

WORKING PAPERS

Space-filling location selection

Michela BIA
Philippe VAN KERM

CEPS/INSTEAD Working Papers are intended to make research findings available and stimulate comments and discussion. They have been approved for circulation but are to be considered preliminary. They have not been edited and have not been subject to any peer review.

The views expressed in this paper are those of the author(s) and do not necessarily reflect views of CEPS/INSTEAD. Errors and omissions are the sole responsibility of the author(s).

Space-filling location selection*

Michela Bia

CEPS/INSTEAD, Luxembourg

Philippe Van Kerm

CEPS/INSTEAD, Luxembourg

September 2013

Abstract

This note describes a Stata implementation of a space-filling location selection algorithm. It optimally selects a subset from an array of locations so that the spatial coverage of the array by the selected subset is optimized according to a geometric criterion. Such an algorithm is useful in site selection problems, but also in various non-parametric estimation procedures, e.g. to select (multivariate) knot locations in spline regression analysis.

Keywords: spatial sampling, space-filling design, site selection, multivariate knot selection, point-swapping

*This research is part of the project “*Estimation of direct and indirect causal effects using semi-parametric and non-parametric methods*” supported by the Luxembourg “Fonds National de la Recherche” cofunded under the Marie Curie Actions of the European Commission (FP7-COFUND). Van Kerm acknowledges funding for the project “*Information and Wage Inequality*” supported by the Luxembourg Fonds National de la Recherche (contract C10/LM/785657).

1 Introduction

Spatial statistics often involves geographical sampling from a set of locations (Cox et al., 1997), e.g., for networks construction for air quality monitoring (Nychka and Saltzman, 1998) or for the evaluation of exposure to environmental chemicals (Kim et al., 2010). Location selection is also useful in estimation of multivariate non-parametric or semi-parametric regression models. Estimation on a small set of grid points can reduce computational burden dramatically. Similarly, proper selection of knot location is essential to spline regression methods (Ruppert et al., 2003): poor knot locations can generate poor spline estimators not even competitive even with polynomial regression (Spiriti et al., 2012).

This article describes a Stata implementation of an algorithm for ‘space-filling’ spatial sampling.¹ The algorithm developed in Royle and Nychka (1998) selects an optimal set of ‘design points’ from a discrete set of ‘candidate points’ such that the coverage of the candidate points by the design points is optimized according to a geometric coverage criterion. The algorithm involves iterative “point-swapping” between the candidate points and the design points until no swapping can further improve the coverage of the candidate points by the design points. The coverage criteria is geometric but it is not restricted to spatial, 2-dimensional data: the procedure can be used in miscellaneous settings when optimal subsampling of multivariate data is needed.

We describe Royle and Nychka’s (1998) algorithm in Section 2 and its implementation in Stata in Section 3. We illustrate several uses of the `spacefill` package in Section 4. We show how it can be applied for generating a multidimensional regular grid of fixed size that optimally ‘covers’ a dataset.

2 Geometric coverage criterion and the point-swapping algorithm

2.1 Geometric coverage criteria

The space-filling design selection considered here is based on optimization with respect to the geometric coverage of a set of data points. We refer to data points as ‘locations’ although they are not restricted to geographic locations identified by spatial coordinates –in principle any

¹An R implementation of Royle and Nychka’s algorithm is available in Furrer et al. (2013).

uni- or multi-dimensional coordinates can be used to ‘locate’ points (see the last example in Section 4).

Following Royle and Nychka’s (1998) notation, let \mathcal{C} denote a set of N candidate locations (the ‘candidate set’). Let \mathcal{D}_n be a subset of n locations selected from \mathcal{C} . \mathcal{D}_n is a ‘design’ of size n and the locations selected in \mathcal{D}_n are ‘design points’. The geometric metric for the distance between any given location \mathbf{x} and the design \mathcal{D}_n is

$$d_p(\mathbf{x}, \mathcal{D}_n) = \left(\sum_{\mathbf{y} \in \mathcal{D}_n} \|\mathbf{x} - \mathbf{y}\|^p \right)^{\frac{1}{p}} \quad (1)$$

