Marburg Geography

Working Papers on
Innovation and Space

# Optimizing Distance-Based Methods for Big Data Analysis

# 09.13

Tobias Scholl and Thomas Brenner

Philipps Universität Marburg

**Impressum:**

Working Papers on Innovation and Space
Philipps-Universität Marburg

Herausgeber:

Prof. Dr. Dr. Thomas Brenner
Deutschhausstraße 10
35032 Marburg
E-Mail: thomas.brenner@staff.uni-marburg.de

Erschienen: 2013

# Optimizing Distance-Based Methods for Big Data Analysis

**Tobias Scholl**[1]

House of Logistics and Mobility (HOLM), Frankfurt and Economic Geography and Location Research, Philipps-University, Marburg.

**Thomas Brenner**

Economic Geography and Location Research, Philipps-University, Marburg.

**Abstract:**

Distance-based methods for measuring spatial concentration such as the Duranton-Overman index undergo an increasing popularity in the spatial econometrics community. However, a limiting factor for their usage is their computational complexity since both their memory requirements and running-time are in O(n2). In this paper, we present an algorithm with constant memory requirements and an improved running time, enabling the Duranton-Overman index and related distance-based methods to run big data analysis. Furthermore, we discuss the index by Scholl & Brenner (2012) whose mathematical concept allows an even faster computation for large datasets than the improved algorithm does.

**Keywords:** Spatial concentration, Duranton-Overman index, big-data analysis, MAUP, distance-based measures

**JEL Classifications:** C40, M13, R12

1 Corresponding Author: Tobias Scholl, House of Logistics and Mobility (HOLM), Frankfurt and Economic Geography and Location Research, Philipps-University, Marburg, Germany. E-Mail: tobias.scholl@nephele-idea.de.

Acknowledgements: This paper was partly written during a visiting research stay at ETH Zurich, Chair of System Design, financed by the German Academic Exchange Service (DAAD). We thank Dr. Ingo Scholtes for the fruitful discussions.

# 1 Introduction

In a recent article, Harvey Miller sees spatial science being trapped in an avalanche of an "unprecedented amount of fine-grained data on cities, transportation, economies, and societies, much of these data referenced in geo-space and time" (Miller (2010): 181). Indeed, new data sources such as location-aware technologies or point-of-sale data in combination with easy access to computational power enable new insights into spatial science. However, as Miller points out, two aspects limit our ability to dig ourselves out of this data avalanche: First, there is a lack of suitable computational methods for many research designs and second, some of the existing methods are affected by high computational requirements.

One example for the latter case is the growing interests of spatial econometrics in distance-based methods for measuring spatial concentration. While these metrics have a more longstanding tradition in disciplines such as forestry or astronomy, the work of Duranton Overman (2005) has successfully established distance-based methods in spatial econometrics. Distance-based methods circumvent the Modifiable Areal Unit Problem, a fundamental problem in spatial science (see Openshaw 1984), and thus allow for a more realistic observation of spatial concentration without an ex-ante discretization of space.

Despite the clear methodological progress of the Duranton-Overman index (DO-index henceforth), the method and similar methods such as the M-Index (Marcon & Puech 2010) since then have rarely been used in comparison to MAUP affected indices such as the Ellison-Glaeser, or the Gini index. Three points can explain this paradox: First, availability of fine-grid data is still a problem. Second, until recently, there was no statistical program available for the DO-index, so that applying it always required own programming. Third, the index shows high computational requirements that hamper or even prevent its usage for large datasets. Since point one should become more and more obsolete and two R packages haven been developed for the DO-index recently, it is worth looking at the latter point.

