Gruijter (2010) proposed two algorithms that do not follow the spatially balanced sampling literature, but they share the same concept of spatial coverage. They are based on $k$ means cluster analysis (Hartigan 1975) of a grid of the region of interest, and the obtained partition of the grid can be used as strata in stratified random sampling. Such algorithms are implemented in the R package **spcosa** (Walvoort, Brus, and de Gruijter 2021). The main difference between this approach and spatially balanced sampling is how distances are taken into account. In the former, the distances are used in order to construct clusters, which are later used as strata; in the latter, the distances are directly taken into account when selecting units, through the second-order inclusion probabilities. Therefore, when using compact geographical strata two close units lying in two different adjacent strata have the same probabilities to be selected together as two far units lying in the same two adjacent strata. This does not generally happen in spatially balanced sampling, since the second-order inclusion probabilities are determined as functions of the distance between units (Benedetti, Piersimoni, and Postiglione 2017a).

## 3. Product within distance and sum within distance designs

In this section we focus on two sampling designs introduced by Benedetti and Piersimoni (2017b) and implemented in the **Spbsampling** package. These methods select a sample with fixed size $n$ and are based on an iterative procedure that uses a distance matrix and a summary index. The distance matrix $\mathbf{D}_U = \{d_{ij}; i = 1, \ldots, N; j = 1, \ldots, N\}$ contains the distances for all the pairs of units in the population. Such distances can be computed with any metric, in any dimension (e.g., Euclidean distance, Manhattan distance, or some similarity measure). This provides high flexibility, as it allows for different spatial frameworks characterized by different distances, and to spread the sample in a covariates space through the use of some *ad hoc* distances. When the units are points, the computation is straightforward. If not, as in the case of irregular polygons, then distances between centroids can be used. Moreover, note that when using spatial distances expressed as latitude and longitude, the coordinates need to be projected and not geographic. Let $M(\mathbf{D}_s)$ be an index of the distance matrix of a given sample $s$ that summarizes the distances of the sample units. We expect that the higher $M(\mathbf{D}_s)$, the lower the SBI. We consider

$$M_0\left(\mathbf{D}_s\right) = \prod_{i \in s} \prod_{j \neq i} \prod_{j \in s} d_{ij}, \tag{5}$$

and

$$M_1\left(\mathbf{D}_s\right) = \sum_{i \in s} \sum_{j \neq i} \sum_{j \in s} d_{ij}. \tag{6}$$

Denote with $\mathbf{s}^{(t)} = \{l_1, \ldots, l_n\}$ the configuration at iteration $t$, that is a vector of unit labels. Note that a configuration is indeed a sample. The algorithm starts from a simple random sample of dimension $n$, $\mathbf{s}^{(0)}$. Then, each iteration $t$ is composed of the following steps:

1. Select two units randomly: one inside and one outside the current configuration $\mathbf{s}^{(t)}$.
2. Define a new configuration, say $\mathbf{s}_e^{(t)}$, in such a way that units $i$ and $j$ selected in the previous step are interchanged. Therefore, $\mathbf{s}^{(t)}$ and $\mathbf{s}_e^{(t)}$ differ only for one unit.
3. Update to $\mathbf{s}_e^{(t)}$ or retain the current configuration $\mathbf{s}^{(t)}$, according to an acceptance rule. In particular, update to the new configuration $\mathbf{s}_e^{(t)}$ with probability:

$$p = \min\left[1, \left(\frac{M\left(\mathbf{D}_{s_e^{(t)}}\right)}{M\left(\mathbf{D}_{s^{(t)}}\right)}\right)^{\beta}\right], \tag{7}$$

   where $\beta$ is a known constant. Note that the higher the index $M(\mathbf{D}_{s_e^{(t)}})$, the higher the overall distance of the new configuration, the higher the probability of updating the configuration to the new state. Therefore, the acceptance rule gives more chance to stay in configurations that are relatively more spread. Moreover, the parameter $\beta \in \mathbb{R}$ regulates the amount of spread: the higher its value is, the more spread the sample will be. Hence, this parameter allows us to choose *a priori* the level of spread desired, if any. We dedicated Section 5.2 on the role of $\beta$.

4. Iterate Steps 1, 2 and 3 for $N$ times.

The algorithm runs for $t = 1, \ldots T$ iterations, for a total of $T \times N$ steps, unless at iteration $t < T$ no updates occurred, in that case the algorithm is stopped. The larger the number $T$ of iterations (and therefore the number of steps), the better the convergence of the algorithm, the slower the algorithm. A suitable value for $T$ to face this trade-off is empirically found to be 10. Indeed, we observed by means of several applications of this algorithm to real and simulated populations that it never reaches the 10th iteration.

Due to this procedure, we are able to achieve a random outcome from a multivariate probability $p\left(s\right)$ proportional to the specific index $M\left(\mathbf{D}_s\right)$ used in the acceptance rule (7), that is $p\left(s\right) = M\left(\mathbf{D}_s\right) / \sum_{s \in \mathcal{S}} M\left(\mathbf{D}_s\right)$.

In theory, any summary index can be used in (7). The sampling designs implemented in the package differ according to the index $M\left(\mathbf{D}_s\right)$ used. In particular, we refer to these sampling designs with the term PWD (*product within distance*) when we use (5), and with the term SWD (*sum within distance*) if we use (6).