with $p < 0$. $d_p(\mathbf{x}, \mathcal{D}_n)$ measures how well the design \mathcal{D}_n ‘covers’ the location \mathbf{x} . When $p \rightarrow -\infty$, $d_p(\mathbf{x}, \mathcal{D}_n)$ tends to the shortest Euclidian distance between \mathbf{x} and a point in \mathcal{D}_n (Johnson et al., 1990). A design \mathcal{D}_n^* is considered to optimally cover the set \mathcal{C} for parameters p and q if it minimizes

$$C_{p,q}(\mathcal{C}, \mathcal{D}_n) = \left(\sum_{\mathbf{y} \in \mathcal{C}} d_p(\mathbf{x}, \mathcal{D}_n)^q \right)^{\frac{1}{q}} \quad (2)$$

over all possible designs \mathcal{D}_n from \mathcal{C} : the optimal design minimizes the q -power mean of the “coverages” of each candidate points.

2.2 A point-swapping algorithm

In most applications, identification of the optimal design from calculation of the coverage criterion for all possible subsets of size n from N is computationally prohibitive. Royle and Nychka (1998) propose a simple point-swapping algorithm to determine \mathcal{D}_n^* . Starting from a random initial design \mathcal{D}_n^0 , the algorithm iteratively attempts to swap a point from the design with the point from the candidate set that leads to the greatest improvement in coverage. If this tentative swap improves coverage of the candidate set by the design, the latter is updated, otherwise the swap is ignored. The process is repeated until no swap between a design point and a candidate point can improve coverage. Significant speed improvement is obtained by restricting potential swaps for a point in the design to its k nearest neighbours in the candidate set (according to (1)). See Royle and Nychka (1998) for details.

The point-swapping algorithm makes it straightforward to impose constraints on the inclusion or exclusion of specific locations: such points are considered in calculations of the

geometric criteria but excluded from any potential swap. Non random initial design points can also be used.

Critically, while the algorithm always converges to a solution, it is not guaranteed to converge to the globally optimal \mathcal{D}_n^* for any initial design, in particular when potential swaps are limited to nearest neighbours. It is therefore recommended to repeat estimation for multiple initial design sets and select the design with the best coverage across repetitions (see Section 4).

3 The `spacefill` command

The command `spacefill` performs space-filling location selection using Royle and Nychka's (1998) point-swapping algorithm. It operates on N observations from variables identifying the coordinates of the data points and returns the subset of $n < N$ observations that optimally covers the data.

`spacefill` options allow forcing inclusion or exclusion of particular observations, user-specified initial design, automatic standardization of location coordinates. When weights are specified, `spacefill` performs weighted calculation of the aggregate coverage measure (see Eq. 2). We show in Section 4 that combining weights and restrictions on candidate locations makes it easy to create an 'optimal' regular grid over a dataset.

3.1 Syntax

```
spacefill varlist [weight] [if] [in] , [ ndesign(#) design0(varlist) fixed(varname)
  exclude(varname) p(#) q(#) nruns(#) nnpoints(#) nnfrac(#) standardize
  generate(newvar) genmarker(newvar) noverbose ]
```

`fweight`, `aweight`, `pweight` and `iweight` are allowed; see [U] **11.1.6 weight – Weights**.

`varlist` and the `[if]` or `[in]` clauses identify the data from which the optimal subset is selected.

3.2 Options

`ndesign(#)` specifies n , the size of the design. Default is 4.

`design0 (varlist)` identifies a (set of) initial designs identified by non-zero *varlist*. If multiple variables are passed, one optimization is performed on each initial design and the selected design is the one with best coverage.

`fixed (varname)` identifies observations that are included in all designs when *varname* is non-zero.

`exclude (varname)` identifies observations excluded from all designs when *varname* is non-zero.

`p (#)` specifies a scalar value for the parameter p ($p = -1$ gives harmonic distance, and $p = -\infty$ gives the minimum distance). Default is -5 as recommended in Royle and Nychka (1998).

`q (#)` specifies a scalar value for the parameter q . Default is 1 (the arithmetic mean).

`nruns (#)` sets the number of independent runs to perform with random initial designs. Default is 5.