With respect to the existing literature, the high computational requirements of the DO-index turn out to be a crucial issue: Despite applicable data, Vitali et.al (2009) partially abandon the DO-index due to its "tremendous computational requirements" (Vitali et.al 2009: 20). Ellison et al. (2010) simplify the DO-index in several aspects in order to apply it on the whole population of manufacturing firms in the USA. Nevertheless, they state that the index "is much more computationally intensive vis-a-vis simpler discrete indices" and amount its computing time to three months for their research (Ellison et al. 2009: 5). This computational problem is also confirmed by Kosfeld et al. (2011) who summarize that the computation of some distance based methods is "not a question of hours but of days" (Kosfeld et al. 2011: 312). The computational complexity of distance-based methods arises from the simple fact that measuring spatial concentration bases on the observation of bilateral distances be-

tween single points or firms, respectively. Thus, both their running time and their memory requirements are in $O(n^2)$ where $n$ is the number of firms.

In this paper, we show that the quadratic RAM requirements are the most problematic point when running the DO-index and similar distance-based methods on huge data sets and present an improved algorithm with constant memory requirements and an improved running time. Beside the DO-index we discuss the index of Scholl & Brenner (2012) whose mathematical concept deviates from other distance-based methods in such way that it is even more suitable for large datasets than the improved algorithm of the DO-index.

The paper is structured as follows: After a short description of the general computation of distance-based methods, section 2 outlines the DO-index and its computational complexity in more details. A new algorithm for the DO-index with an enhanced running time and constant memory requirements is presented afterwards. In section 4, the index by Scholl & Brenner (2012) is discussed and benchmarked with the DO-index when running big-data analysis. The last section concludes.

# 2 Existing Algorithms

Although the existing distance-based indices differ in their calculation to some degree, the majority bases on two fundamental principles[1]: First, bilateral distances between each point pair of the observed industry are computed and the occurrence of neighborhoods at or within a distance is counted. Second, the observed distances are tested against the null hypothesis that they are the outcome of a random distribution of points, which is done by applying Monte-Carlo simulations. These two principles, computing bilateral distances and running Monte-Carlo simulations, give an intuitive classification of the computational complexity of distance-based methods: Both, the running-time and the memory requirements are quadratic due to the computation of bilateral distances and each computation is repeated a lot of times for the Monte-Carlo simulations.

Scholl & Brenner (2012) propose a rather different index that deviates from the above mentioned two principles. Their metric also bases on bilateral distances but instead of counting neighborhoods at or within a certain distance, they compute cluster values for each firm and use non-parametric methods instead of Monte-Carlo simulations in order to tests for the null hypothesis of a random distribution. Despite these differences, their index meets all five criteria for a spatial statistical test of localization, proposed by Duranton and Overman (2005) and leads to similar global outcomes (see Scholl & Brenner 2012).

Given the fact that up to now the DO-index is the most established distance-based method in

[1] See Marcon & Puech (2012) for a detailed discussion and a synopsis of the existing distance-based indices.

spatial econometrics, we will discuss its mathematical concept and its computational requirements in more details in the following sections. In section 2.2, an improved algorithm with constant memory requirements is presented whose core concept however is not limited to the DO-index but is applicable to the majority of distance-based metrics. Due to its different computation, the index by Scholl & Brenner is discussed separately afterwards.

Before we present the algorithms, we briefly explain the notations and concepts that we use:

- The *time complexity* of an algorithm is simply the time that it needs to execute and can be described by the *O* notation.
- The *space complexity* of an algorithm stands for the needed amount of memory space, also represented by the *O* notation. In the paper, all statements on the space complexity apply for the computation only, i.e. we use a model of a 3-tape Turing machine with a read-only-, write-only- and a work tape.
- *O* (Big Oh) represents the upper bound of an algorithm. For instance, if the time complexity of an algorithm is $O(n^2)$ its running time grows asymptotically no faster than $n^2$ where *n* stands for the size of the input.