Essentially, the algorithm is based on a Markov chain Monte Carlo (MCMC) scheme, where at each iteration, a new configuration is defined based on the previous one, and it is accepted according to an acceptance rule, which is the Metropolis-Hasting criterion (Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller 1953; Hastings 1970). By means of this procedure, we want to obtain independent samples from the unknown target distribution $p(s)$ with probability proportional to the index used in (7).

If we perform this procedure using the distance matrix as it is, the selection mechanism has unknown inclusion probabilities, which is a problem since we cannot compute the HT estimator. Fortunately, we can regulate the first-order inclusion probabilities by means of standardization of the distance matrix. Indeed, Benedetti and Piersimoni (2017b) found that the $\pi_i$s follow the rule

$$\hat{\pi}_i = k_1 \left( \prod_{j=1}^{N} d_{ij} \right)^{k_2}, \tag{8}$$

where $k_1$ and $k_2$ are parameters that depend mainly on $N$ and $n$, and $\hat{\pi}_i$ is an empirical estimate of $\pi_i$ given by the relative frequency of the selection of unit $i$ obtained by a Monte Carlo simulation. From (8) we can note that the $\pi_i$s depend on the $d_{ij}$s, therefore we can regulate the $\pi_i$s by means of standardization of the distance matrix. Specifically, in order to achieve equal first-order inclusion probabilities, which are equal to $n/N$, in the case of SWD we iteratively constrain the row (or column) sums of the matrix to known totals (e.g., 1), and in the case of PWD we iteratively constrain the row (or column) sums of the logarithmic transformed matrix to known totals (e.g., 0), see Benedetti and Piersimoni (2017b). Note that in order to preserve the symmetry of the matrix we perform an average with its transpose at each iteration. Moreover, we believe that a modification of the aforementioned standardization procedure can lead to a selection mechanism with unequal inclusion probabilities $\pi_i$ set *a priori*. This is a current topic of investigation.

## 3.1. A fast selection of spatially balanced samples

The two procedures described so far could have a high computational burden because they have at least a computational cost of order $N^2$. When the target population is very large or we are interested in selecting a large number of samples (for example when performing a simulation) or both, these methods could be very computationally intensive. In this situation, the use of the HPWD (*heuristic product within distance*) sampling design (Benedetti and Piersimoni 2017a) can address the problem, because it is a drawn-by-drawn algorithm that in at most $n$ steps selects a spatially balanced sample of fixed dimension $n$ through a sequence of $n$ random selections of size one with varying selection probabilities, which will be updated in each step depending on the unit selected in the previous step. Particularly, the algorithm uses a greedy heuristic, which means that for each step it always makes the local optimal choice (Cormen, Leiserson, Rivest, and Stein 2009). It starts with a random selection of a unit $i$ with equal probability. Then, at every step $t \leq n$ the selection probabilities $\pi_j^{(t)}$s for all the other units $j$ of the population are updated according to the rule

$$\pi_j^{(t)} = \frac{\pi_j^{(t-1)} \overline{d}_{ij}}{\sum_{j=1; j \neq i}^{N} \pi_j^{(t-1)} \overline{d}_{ij}} \forall j = 1 \dots N \text{ and } j \neq i, \tag{9}$$

where $i$ is the unit randomly selected in the step $t-1$, and $\overline{d}_{ij} = \phi\left(d_{ij}^{\beta}\right)$, where $\phi$ is a standardization applied to $\mathbf{D}_U$ in order to have known and fixed products by row (and column), as we have seen for the case of PWD in the previous section, and $\beta$ is a known constant and has the same role observed previously for the SWD and PWD designs. Note that $\overline{d}_{ii} = 0$ by definition, so that units already selected in the sample have $\pi_i = 0$, therefore the random selection is without replacement. The greedy heuristic used in (9) slightly resembles the acceptance rule in the PWD algorithm, and the empirical $\pi_i$s obtained by Monte Carlo simulation for the HPWD are close to the empirical $\pi_i$s obtained by Monte Carlo simulation for the PWD (Benedetti and Piersimoni 2017a).

## 4. Package Spbsampling

Package **Spbsampling** implements spatially balanced sampling designs. In particular, the implemented designs are PWD, SWD, and HPWD, which we have illustrated in the previous section, and they are implemented by `pwd()`, `swd()`, and `hpwd()`, respectively. Other functions perform distance matrix standardization and compute the spatial balance index.

The functions `pwd()` and `swd()` require the following arguments:

- `dis`: distance matrix of the target population (of dimension $N \times N$);
- `n`: sample size;
- `beta`: parameter $\beta$ for the algorithm, set by default to `10`;
- `nrepl`: number of samples to draw, set by default to `1`;
- `niter`: maximum number of iterations for the algorithm, set by default to `10`.

Both functions return a list with the object `s`, which is a matrix of dimension `nrepl`$\times$`n` that contains the `nrepl` selected samples, each of them stored in a row (in particular, the $b$th row contains all labels of units selected in the $b$th sample), and the object `iterations`, which is a vector of length `nrepl` where the $b$th element contains the number of iterations run by the algorithm in order to select the $b$th sample.

The function `hpwd()` uses the same arguments than `pwd()` and `swd()`, apart from `niter`. The output is a matrix of selected samples, as described for the object `s` of the `pwd()` and `swd()` output.