`nnpoints (#)` specifies the number of nearest neighbours considered in the point-swapping iterations. Limiting checks to nearest neighbours results in speed improvement.

`nnfrac (#)` specifies the fraction of data to consider as nearest neighbours in the point-swapping iterations. `nnfrac (#)` and `nnpoints (#)` are mutually exclusive. Default is 0.50.

`standardize` standardizes all variables in *varlist* to zero mean and unit standard deviation when calculating distances between observations.

`generate (newvarname)` specifies the names for new variable containing the locations of the best design points. If one variable is specified, it is used as a *stubname*, otherwise the number of new variable names must match the number of variables in *varlist*.

`genmarker (newvarname)` specifies the name of a new binary variable equal to 1 for observations selected in the best design and 0 otherwise.

`noverbose` suppresses display of output.

4 Examples

We illustrate application of `spacefill` on the Ozone2 dataset available in the R *'fields'* package (Furrer et al., 2013). The dataset contains air quality information in 147 locations in the US Midwest in the Summer 1987. Locations are identified by their relative latitude (`lat`) and longitude (`lon`).

Let us start with selection of an optimal design of size 10 from the 147 locations, using default values $p = -5$ and $q = 1$, candidate swaps limited to the nearest half of the locations and five runs with random starting designs.

```
. insheet using Ozone2.txt
(3 vars, 147 obs)
. spacefill lon lat, ndesign(10)
Run 1 .... (Cpq = 100.34)
Run 2 .... (Cpq = 96.92)
Run 3 ..... (Cpq = 94.19)
Run 4 .... (Cpq = 95.00)
Run 5 .. (Cpq = 95.19)
. return list
scalars:
      r(q) = 1
      r(p) = -5
      r(nn) = 69
      r(Cpq) = 94.19164847896585
r(nexcluded) = 0
r(nfixed) = 0
r(ndesign) = 10
r(N) = 147
macros:
      r(varlist) : "lon lat"
matrices:
      r(Best_Design) : 10 x 2
. matrix list r(Best_Design)
r(Best_Design) [10,2]
      lon      lat
r1 -87.752998   41.855
r2 -90.160004   38.612
r3 -85.841003   39.935001
r4 -87.57       38.021
r5 -91.662003   41.992001
r6 -84.476997   39.106998
r7 -85.578003   38.137001
r8 -85.671997   42.985001
r9 -83.403      42.388
r10 -88.283997  43.333
```

Notice that the first run leads to a somewhat higher aggregate distance to the design points ($Cpq=100.34$) than the other runs. This stresses the importance of multiple starting designs. Figure 1 shows the selected locations in the best design (achieved at Run 3 where $Cpq=94.19$).

Speed improvements can be achieved by restricting potential swaps to a smaller number of nearest neighbours. Limiting search to 25 nearest neighbours (against 69 –the default one half

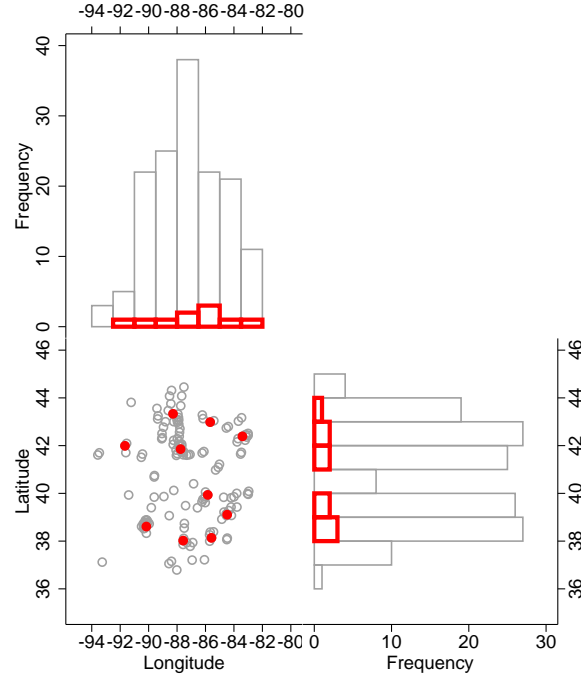


Figure 1. Scatter plot and histogram of longitude and latitude for all 147 locations (grey histograms and grey hollow circles) and 10 best design points (red thick green histograms and red solid circles) with $p = -5$ and $q = 1$ (default).