## 2.1 The DO-index

The basic idea of the DO-index is to check whether the number of neighborhoods at a specific distance between firms is significantly higher or lower than expected by random. To this end, a smoothed density over all observed distances, expressed by the term *K(d)*, is used. The first step to compute *K(d)*-values is to build the geographical distances between all possible pairs of firms and compute a kernel density estimation of the observed values at a given number of distance intervals. Duranton & Overman (2005) use a distance interval of 1km and consider only those distances that are below the median distance between firms in the area under investigation. Hence the formula is:

$$\widehat{K}(d) = \frac{1}{n(n-1)h} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} f\left(\frac{d - d_{i,j}}{h}\right), \qquad (1)$$

where h is the optimal bandwidth[2] and f stand for the kernel function (Duranton & Overman 2005:1083).

The solid line in Figure 1 plots the *K(d)*-values for an illustrative industry. The dashed and dotted lines refer to the local and global confidence intervals that will be explained now.

[2] Optimal bandwidth: $1.06sn^{-0,2}$, where *n* is the observed number and *s* is the standard deviation (Klier & McMillen 2006: 12).

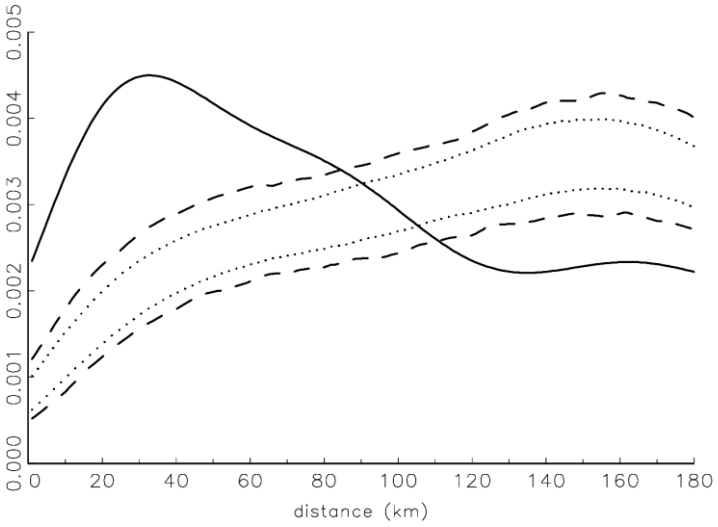


Figure 1: *K*-density, local confidence intervals and global confidence bands for an illustrative industry. Source: Duranton & Overman 2005.

To control whether the *K(d)*-values of an observed industry show significant spatial concentration or dispersion at specific distances, confidence bands are needed that are constructed by a Monte-Carlo approach: Let *N* be the number of firms in the industry under investigation. We draw *N* firms out of the population of all firms in the area of investigation. These drawn firms represent a random industry localization, whose bilateral distances are computed and kernel smoothed. The step of drawing random firms and computing their bilateral distances is done 1000 times. For the 1000 benchmark simulations the values of the kernel density estimation are sorted in ascending order for each distance interval. The 5-th and 95-th percentile are selected , whereby we obtain a lower 5% and an upper 5% confidence interval that Duranton and Overman call local confidence intervals or $\overline{K_A}(d)$ and $\underline{K_A}(d)$ respectively, (dotted lines in Figure 1) (Duranton & Overman 2005:1086). The industry in Figure 1 exceeds the upper local confidence interval between 0 and 90 km, stating that this industry shows significantly more neighborhoods at small distances.

Due to the fact that the *K(d)*-function is built separately for each km, an industry will probably hit the local bands once. In order to test whether an industry is generally more concentrated, Duranton and Overman propose the computation of global confidence intervals. By means of the 1000 simulations, the upper global confidence interval $\overline{\overline{K}}(d)$ is computed in such way that only 5 % of the thousand simulations hit the global confidence interval; the same is performed for the lower interval (Duranton & Overman 2005:1087).