As mentioned earlier, the distance matrix standardization has important implications in these sampling designs. The functions used to perform this task are `stprod()`, designed for PWD and HPWD design, and `stsum()`, designed for SWD design. They have the same following parameters:

- `mat`: distance matrix to be standardized (of dimension $N \times N$);
- `con`: constraints, of length $N$ (because we have a constraint for each row/column of the distance matrix);
- `differ`: maximum accepted difference between the sums of the rows/columns of the distance matrix and the required constraints; by default set equal to `1e-15`;
- `niter`: maximum number of iterations; by default set equal to `1000`.

The output is a list with the objects `mat`, `iterations`, and `conv`, which are the standardized matrix, the number of iterations run by the algorithm, and the convergence reached, respectively. It is necessary to point out that the standardization of the distance matrix through the use of function `stprod()` could take some minutes, especially when the population size $N$ is large.

Finally, the function `sbi()` allows to compute the spatial balance index given by (4). The inputs required are:

- `dis`: distance matrix;
- `pi`: vector of the first order inclusion probabilities of the units in the population;
- `s`: vector of sampled units labels.

## 5. Examples

In order to illustrate the functionalities of the package, we use the following three datasets: (i) `lucas_abruzzo` from package **Spbsampling**, which contains data regarding land use/cover of the Italian region Abruzzo and provided by the European field survey program (called LUCAS, which stands for land use/cover area frame statistical survey) funded and executed by Eurostat, (ii) `meuse.grid` and (iii) `meuse` from package **sp** (Pebesma and Bivand 2005). The latter contains data collected in a flood plain of the river Meuse near the village of Stein, Netherlands, and regards topsoil heavy metal concentrations (along with a number of soil and landscape variables at the survey locations), the former is a dataset with points for a grid that covers the Meuse study area. For illustration, we use these datasets as if they were populations and select samples from them. Therefore, the population size is 2699 for the `lucas_abruzzo` dataset, 3103 for the `meuse.grid` dataset, and 155 for the `meuse` dataset. In Figure 1 the two study areas are plotted.

We proceed as follows: In Section 5.1 we provide an overview of the usage of the sampling designs using the `lucas_abruzzo` dataset, in Section 5.2 we give a brief analysis of the effect of the parameter $\beta$ in the selected samples through the use of the `meuse.grid` dataset, and finally in Section 5.3 we illustrate one scenario where we spread the sample on the covariates space with the use of `meuse` dataset.

### 5.1. Overview of the package use

The classic procedure for the selection of a sample through these designs is generally composed by:

1. Computation of the distance matrix $\mathbf{D}_U$;
2. Standardization of the distance matrix, through the function `stprod` or `stsum` according to which sampling design is used (if equal probabilities are desired);
3. Selection of the sample using the corresponding function.

We can compute the distance matrix using the function `dist()`, which is part of the base R distribution. By default the function computes the Euclidean distance and the main argument

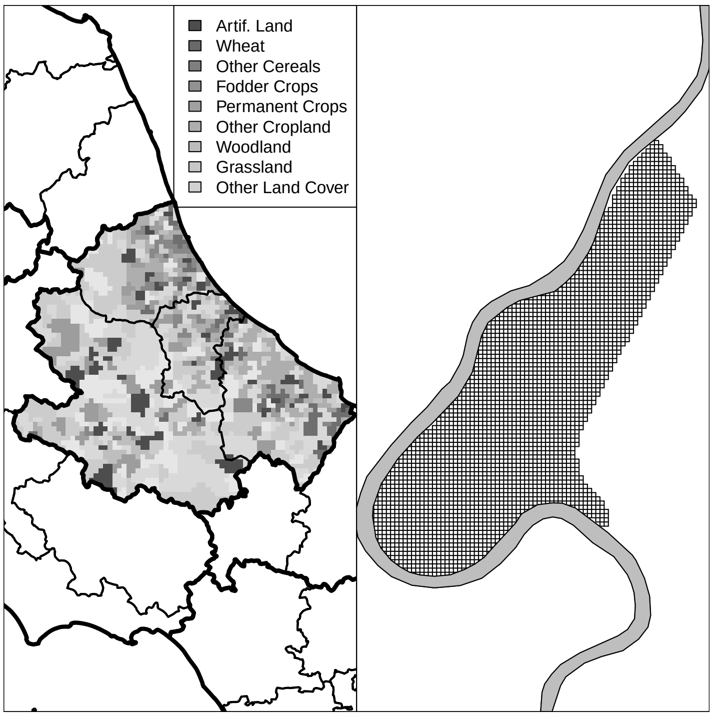

Figure 1: Population `lucas_abruzzo` on the left, `meuse` grid on the right.

required is a matrix containing the coordinates of the points. Through the parameter `method` it is possible to use a different distance function, e.g., `"manhattan"`. For our example we use the Euclidean distance and we compute the distance matrix as below.

```
R> library("Spbsampling")
R> data("lucas_abruzzo", package = "Spbsampling")
R> dis_la <- as.matrix(dist(cbind(lucas_abruzzo$x, lucas_abruzzo$y)))
```

Note that we can also compute the distance matrix by means of the `st_distance()` function of the package **sf** (Pebesma 2018). One advantage is that it calculates the distance matrix in meters unit even when the coordinates are expressed in latitude and longitude.

```
R> lucas_abruzzo_sf <- st_as_sf(lucas_abruzzo, coords = c("x", "y"))
R> dis_la_sf <- st_distance(lucas_abruzzo_sf)
```