of the locations– in the first example), our second example below runs in 6 seconds against 11 seconds for our initial example, without much loss in the coverage of the resulting design ($C_{pq}=96.59$). On the other hand, running `spacefill` with the full candidates as potential swaps runs in over 30 seconds for an optimal design with $C_{pq}=91.96$.

```
. spacefill lon lat , ndesign(10) npoints(25) genmarker(set1)
Run 1 ..... (Cpq = 117.02)
Run 2 .... (Cpq = 109.93)
Run 3 .. (Cpq = 110.99)
Run 4 .. (Cpq = 101.05)
Run 5 ..... (Cpq = 96.59)
. spacefill lon lat , ndesign(10) nnfrac(1)
Run 1 ... (Cpq = 91.96)
Run 2 .... (Cpq = 91.96)
Run 3 .. (Cpq = 91.96)
Run 4 ... (Cpq = 92.32)
Run 5 ... (Cpq = 91.96)
```

Let us now illustrate use of the `genmarker`, `fixed` and `exclude` options. `genmarker(set1)` in the previous call generated a dummy variable equal to 1 for the 10 points selected into the best design and 0 otherwise. We now specify `exclude(set1)` to derive a new design with 10 different locations and then use `fixed(set2)` to force this new design into a design of size 15.

```
. spacefill lon lat, ndesign(10) npoints(25) exclude(set1) genmarker(set2) noverbose
```

```

10 points excluded from designs (set1>0)
. spacefill lon lat, ndesign(15) npoints(25) fixed(set2) genmarker(set3) noverbose
10 fixed design points (set2>0)
. list set1 set2 set3 if set1+set2+set3>0

```

	set1	set2	set3
4.	1	0	0
10.	0	1	1
25.	1	0	0
40.	1	0	0
48.	0	1	1
55.	1	0	0
58.	0	1	1
60.	1	0	0
61.	0	1	1
63.	0	0	1
67.	0	0	1
74.	1	0	0
77.	0	0	1
80.	0	1	1
82.	0	1	1
89.	0	0	1
91.	0	1	1
97.	1	0	0
107.	0	1	1
109.	1	0	0
121.	0	1	1
125.	0	0	1
135.	0	1	1
140.	1	0	0
143.	1	0	0

The key parameters q and p of the coverage criterion can also be flexibly specified. Figure 2 illustrates three designs selected with default parameters $p = -5$ and $q = 1$ (dots), with $p = -1$ and $q = 1$ (hollowed squares), and with $p = -1$ and $q = 2$ (crosses). With $p = -5$ the distance of a location to the design is mainly determined by the distance to the closest point of the design; $p = -1$ takes into account the distance to all points in the design, leading to more central location selections. Setting $q = 2$ penalizes large distances between design and non-design points, leading to location selections more spread out towards external points. Overall, these three parameter combinations result in similar designs. Note in these examples our use of user-specified random starting designs with option `design0` to ensure comparison is made on common initial values.

```

. gen byte init1 = 1 in 1/10
(137 missing values generated)
. gen byte init2 = 1 in 11/20
(137 missing values generated)
. gen byte init3 = 1 in 21/30
(137 missing values generated)
. gen byte init4 = 1 in 31/40
(137 missing values generated)
. gen byte init5 = 1 in 41/50
(137 missing values generated)
. local options nnfrac(0.3) nruns(10) design0(init1 init2 init3 init4 init5) noverbose
. spacefill lat lon, 'options' gen(Des)

```

```
. spacefill lat lon, `options' gen(Des_BIS) p(-1) q(1)
. spacefill lat lon, `options' gen(Des_TER) p(-1) q(2)
```