## 2.2 Analysis of the computational complexity

Listing 1: Existing algorithm of the DO-index

```
/* INPUT */
1:  double[] lat; //array with the latitude of each firm
2:  double[] long; // array with the longitude of each firm
3:  stepInterval =1 //discretizise distances in 1 km distance intervals
4:  medianDistance=180; //define the median of bilateral distance
/* CALCULATION */
5:   mat = square_matrix(lat, long) // function that computes the square matrix of bilateral distances
6:   mat = lower.tri(mat) // get lower triangle of the distance matrix
7:   smooth=kde(mat, stepInterval, medianDistance) // kernel smoothing the observed distances
```

In the following, we will discuss the computational complexity of the DO-index by means of the recently published R-packages *McSpatial* by Daniel McMillen (2012) and *dbmss* by Marcon et al. (2012). In order to focus on complexity aspects, only the core steps of the algorithms, computing bilateral distances and kernel density estimation, are presented.

We start with the position of all firms, given by two arrays with the firms' latitude and longitude; the median distance and the distance interval are set to 180 km and 1 km. In line 5, the square matrix of bilateral distances is built by vector multiplication of the latitude and longitude array. This is the crucial step in the algorithm that leads to its quadratic space and time complexity. We can specify the space and time complexity of computing a matrix of bilateral distances to ($n$*($n$-1)/2) as there are only $n$*($n$-1)/2 unique bilateral distances since $d_{i,j} = d_{j,i}$. However, concerning the implementation of the code in R, both time and space complexity are quadratic since the full square matrix is computed in step 5 first before cutting off the lower triangle in step 6. This intermediate step can be explained by the fast matrices algorithms in R that are actually more efficient than computing the distances in loops when relying on internal R computation only. The last point of the algorithm is kernel smoothing the observed distances at the given number of distance intervals in step 7.

Given the fact that the time complexity of computing bilateral distances cannot be lower than ($n$*($n$-1)/2), it is worth to look at the space complexity. Concerning the index' practical applicability, this is even the more crucial point, since each distance is a double value and is stored in the RAM. Additionally, R internally treats all variables as a vector object with a total size of 48 bytes what leads to an enormous demand for RAM when observing large industries (Table 1).

| Industry | Number of firms | Required RAM in GB |
|---|---|---|
| French Chemical, rubber and plastic industry[3] | 2,158 | 0.20 |
| German Automobile Industry[4] | 82,637 | 305.27 |
| European Logistics Industry[5] | ~700,000 | 21,904.70 |

Table 1: RAM requirements of bilateral distance square-matrices of illustrative industries in R.

# 3 An algorithm with constant memory requirements

To start with the basic idea of our algorithm it is important to remember that the actual focus of the DO-index is giving information about dispersion or concentration at specific distance intervals. Thus we are actually not interested in the bilateral distances but in their occurrence at these intervals. This suggests that condensing the storage of data to the distance intervals should be a good strategy for reducing computational complexity. This is actually the way our algorithm operates and this also leads to its constant memory requirements as they are limited by the number of intervals.

Using the data of Duranton and Overman as an example, we initialize an array with the length of 180 (mean distance in UK: 180 km, step-interval $si$=1km). We loop over all firms and compute bilateral distances to all other firms in a second loop. But instead of storing the distance, we can directly update the distance-interval array, given the information of the distance. For the simple case of $si$=1, we just have to round the distance and get its integer value. This value represents the position of the distance in the distance interval, so we can increment the array at that position. By not storing the distances, we can reduce the space complexity from $O(n^2)$ to $O(di)$ where $di$ stands for the number of distance intervals. The time complexity of the algorithm is $(n*(n-1)/2)$ since the two loops compute only unique bilateral distances. Since R is slow concerning loops, we use the R-package *inline* to run the loops directly in C what notably enhances the computational performance in practice.

For testing the algorithms, we use a dual-core PC with 2x2 GH, 4 GB RAM and Ubuntu as operating system. Table 2 shows that for a set of 2000 firms, both algorithms show similar

[3] Source: Marcon & Puech (2003): 422.
[4] Source: German Federal Ministry of Economics and Technology (2010).
[5] Own calculations based on the number of firms listed in Bureau van Dijk's Amadeus database that belong to the logistics industry as defined by the European Cluster Observatory.