## *Product within distance (PWD)*

Assume we want inclusion probabilities equal to $n/N$. The first step is the standardization of the distance matrix. When using the PWD design we need to constrain the row (or column) sums of the logarithmic transformed matrix to a known constant (e.g., 0). The function `stprod` is specifically designed for this, and we perform it with constraints equal to 0, through the following code:

```
R> con <- rep(0, nrow(dis_la))
R> stand_dist_la_pwd <- stprod(mat = dis_la, con = con)$mat
```

Now, since we have the standardized matrix, we perform the selection of one sample of dimension 10 by means of `pwd()`. The call is:

```
R> set.seed(42)
R> s_pwd_la <- pwd(dis = stand_dist_la_pwd, n = 10)$s
R> s_pwd_la

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]  659 1362 1951 2244  388  178 1120 1833 1369  2342
```

In the output we have all the labels of the selected units. Note that if we need the spatial coordinates of the selected points the following code can be used.

```
R> lucas_abruzzo[s_pwd_la[1, ], c("x", "y")]

            x       y
41251 1914000 4710000
42950 1856000 4686000
44101 1924000 4666000
44854 1866000 4654000
40374 1862000 4724000
39545 1890000 4736000
42419 1894000 4690000
43919 1964000 4668000
42968 1946000 4686000
45170 1898000 4654000
```

As last step, we compute the SBI with the use of the function `sbi()`. We need the inclusion probabilities, which are all equal to $n/N$ since we have standardized the matrix, and the labels of the selected units. Hence, the code is:

```
R> pi <- rep(10 / nrow(lucas_abruzzo), nrow(lucas_abruzzo))
R> sbi(dis = dis_la, pi = pi, s = s_pwd_la[1, ])

[1] 0.02130354
```

Recall that the lower the SBI, the higher the spread in the sample. We plot the obtained sample in Figure 2.

If we want more than one sample, we can use the parameter `nrepl`. For example, selecting two samples instead of one can be done using

```
R> set.seed(42)
R> s2_pwd_la <- pwd(dis = stand_dist_la_pwd, n = 10, nrepl = 2)$s
R> s2_pwd_la

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]  659 1362 1951 2244  388  178 1120 1833 1369  2342
[2,] 1061 2073 2511  248 1594 1340  229  552 1232  2637
```

Noting that the replications are independent, in the first row we have the labels of the units selected in the first sample, and in the second row we have the labels of the units selected in the second sample.

If dependent replications are of interest, e.g., two samples with an overlapping portion of units need to be selected, then coordination of spatially balanced samples needs to be employed. This means that the $\pi_i$s are controlled for both the units in each sample and between the samples. Toward this end, an interesting proposal based on the logic of permanent random numbers was suggested by Grafström and Matei (2018). Moreover, the PWD algorithm can be adapted to handle this setting. Indeed, it should be enough to define as starting sample of the algorithm the configuration we want to coordinate with, and that the algorithm keeps swapping units until a prefixed rate of overlapped units has been achieved. This procedure could allow to control dependence between and within replications, and it could be useful in a potentially appealing application such as cross-validation, where usually a control of this aspect is needed.

### *Sum within distance (SWD)*

As in the previous example, we select one sample of dimension 10. The distance matrix is the same computed before, so we need only to obtain the standardized one, this time keeping in mind the different sampling design. That means that we have to constrain the row (or column) sums of the matrix to a known constant (e.g., 1), using function `stsum()` and constraints equal to 1.

```
R> con <- rep(1, nrow(dis_la))
R> stand_dist_la_swd <- stsum(mat = dis_la, con = con)$mat
```

We use the function `swd()` to select the sample

```
R> set.seed(42)
R> s_swd_la <- swd(dis = stand_dist_la_swd, n = 10)$s
R> s_swd_la
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,] 2061 1020  655 2099 1879 2679  385  134 1738   545
```

and, as before, we compute the spatial balance index using the following code

```
R> pi <- rep(10 / nrow(lucas_abruzzo), nrow(lucas_abruzzo))
R> sbi(dis = dis_la, pi = pi, s = s_swd_la[1, ])
```

```
[1] 0.096017
```

and plot the sample in Figure 2.

### *HPWD*

The usage of the function `hpwd()` follows closely that of `pwd()`. Hence, we should standardize the distance matrix with the same procedure as before. Since we have already done that, we can directly proceed to draw one sample of dimension 10 using the `stand_dist_la_pwd` matrix and the `hpwd()` function:

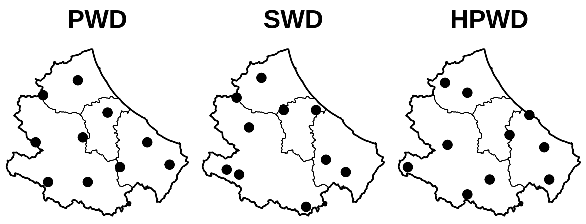

Figure 2: Samples drawn from the `lucas_abruzzo` population by means of the implemented sampling designs.

```
R> set.seed(42)
R> s_hpwd_la <- hpwd(dis = stand_dist_la_pwd, n = 10)
R> s_hpwd_la

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,] 2470 1082  206 2252 1298  352 2164  636 1913  1352
```

The SBI index is computed by

```
R> pi <- rep(10 / nrow(lucas_abruzzo), nrow(lucas_abruzzo))
R> sbi(dis = dis_la, pi = pi, s = s_hpwd_la[1, ])

[1] 0.1814763
```

and the sample is plotted in Figure 2.