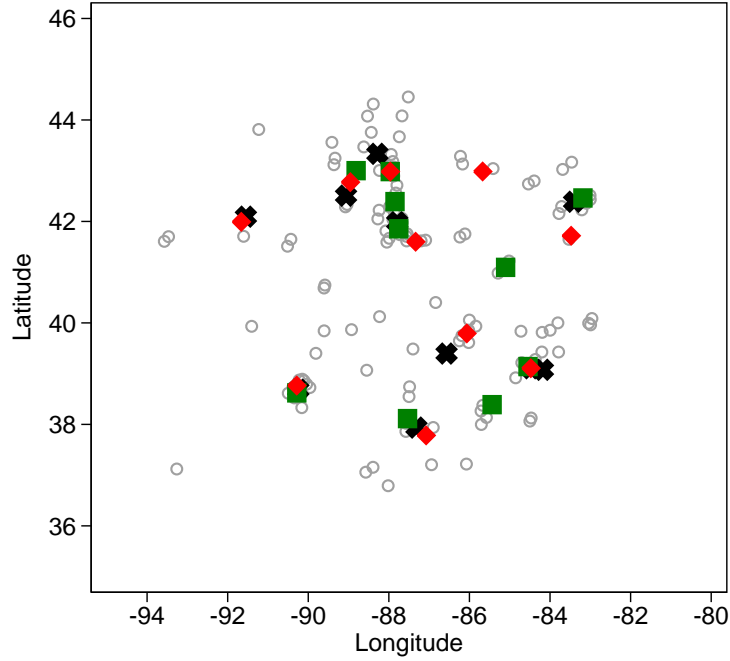


Figure 2. Scatter plot of longitude and latitude for all 147 locations (grey hollow circles) and best design points with default $p = -5$ and $q = 1$ (red dots), with $p = -1$ and $q = 1$ (green squares) and with $p = -1$ and $q = 2$ (black crosses).

By combining the `exclude` options and weights, `spacefill` can be used to find an optimal design from an external set of locations. This is particularly useful to construct a regular grid covering the data. We start by generating a dataset with a large number of candidate grid points using `range` ([R] **range**) and `fillin` ([R] **fillin**). We append this generated dataset to our locations data. Actual observations from our sample are identified by `sample==0` while the generated candidate locations on the regular grid have `sample==1`.

We can now run `spacefill` to select from the candidate grid points a smaller subset of grid points that optimally covers the actual locations. To do so, we run `spacefill` on the whole set of data points with (i) `exclude(sample)` to select points from the grid only and (ii) with `[iw=sample]` so that the aggregate distance is computed only between the design points on the grid and the actual locations. A set of 25 optimally chosen grid points from a candidate grid of 11×16 points are shown in Figure 3.

```
. clear
```

```

. set obs 16
obs was 0, now 16
. range lon -95 -80 16
. range lat 36 46 11
(5 missing values generated)
. fillin lon lat
. gen byte sample = 0
. save gridlatlon.dta , replace
file gridlatlon.dta saved
. clear
. insheet using Ozone2.txt
(3 vars, 147 obs)
. keep lat lon
. gen byte sample = 1
. append using gridlatlon
. spacefill lon lat [iw=sample], exclude(sample) ndesign(25) nnpoints(100) genmarker(subgrid1)
147 points excluded from designs (sample>0)
Run 1   ...                               (Cpq =           63.33)
Run 2   ...                               (Cpq =           63.09)
Run 3   ....                              (Cpq =           63.48)
Run 4   ....                              (Cpq =           63.13)
Run 5   ...                               (Cpq =           63.16)

```

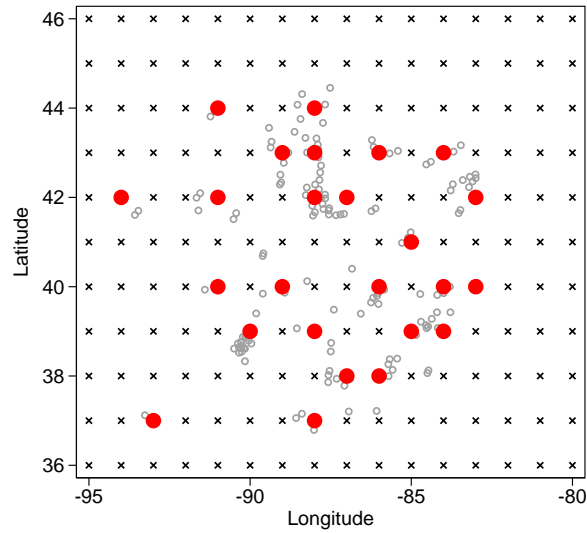


Figure 3. Actual locations (hollowed grey circles), candidate grid points (crosses), and 25 optimally selected grid points (red solid dots).