Listing 2: Improved algorithm with constant memory requirements

```
/* INPUT */
1:  double[] lat; //array with the latitude position of each firm
2:  double[] long; // array with the longitude position of each firm
3:  stepInterval =1; //discretizise distances in 1 km distance intervals
4:  medianDistance=180; //define the median of bilateral distances
5:  intervals= medianDistance/stepInterval;
6:  double[] distArray=array(intervals) // initialize array with constant length
/* CALCULATION */
7:  for (i=0 to length(lat)) do // interate over all entries
8:      lat1 = lat[i]; // get latitude and longitude of the firm i
9:      long1=long[i];
10:     j=i+1;
11:     for (j=0 to length(lat)) do // concern only unique bilateral distances
12:         lat1 = lat[j]; // get latitude and longitude of the firm j
13:         long1=long[j];
14:         distance= getDistance (lat1, long1, lat2, long2); // compute the orthodromic distance
15:         if (distance <= medianDistance) do
16:             pos= (int) round(distance/stepInterval,0); // get the position of the array bin to be updated
17:             distanceArray[pos]++; // Increment the array
18:         end if
19:     end for
20: end for
21: smoothDistance =kde(distanceArray) //kernel-smooth the distance array
```

running times while the improved index is up to three times faster for larger sets. The shorter running time arises from two points: First, the calculation of bilateral distances is not quadratic but reduced to ($n$*($n$-1)/2) and second, the kernel density estimation is more efficient since we directly start with a sorted array of occurrence at specific intervals. While the running time of the of the kernel density estimation for the standard algorithm is in $O(n^2)$ it is in $O(di)$ for the improved algorithm, where *di* stands for the number of distance intervals.

More obvious are however the differences in the RAM requirement that are constant for the improved algorithm (8640 byte) while they increase squarely for the standard algorithm. For a PC with 4 GB RAM, 7000 firms are the actual maximum.

In comparison to other attemps that try to fit the DO-index for large data sets, such as reported in Elison et al. (2012), our proposed algorithm does not calculate an approximation of the index. Nonetheless, there is a slight deviation from the original computation since we round the bilateral distance between two firms at the kilometer level in line 16 of our code what affects the results of the kernel density estimation. However, the step of increasing the precision to meter or even centimeter level is trivial and does not influence the time or space complexity of our index. To calculate the index at a meter precision for instance, *distArray* is simply initialized as a two dimensional array, where the second dimension ranges from 0

to 1000. While this increases RAM requirements by a factor of 1000, the space complexity is still in $O(di)$ and thus independent from the number of investigated firms.

| | **2000 firms** | **5000 firms** | **7000 firms** |
|---|---|---|---|
| Standard Algorithm | 0.716 sec.<br>0.18 GB | 5.357 sec.<br>1.12 GB | 11.82 sec.<br>2.19 GB |
| Improved Algorithm | 0.572 sec.<br>8640 byte | 2.513 sec.<br>8640 byte | 4.856 sec.<br>8640 byte |

Table 2: Runnig time and RAM requirements of calculating the DO-index for different firm populations.

# 4 Cluster Index by Scholl & Brenner

After having presented an efficient implementation of the DO-index we want to discuss a further computationally very efficient index for measuring spatial concentration that has been developed by Scholl & Brenner (2012).

As mentioned in section 2, their index differs from the other distance based measures as it does not consider the occurrence of bilateral distances at or up to a distance but computes firm specific cluster values, called $D_i$ values, by summing up inverted distances:

$$D_i = \frac{1}{J-1} \sum_{j=1, j \neq i}^{J} \left(f(d_{i,j})\right)^{-1}. \tag{2}$$

The term $\left(f(d_{i,j})\right)^{-1}$ stands for all possible functions that compute the inverted orthodromic distance between two points so that close neighborhoods have a high influence on a $D_i$ value while the weight of large distances converges to zero. In the simplest case, this is the hyperbola function. Because $D_i$ values are normalized by the term $\frac{1}{J-1}$, the index is independent of the number of firms.