*Computing time*

As previously discussed, the standardization matrix is an important step. Indeed, it allows us to select samples with equal first-order inclusion probabilities. Therefore, the time required to standardize the distance matrix is an aspect we should consider when planning any spatial survey. To this end we report in Table 1 the elapsed time for the standardization of distance matrices simulated with different population sizes. It can be noticed that the `stprod()` function is significantly slower than the `stsum()` function. This is due to the fact that in `stprod()` the accuracy is related to the logarithm of products, which is more difficult to reach. The differences between the two functions increase as $N$ increases. Therefore, if the target population is very huge, it is recommended to use the SWD design, which requires the `stsum()` standardization function. Note that the time needed for the selection of one sample is negligible. If simulation with large number of samples is of interest, then HPWD is better suited since it is more computationally efficient. In terms of spatial balance, PWD performs better than SWD, while HPWD provides similar results to PWD (Benedetti and Piersimoni 2017a,b).

## 5.2. The role of $\beta$

Now, we analyze the effect that the value of $\beta$ has in these sampling designs and consequently on the selected samples. Toward this end, we select from the dataset `meuse.grid` different samples, with sample size $n = 20$ and different values of $\beta \in \{1, 5, 10\}$, and then through the

| | $N = 1000$ | $N = 2000$ | $N = 3000$ | $N = 4000$ | $N = 5000$ |
|---|---|---|---|---|---|
| `stsum()` | 1.26 | 5.80 | 12.34 | 26.77 | 37.82 |
| `stprod()` | 10.47 | 64.20 | 133.31 | 326.52 | 429.30 |
| | $N = 6000$ | $N = 7000$ | $N = 8000$ | $N = 9000$ | $N = 10000$ |
| `stsum()` | 59.27 | 74.95 | 103.92 | 126.77 | 155.42 |
| `stprod()` | 711.70 | 884.69 | 1334.55 | 1517.31 | 1943.91 |

Table 1: Time expressed in seconds to standardize a distance matrix (on a 2.3Ghz Dual Core) by means of `stsum()` and `stprod()` when the population size $N$ varies.

use of SBI we inspect the spread of the samples. Plots are also displayed (with axes aspect ratio equal to 1). The role of $\beta$ is crucial in these sampling designs since it controls the spread of the selected sample; the higher $\beta$ is the more spread the sample is expected to be. Also, it is worth noting that for negative values of $\beta$ these sampling designs provide cluster samples. Since in this paper we are interested in well spread samples, we do not consider such $\beta$ values. Moreover, for $\beta = 0$ we observe that these sampling designs behave approximately like the simple random sampling method.

First, we compute the distance matrix for this population

```
R> library("sp")
R> data("meuse.grid", package = "sp")
R> dis_m <- as.matrix(dist(cbind(meuse.grid$x, meuse.grid$y)))
```

and we start with the PWD and HPWD designs because they require the same standardized distance matrix that we obtain with

```
R> con <- rep(0, nrow(meuse.grid))
R> stand_dist_m_pwd <- stprod(mat = dis_m, con = con)$mat
```

With this standardized matrix, we can perform the selection of the samples using

```
R> set.seed(42)
R> s_pwd1 <- pwd(dis = stand_dist_m_pwd, n = 20, beta = 1)$s
R> set.seed(42)
R> s_pwd5 <- pwd(dis = stand_dist_m_pwd, n = 20, beta = 5)$s
R> set.seed(42)
R> s_pwd10 <- pwd(dis = stand_dist_m_pwd, n = 20, beta = 10)$s
```

The function `sbi()` is used for computing the relative index.

```
R> pi <- rep(20 / nrow(meuse.grid), nrow(meuse.grid))
R> sbi(dis = dis_m, pi = pi, s = s_pwd1[1, ])

[1] 0.1696672

R> sbi(dis = dis_m, pi = pi, s = s_pwd5[1, ])

[1] 0.04192714
```

```
R> sbi(dis = dis_m, pi = pi, s = s_pwd10[1, ])
```

```
[1] 0.03573681
```

We can see that the higher we set $\beta$, the lower is SBI, which in turn implies that the sample is more spread. We do the same for the HPWD design:

```
R> set.seed(42)
R> s_hpwd1 <- hpwd(dis = stand_dist_m_pwd, n = 20, beta = 1)
R> set.seed(42)
R> s_hpwd5 <- hpwd(dis = stand_dist_m_pwd, n = 20, beta = 5)
R> set.seed(42)
R> s_hpwd10 <- hpwd(dis = stand_dist_m_pwd, n = 20, beta = 10)
R> pi <- rep(20 / nrow(meuse.grid), nrow(meuse.grid))
R> sbi(dis = dis_m, pi = pi, s = s_hpwd1[1, ])
```

```
[1] 0.1215474
```

```
R> sbi(dis = dis_m, pi = pi, s = s_hpwd5[1, ])
```

```
[1] 0.07120709
```

```
R> sbi(dis = dis_m, pi = pi, s = s_hpwd10[1, ])
```

```
[1] 0.04609553
```

and also in this case the behavior is the same. Finally, we turn the attention to the SWD design. We compute the standardized matrix

```
R> con <- rep(1, nrow(meuse.grid))
R> stand_dist_m_swd <- stsum(mat = dis_m, con = con)$mat
```

and the samples

```
R> set.seed(42)
R> s_swd1 <- swd(dis = stand_dist_m_swd, n = 20, beta = 1)$s
R> set.seed(42)
R> s_swd5 <- swd(dis = stand_dist_m_swd, n = 20, beta = 5)$s
R> set.seed(42)
R> s_swd10 <- swd(dis = stand_dist_m_swd, n = 20, beta = 10)$s
```