Our final example illustrates use of `spacefill` with multi-dimensional and non-spatial data taken from the *Panel Socio-Economique Liewen zu Lëtzebuerg* (PSELL3/EU-SILC) collected in 2007.² We extract information on the height, weight and wage of a random subsample of 500 women. We use `spacefill` to select a subset of 50 women with characteristics on these three variables that best ‘cover’ the sample; this subset would be used, e.g., as a set on which to run computationally intensive models. Given the different nature and scaling of the

²PSELL3/EU-SILC is a longitudinal survey on income and living conditions representative of the population residing in Luxembourg. Data are collected annually in a sample of more than 3,500 private households.

three variables, we specify the `standardize` option to compute the geometric distance criterion after standardizing the three variables to have zero mean and unit standard deviation in the sample. Figures 4 and 5 shows bivariate scatter plots of the selected design points.

```
. summarize height weight wage
```

Variable	Obs	Mean	Std. Dev.	Min	Max
height	500	165.21	6.8886	150	192
weight	500	65.368	12.80502	43	127
wage	500	2720.688	1920.047	300	10000

```
. spacefill height weight wage, ndes(50) nnfrac(0.05) generate(BH BW BWa) standardize
Run 1 ... (Cpq = 198.11)
Run 2 ..... (Cpq = 195.91)
Run 3 ..... (Cpq = 196.08)
Run 4 ..... (Cpq = 198.09)
Run 5 ... (Cpq = 195.56)
. matrix list r(Best_Design)
r(Best_Design) [50,3]
      height  weight  wage
r1      162      50   1500
r2      165      76   3700
r3      170      64   1800
r4      169      54   9510
r5      172      92   4300
r6      165      49   6968
r7      160      49   1900
r8      165      93   1500
r9      176      87   1980
r10     156      56   2000
r11     164      75   5500
r12     167      75   9000
r13     160      56    756
r14     170      75   2700
r15     178      70   1600
r16     160      64   1620
r17     179      77   6138
r18     161      52   8894
r19     180      84   8600
r20     154      85   4126
r21     150      65    771
r22     170      72   7366
r23     156      66    881
r24     178     127   3700
r25     166      58   1100
r26     182     103   1820
r27     165      54   1500
r28     170      69   1350
r29     152      49   2200
r30     168      62   2544
r31     159      88    412
r32     155      50   1500
r33     165      68   1547
r34     156      51   6000
r35     170      95   2300
r36     165      76    600
r37     168      59   4308
r38     166      61   5100
r39     170      58   7470
r40     174      67   2900
r41     155      50   3199
r42     166      99   7000
r43     177      53   1800
r44     170      58   3750
r45     168      73   1900
r46     192     103   1250
r47     170     113   3800
r48     178     101   4665
r49     153      84    900
r50     176      71   4000
```