Concerning the computational requirements, the DO-index and the index by Scholl & Brenner seem to be rather similar: The running time of both indices is bounded by $O(n^2)$ while the RAM requirements of the index by Scholl & Brenner are not constant but are linear, bounded by $O(n)$, where $n$ stands for the number of observed firms (see listing 3).

Listing 3: Cluster index by Scholl & Brenner (2012)

```
/* INPUT */
1:  double[] lat; //array with the latitude position of each firm
2:  double[] long; // array with the longitude position of each firm
3:  double[] clusterValueArray=array(lat.length()) // initialize array with linear length
/* CALCULATION */
4:  for (i<lat.length()) do // interate over all entries
5:      lat1 = lat[i]; // get latitude and longitude of firm i
6:      long1=long[i];
7:      j=i+1;
8:      for(j<lat.length()) do // concern only unique bilateral distances
9:          lat2 = lat[j]; // get latitude and longitude of firm j
10:         long2=long[j];
11:         distance= getDistance (lat1, long1, lat2, long2); // compute the orthodromic distance
12:         distance = invertDistance(distance)/( lat.length()-1); // invert the distance and normalize value
13:         clusterValueArray[i]+=distance; // update the cluster index array for firm i and j
14:         clusterValueArray[j]+=distance;
15:     end for
16: end for
17: smoothedValues =kde(clusterValueArray) //kernel-smooth the cluster-values array
```

However, the absence of Monte-Carlo simulations notably enhance the actual running time of the index for large data sets. The option of circumventing simulations is possible here since the index by Scholl & Brenner can compute independent random values of their index in the following way[6]:

First, a large sample of random firms (or points), denoted by $I$ is drawn. For each firm $i \in I$ an independent randomly drawn set $J_i$ containing $|I|$-1 firms is built. This allows calculating for each firm $i_1$ its $D_i$ value according to formula (2) using all firms $j_1 \ldots j_{|N|-1} \in J_1$. This procedure results in a benchmark set of $D_i$ values that are independent to each other as the $D_i$ value of each firm $i$ is built by another set of random firms. Finally, the Kolmogorov-Smirnov and Mann-U test are applied as a two sample test on the benchmark set and the $D_i$ values of the observed industry. By this, the index by Scholl & Brenner is able to test for a random distribution of firms without Monte-Carlo simulations (Scholl & Brenner (2012): 14).

In a nutshell, the computation of the benchmark values is quadratic but in comparison to the DO-index, this computation has to be done only once and not 1000 times. Furthermore,

[6] As described in Duranton & Overman (2005), the DO-index (as well as the majority of distance based methods) has to rely on Monte Carlo results since their sampling of distances is not independent of each other. See Duranton & Overman (2005): 1085 for more details.

since $D_i$ values are normalized by the number of firms, the same benchmark values can be used for different industries in the same area under investigation while the confidence bands of the DO-index are both area and industry specific.

To show the different computational requirements of the DO-index and the index by Scholl & Brenner when applying them to big data, we construct a square with the approximate size of the United States of America and build different sets of hypothetical industries from a spatial Poisson process starting from 20,000 to 110,000 firms. Note, that we can only compare the improved algorithm for DO-index here, since the number of firms is far too big for the standard algorithm. Table 3 shows that the RAM requirements are low and within the range of standard computers for both indices but the absence of Monte-Carlo simulations has a notable influence on the running of the index by Scholl & Brenner. Including the calculation of benchmark values, the running time of the latter one is 4.67 hours while the DO-index needs more than 24 days.