Finally, the SBI values obtained are

```
R> pi <- rep(20 / nrow(meuse.grid), nrow(meuse.grid))
R> sbi(dis = dis_m, pi = pi, s = s_swd1[1, ])
```

```
[1] 0.3359212
```

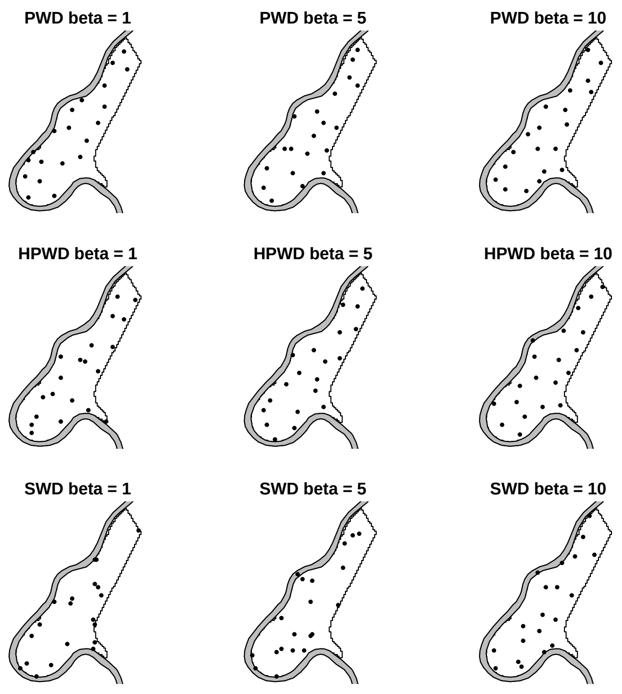

Figure 3: Samples drawn from the `meuse` population by means of the implemented sampling designs and for different $\beta$ values.

```
R> sbi(dis = dis_m, pi = pi, s = s_swd5[1, ])

[1] 0.292

R> sbi(dis = dis_m, pi = pi, s = s_swd10[1, ])

[1] 0.07203216
```

As expected, the values of `sbi()` show the same pattern previously observed. In Figure 3 we can see the selected samples. It is important to recall the trade-off between spatial balance and entropy of the sampling design mentioned in Section 2. Here, we could set extremely high values of $\beta$ in order to select the most spread samples, but the entropy of the selection would be substantially reduced, with the consequence of failing to have asymptotic normality of the HT estimator (Berger 1998a,b).

## 5.3. Spread on the covariates space

In the following example we briefly illustrate one feature of this methodology, that is the possibility to spread the sample in the covariates space. In fact, as previously described, the flexibility of the distance matrix allows to work in different frameworks, e.g., the distance matrix can be defined for units according to variables different than geographical coordinates. We consider the space generated by the covariates `copper`, `lead`, and `zinc`, which are observed levels of chemicals. Therefore, we have a triple of coordinates for each unit. The distance considered is the Euclidean distance. The distance matrix is given by

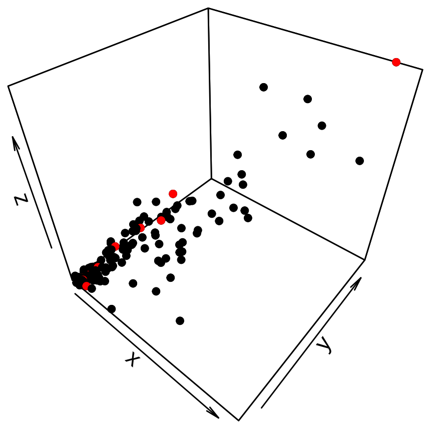

Figure 4: Sample drawn from the `meuse` population by means of PWD, with selected units in red. $x$-axis: `copper`; $y$-axis: `lead`; $z$-axis: `zinc`.

```
R> data("meuse", package = "sp")
R> dis_m_3 <- as.matrix(dist(cbind(meuse$copper, meuse$lead, meuse$zinc)))
```

We perform the PWD design. Hence, first, the distance matrix is standardized using

```
R> con <- rep(0, nrow(meuse))
R> stand_dis_m_3 <- stprod(mat = dis_m_3, con = con)$mat
```

Finally, the sample is selected by

```
R> set.seed(42)
R> s <- pwd(dis = stand_dis_m_3, n = 10)$s
```

and SBI computed through

```
R> sbi(dis = dis_m_3, pi = rep(10 / nrow(meuse), nrow(meuse)), s = s)

[1] 0.0751532
```

In Figure 4 we plot the selected sample. Note that the idea of spread the sample on the space generated by non-geographically covariates cannot be based on the spatial correlation observed in Equation 3. This calls for more investigation and is a topic for future research.