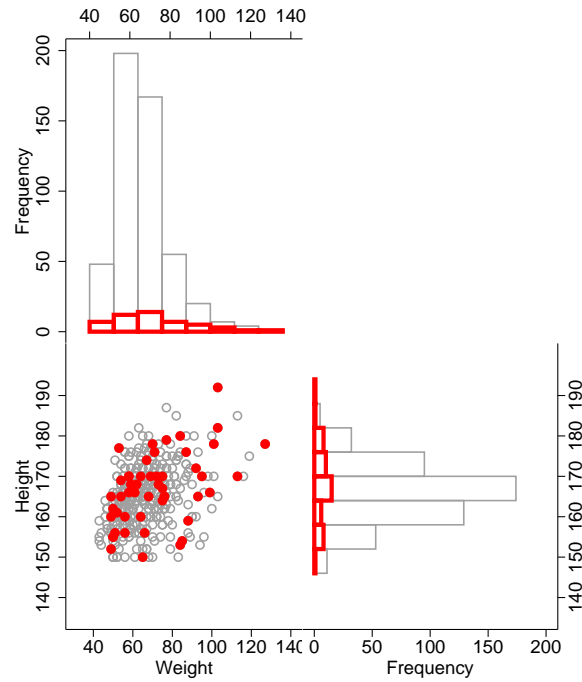


Figure 4. Scatter plot and histogram of height and weight for all data (grey histograms and grey hollow markers) and best design points (thick pink histograms and pink markers), for the standardized values of the height, weight and wage.

References

- Cox, D. D., Cox, L. H. and Ensor, K. B. (1997), ‘Spatial sampling and the environment: some issues and directions’, *Environmental and Ecological Statistics* **4**(3), 219–233.
- Furrer, R., Nychka, D. and Sain, S. (2013), *fields: Tools for spatial data*. R package version 6.7.6, 2013-04-21.
- URL:** <http://CRAN.R-project.org/package=fields>
- Johnson, M. E., Moore, L. M. and Ylvisaker, D. (1990), ‘Minimax and maximin distance designs’, *Journal of Statistical Planning and Inference* **26**(2), 131–148.
- Kim, J., Lawson, A. B., Dermott, S. M. and Aelion, C. M. (2010), ‘Bayesian spatial modeling of disease risk in relation to multivariate environmental risk fields’, *Statistical Medicine* **29**(1), 142–157.
- Nychka, D. and Saltzman, N. (1998), Design of air-quality monitoring networks, *in* D. Nychka,

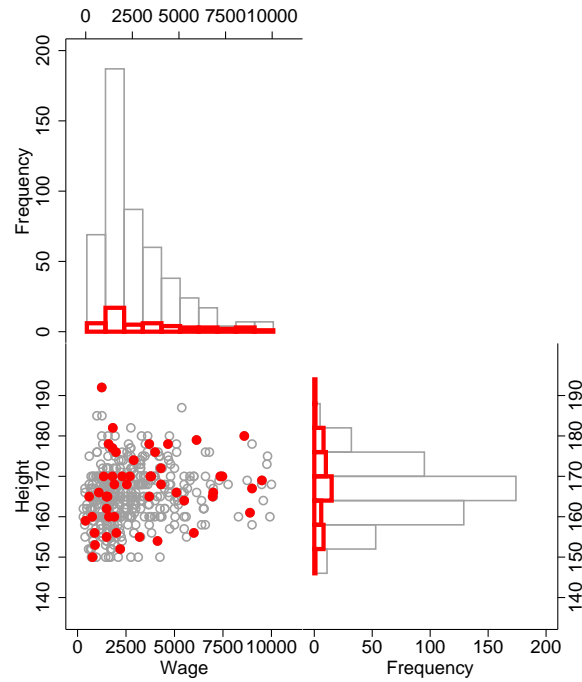


Figure 5. Scatter plot and histogram of height and wage for all data (grey histograms and grey hollow markers) and best design points (thick pink histograms and pink markers), for the standardized values of the height, weight and wage.

W. W. Piegorsch and L. H. Cox, eds, ‘Case Studies in Environmental Statistics’, Vol. 132 of *Lecture Notes in Statistics*, Springer US, pp. 51–76.

Royle, J. A. and Nychka, D. (1998), ‘An algorithm for the construction of spatial coverage designs with implementation in *splus*’, *Computers & Geosciences* **24**(5), 479–488.

Ruppert, D., Wand, M. P. and Carroll, R. J. (2003), *Semiparametric regression*, Cambridge Series in Statistics and Probabilistic Mathematics, Cambridge University Press, New York.

Spiriti, S., Eubank, R., Smith, P. and Young, D. (2012), ‘Knot selection for least squares and penalized splines’, *Journal of Statistical Computation & Simulation* **1**(1), 1–17.



3, avenue de la Fonte
L-4364 Esch-sur-Alzette
Tél.: +352 58.58.55-801
www.ceps.lu