| | **20,000 firms** | **50,000 firms** | **80,000 firms** | **110,000 firms** |
|---|---|---|---|---|
| Improved DO-Algorithm[7] | 11.17 h.<br>19,200 byte | 69.40 h.<br>19,200 byte | 176.06 h.<br>19,200 byte | 332.622 h.<br>19,200 byte |
| Index by Scholl & Brenner[8] | 53.86 sec.<br>960,000 byte | 310.81 sec.<br>2.400,000 byte | 700.41sec.<br>3.840,000 byte | 1217.02 sec.<br>5.280,000 byte |

Table 4: Runnig time and RAM requirements of computing the DO-index and the index by Scholl&Brenner for different industries.

[7] Running time includes 1000 Monte-Carlo simulations for each industry.
[8] Number of firms for benchmark calculation: 200,000 (running time: 14516.60 sec. RAM requirements: 9.600,000 byte).

# 5 Conclusion

In this paper, we have introduced an improved algorithm for the DO-index, the most established distance based method for measuring concentration in spatial econometrics. The algorithm shows a lower running time and constant space complexity allowing the computation of huge-datasets even with standard computational power. Our findings are not limited to the DO-index but the basic idea of the algorithm, condensing information to intervals, is applicable to most of the other existing distance-based methods as well.

Furthermore, we have discussed the index of Scholl & Brenner (2012). Testing the improved DO-index against the index by Scholl & Brenner, we have shown that the latter metric is even more suitable for big data analysis since it does not require Monte-Carlo simulations and is therefore much faster.

The next steps are applying the methods to real world big data analysis such as a pan-European analysis of spatial concentration. Furthermore, in order to make distance-based methods more applicable, R-packages of both indices will be published on the CRAN repository.

# 6 References

Duranton, Gilles; Overman, Henry G. (2005): Testing for Localization Using Micro-Geographic Data. In: Review of Economic Studies 72: 1077–1106.

Ellison, Glenn; Glaeser, Edward; Kerr, William (2010): What Causes Industry Agglomeration? Evidence from Coagglomeration Patterns. In: American Economic Review 100 (2010): 1195–1213.

German Federal Ministry of Economics and Technology (2010): Möglichkeiten und Grenzen einer Verbesserung der Wettbewerbssituation der Automobilindustrie durch Abbau von branchenspezifischen Kosten aus Informationspflichten. Stuttgart.

Klier, Thomas; McMillen, Daniel P. (2008): Evolving Agglomeration in the U.S. Auto Supplier Industry. In: Journal of Regional Science 48 (1): 245–267.

Kosfeld, Reinhold; Eckey, Hans-Friedrich; Lauridsen, Jørgen (2011): Spatial point pattern analysis and industry concentration. In: The Annals of Regional Science 47:311–328.

Marcon, Eric; Puech, Florence (2003): Evaluating the Geographic Concentration of Industries Using Distance-Based Methods. In: Journal of Economic Geography, 3(4): 409-428.

Marcon, Eric; Puech, Florence (2010): Measures of the geographic concentration of industries: improving distance-based methods. In: Journal of Economic Geography 10 (5): 745-762.

Marcon, Eric; Puech, Florence (2012): A topology of distance-based measures of spatial concentration. URL: http://halshs.archives-ouvertes.fr/halshs-00679993/.

Miller, Harvey J. (2010): The data avalanche is here. Shouldn't we be digging? In: Journal of Regional Science 50 (1): 181–201.

Openshaw, S. (1984): The modifiable areal unit problem. In: Concepts and Techniques in Modern Geography 38.

Scholl, Tobias; Brenner, Thomas (2012): Detecting Spatial Clustering Using a Firm-Level Cluster Index. In: Working Papers on Innovation and Space 02.12: 1-29.

Vitali, Stefania; Mauro, Napoletano; Fagiolo, Giorgio (2009): Spatial Localization in Manufacturing: A Cross-Country Analysis. In: LEM Working Paper Series 2009 (04): 1-37.