## 6. Conclusion

In this paper we presented the R package **Spbsampling**, introducing the implementation for three new spatially balanced sampling designs. For an overview of this type of methodology see Benedetti *et al.* (2015) and Benedetti *et al.* (2017b), while for the new methods implemented see Benedetti and Piersimoni (2017a) and Benedetti and Piersimoni (2017b). In spatial survey statistics, the advantages of using this kind of sampling designs are firmly established (Benedetti *et al.* 2017b). The major advantage that comes from the proposed methods is the flexibility, since we can (i) perform the algorithms over any kind of space or domain, even in dimension higher than two, (ii) use any distance function in order to face different spatial settings, and even a similarity measure, in order to spread the sample on the covariates space, and (iii) set *a priori* a parameter for the level of spread required. Moreover, Benedetti and Piersimoni (2017a) and Benedetti and Piersimoni (2017b) showed that the methods implemented in the package usually perform better in terms of spatial balance than the majority of the other spatially balanced sampling designs implemented in R. In particular, they showed that the PWD design with a value of $\beta = 10$ performs better than the GRTS (Stevens and Olsen 2004), SCPS (Grafström 2012), and LPM (Grafström *et al.* 2012) designs. A possible drawback is the need of a standardized distance matrix as input, which could take a considerable time to compute in case of a large population. Regarding the package itself, it is worth noting the implementation in C++ with the use of the **Armadillo** library. Future developments will be the implementation of functions useful to achieve unequal first order inclusion probabilities.

## References

Benedetti R, Palma D (1995). "Optimal Sampling Designs for Dependent Spatial Units." *Environmetrics*, **6**(2), 101–114. doi:10.1002/env.3170060202.

Benedetti R, Piersimoni F (2017a). "Fast Selection of Spatially Balanced Samples." arXiv:1710.09116 [stat.ME], URL https://arxiv.org/abs/1710.09116.

Benedetti R, Piersimoni F (2017b). "A Spatially Balanced Design with Probability Function Proportional to the within Sample Distance." *Biometrical Journal*, **59**(5), 1067–1084. doi:10.1002/bimj.201600194.

Benedetti R, Piersimoni F, Postiglione P (2015). *Sampling Spatial Units for Agricultural Surveys*. Springer-Verlag. doi:10.1007/978-3-662-46008-5.

Benedetti R, Piersimoni F, Postiglione P (2017a). "Alternative and Complementary Approaches to Spatially Balanced Samples." *Metron*, **75**(3), 249–264. doi:10.1007/s40300-017-0123-1.

Benedetti R, Piersimoni F, Postiglione P (2017b). "Spatially Balanced Sampling: A Review and a Reappraisal." *International Statistical Review*, **85**(3), 439–454. doi:10.1111/insr.12216.

Berger YG (1998a). "Rate of Convergence for Asymptotic Variance of the Horvitz-Thompson Estimator." *Journal of Statistical Planning and Inference*, **74**(1), 149–168. doi:10.1016/s0378-3758(98)00107-4.

Berger YG (1998b). "Rate of Convergence to Normal Distribution for the Horvitz-Thompson Estimator." *Journal of Statistical Planning and Inference*, **67**(2), 209–226. doi:10.1016/S0378-3758(97)00107-9.

Bondesson L, Thorburn D (2008). "A List Sequential Sampling Method Suitable for Real-Time Sampling." *Scandinavian Journal of Statistics*, **35**(3), 466–483. doi:10.1111/j.1467-9469.2008.00596.x.

Brus DJ, DeGruijter JJ (1993). "Design-Based Versus Model-Based Estimates of Spatial Means: Theory and Application in Environmental Soil Science." *Environmetrics*, **4**(2), 123–152. doi:10.1002/env.3170040202.

Cormen TH, Leiserson CE, Rivest RL, Stein C (2009). *Introduction to Algorithms*. MIT Press, Cambridge.

Deville JC, Tillè Y (2000). "Selection of Several Unequal Probability Samples from the Same Population." *Journal of Statistical Planning and Inference*, **86**(1), 215–227. doi:10.1016/s0378-3758(99)00164-0.

Deville JC, Tillè Y (2004). "Efficient Balanced Sampling: The Cube Method." *Biometrika*, **91**(4), 893–912. doi:10.1093/biomet/91.4.893.

Eddelbuettel D, Balamuta JJ (2017). "Extending R with C++: A Brief Introduction to **Rcpp**." *PeerJ Preprints*, **5**, e3188v1. doi:10.7287/peerj.preprints.3188v1.

Eddelbuettel D, Sanderson C (2014). "**RcppArmadillo**: Accelerating R with High-Performance C++ Linear Algebra." *Computational Statistics & Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005.

Grafström A, , Lundström NLP, Schelin L (2012). "Spatially Balanced Sampling through the Pivotal Method." *Biometrics*, **68**(2), 514–521. doi:10.1111/j.1541-0420.2011.01699.x.

Grafström A (2010). "Entropy of Unequal Probability Sampling Designs." *Statistical Methodology*, **7**(2), 84–97. doi:10.1016/j.stamet.2009.10.005.

Grafström A (2012). "Spatially Correlated Poisson Sampling." *Journal of Statistical Planning and Inference*, **142**(1), 139–147. doi:10.1016/j.jspi.2011.07.003.

Grafström A, Lisic J (2022). ***BalancedSampling**: Balanced and Spatially Balanced Sampling*. R package version 1.6.3, URL https://CRAN.R-project.org/package=BalancedSampling.

Grafström A, Lundström NLP (2013). "Why Well Spread Probability Samples are Balanced." *Open Journal of Statistics*, **3**(1), 36–41. doi:10.4236/ojs.2013.31005.

Grafström A, Matei A (2018). "Coordination of Spatially Balanced Samples." *Survey Methodology*, **44**(2), 215–238.

Grafström A, Schelin L (2014). "How to Select Representative Samples." *Scandinavian Journal of Statistics*, **41**(2), 277–290. doi:10.1111/sjos.12016.

Grafström A, Tillè Y (2013). "Doubly Balanced Spatial Sampling with Spreading and Restitution of Auxiliary Totals." *Environmetrics*, **24**(2), 120–131. doi:10.1002/env.2194.

Hartigan JA (1975). *Clustering Algorithms*. John Wiley & Sons.

Hastings WK (1970). "Monte Carlo Sampling Methods Using Markov Chains and Their Applications." *Biometrika*, **57**(1), 97–109. doi:10.1093/biomet/57.1.97.

Heffner RA, Butler MJ, Reilly CK (1996). "Pseudoreplication Revisited." *Ecology*, **77**(8), 2558–2562. doi:10.2307/2265754.

Horvitz DG, Thompson DJ (1952). "A Generalization of Sampling without Replacement from a Finite Universe." *Journal of the American Statistical Association*, **47**(260), 663–685. doi:10.1080/01621459.1952.10483446.

Hurlbert SH (1984). "Pseudoreplication and the Design of Ecological Field Experiments." *Ecological Monographs*, **54**(2), 187–211. doi:10.2307/1942661.

Isaki CT, Fuller WA (1982). "Survey Design under the Regression Superpopulation Model." *Journal of the American Statistical Association*, **77**(377), 89–96. doi:10.1080/01621459.1982.10477770.

Kincaid TM, Olsen AR, Weber MH (2022). ***spsurvey**: Spatial Survey Design and Analysis.* R package version 5.3.0, URL https://CRAN.R-project.org/package=spsurvey.

Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953). "Equation of State Calculations by Fast Computing Machines." *The Journal of Chemical Physics*, **21**(6), 1087–1092. doi:10.1063/1.1699114.

Müller WG (2007). *Collecting Spatial Data: Optimum Design of Experiments for Random Fields.* Springer-Verlag. doi:10.1007/978-3-540-31175-1.

Pantalone F, Benedetti R, Piersimoni F (2022). ***Spbsampling**: Spatially Balanced Sampling.* R package version 1.3.5, URL https://CRAN.R-project.org/package=Spbsampling.

Pebesma E (2018). "Simple Features for R: Standardized Support for Spatial Vector Data." *The R Journal*, **10**(1), 439–446. doi:10.32614/rj-2018-009.

Pebesma EJ, Bivand RS (2005). "Classes and Methods for Spatial Data in R." *R News*, **5**(2), 9–13.

R Core Team (2022). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Robertson BL, Brown JA, Mcdonald T, Jaksons P (2013). "BAS: Balanced Acceptance Sampling of Natural Resources." *Biometrics*, **69**(3), 776–784. doi:10.1111/biom.12059.

Sanderson C, Curtin R (2016). "**Armadillo**: A Template-Based C++ Library for Linear Algebra." *Journal of Open Source Software*, **1**, 26. doi:10.21105/joss.00026.

Shannon CE (1948). "A Mathematical Theory of Communication." *The Bell System Technical Journal*, **27**(3), 379–423. doi:10.1002/j.1538-7305.1948.tb01338.x.

Stevens DL, Olsen AR (2003). "Variance Estimation for Spatially Balanced Samples of Environmental Resources." *Environmetrics*, **14**(6), 593–610. doi:10.1002/env.606.

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

Walvoort D, Brus D, de Gruijter J (2021). ***spcosa**: Spatial Coverage Sampling and Random Sampling from Compact Geographical Strata.* R package version 0.4-0, URL https://CRAN.R-project.org/package=spcosa.

Walvoort DJJ, Brus DJ, De Gruijter JJ (2010). "An R Package for Spatial Coverage Sampling and Random Sampling from Compact Geographical Strata by $k$-Means." *Computers & Geosciences*, **36**(10), 1261–1267. doi:10.1016/j.cageo.2010.04.005.

## Affiliation:

Francesco Pantalone
Department of Economics
University of Perugia
Via A. Pascoli, 06123, Perugia, Italy
E-mail: francesco.pantalone@studenti.unipg.it

Roberto Benedetti
Department of Economic Studies (DEc)
"G. d'Annunzio" University
Viale Pindaro 42, 65127, Pescara, Italy
E-mail: benedett@unich.it

Federica Piersimoni
Directorate for Methodology and Statistical Process Design
Istat
Via Cesare Balbo 16, 00184, Rome, Italy
E-mail: piersimo@istat.it

*Journal of Statistical Software* https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
August 2022, Volume 103, Code Snippet 2 *Submitted:* 2020-06-10
doi:10.18637/jss.v103.c02 *Accepted:* 2022-03-02