arTs Università degli Studi di Trieste
Archivio della ricerca – postprint

A methodology for dealing with spatial big data

Gabriella Schoier* and Giuseppe Borruso

DEAMS – Department of Economic, Business, Mathematic and Statistical Sciences,
'Bruno de Finetti' University of Trieste,
Trieste, Italy
Email: gabriella.schoier@deams.units.it
Email: Giuseppe.borruso@deams.units.it
*Corresponding author

Abstract: Spatial data mining (SDM) refers to the mining of knowledge from spatial data. Recently, location-based services have enabled the gathering of a significant amount of geo-referenced data, i.e., of spatial big data (SBD). Spatial datasets often exceed the ability of current computing systems to manage these data with reasonable effort; therefore, data-intensive computing and data mining techniques are useful tools for conducting an analysis. In this paper, we present an approach to the clustering of high-dimensional data that allows a flexible approach to the statistical modelling of phenomena characterised by unobserved heterogeneity. Numerous clustering algorithms have been developed for large databases; density-based algorithms particularly treat a huge amount of data in large spatial databases. We present the *Modified Density-Based Spatial Clustering of Applications with Noise* (MDBSCAN) algorithm and compare it to the classical k-means approach. Both applications use synthetic datasets and a dataset of satellite images.

Keywords: spatial data mining; clustering algorithms; arbitrary shape of clusters; Lagrange-Chebyshev metrics; efficiency of large spatial databases; handling noise; image analysis.

Reference to this paper should be made as follows: Schoier, G. and Borruso, G. (2017) 'A methodology for dealing with spatial big data', *Int. J. Business Intelligence and Data Mining*, Vol. 12, No. 1, pp.1–13.

Biographical notes: Gabriella Schoier is an Associate Professor of Statistics at the University of Trieste. Her research topics are mainly oriented to data mining, spatial data mining, web mining and text mining, social network analysis, customer satisfaction, time series analysis, operational risk, statistical matching.

Giuseppe Borruso is an Associate Professor of Geography at the University of Trieste. His research topics are mainly oriented to cities and geographic information technologies.

This paper is a revised and expanded version of the paper 'On the problem of clustering spatial big data', presented at the International Conference on Computational Science and Its Applications (ICCSA 2015), Session: Workshop on Geographical Analysis, Urban Modeling, Spatial Statistics (GEOG-and-MOD 2015) Banff, AB, Canada, 22–25 June 2015.

1 Introduction

Big data refers to data that cannot be analysed using traditional tools. Big data that are considered in a spatial context are known as spatial big data. Spatial and other kinds of data have exponentially increased in size since the introduction of the geographic information system (GIS) and desktop GIS in particular. GIS users and experts have been tasked with managing big amounts of data for some time; however today, there is currently an explosion of geo-referenced data (see e.g., Cressie, 1993; Bailey and Gatrell, 1996).

This exponential development in the availability of and access to spatially referenced information has induced the need for better analysis techniques to understand different phenomena. In particular spatial clustering algorithms, which group similar spatial objects into classes, can be used for knowledge discovery in spatial databases, i.e., databases managing 2D or 3D points, polygons, etc., or points in some d-dimensional feature space (see, e.g., Fayyad et al., 1996; Han et al., 2001; Koperski et al., 1998; Steinbach et al., 2003; Xu and Wunsch, 2005).

Clustering's aim is to partition a set of objects into clusters such that the objects within a group are more similar to one another than objects in different clusters. Numerous clustering algorithms have been developed for large databases; in particular, density-based algorithms treat large amounts of data in large spatial databases. These algorithms regard clusters as dense regions of units that are separated by regions of low density (representing noise).

The aims of this paper are to both present a density-based algorithm called the *Modified Density-Based Spatial Clustering of Applications with Noise* (MDBSCAN) algorithm for the discovery of clusters of units in large spatial datasets and to compare it with the classical k-means method. This algorithm is a modification of the *density-based spatial clustering of applications with noise* (DBSCAN) algorithm (Ester et al., 1996). The modifications of this algorithm with respect to the original include consideration of spatial and non-spatial variables and the use of the Lagrange-Chebyshev metric instead of the usual Euclidean one. The applications use both synthetic datasets and a dataset of satellite images.

2 The data

In the next two subparagraphs, we present big data and spatial big data.

2.1 Big data

The term big data was coined in the last decade to describe large amounts of data that cannot be analysed using traditional tools. Many organisations have access to huge quantities of data yet lack the ability to extract information because the data are gathered not only in a structured format but also in semi-structured and unstructured formats. Therefore, when analysing a big data problem, three characteristics must be considered: velocity, volume and variety (see, e.g., Zikopoulos et al., 2012).

As one can imagine big data come from different sources.

First of all *Human-sourced information*, this information is now almost entirely digitised and stored everywhere. Examples include social networks (Facebook, Twitter,

etc.), blogs and their comments, personal documents, pictures (Instagram, etc.), videos, (YouTube, etc.), internet searches, mobile data content, user-generated maps and e-mails. These data are loosely structured and often ungoverned.

Another source derived from *Process-mediated data*, these processes record and monitor business events of interest, such as registering a customer, manufacturing a product, taking an order, etc. This process is highly structured and includes transactions, reference tables and relationships, as well as the metadata that determines its context. Examples of this type of data include medical records, commercial transactions, banking/stock records, e-commerce and credit cards.

A third category is *machine-generated data* derived from the phenomenal growth in the number of sensors and machines used to measure and record events and situations in the physical world. As sensors proliferate and data volumes grow, this form of data is becoming an increasingly important component of the information stored and processed by many businesses. Its well-structured nature is suitable for computer processing howether its size and speed are beyond traditional approaches. Examples of machine-generated data include: weather/pollution sensors, traffic sensors/webcams, security/surveillance videos/images, mobile phone location, road sensors (cars, trucks), rail sensors (trains) and air sensors (planes).

2.2 Spatial big data

Spatial data mining can be used for browsing spatial databases, understanding spatial data, discovering spatial relationships, optimising spatial queries.

Spatial data, as other kinds of data, are becoming bigger and bigger, although since the introduction of GIS and desktop GIS in particular, GIS users and experts have become facing with the issue of managing big amount of data, even though often data were much more difficult to retrieve than today. In geographical terms, the nature of data is such that an increase in dataset dimension is always possible, both in terms of the number of records to be considered and in terms of the attributes of the geographical data. Both the vector and raster data formats used in GIS analysis tend to be multidimensional, i.e., containing a quantity of elements that can be considered in any form of grouping and aggregation. In any case at least two fields (if not three) are needed to store the spatial information while all the attribute data contribute to increasing the dimension of the dataset. Satellite imagery in particular represents another situation, in which redundant information is also considered, as very close pixels present very little differences although weighting in the processing, storage and visualisation time of the data. Therefore compression algorithms and proficient clustering tools are needed in order to extract more precise and complete set of geographic information (Bedard, 2014; Chen et al., 2006; Worboys and Duckham, 2004).

3 The methodology

K-means algorithm versus DBSCAN and its modification, i.e., MDBSCAN are explained in the following subparagraphs.

3.1 K-means algorithm

The K-means algorithm is very well known (see, e.g., Jain, 2010). The algorithm allocates the data points (objects) into clusters, so as to minimise the sum of the squared distances between the data points and the centre of the clusters.

The centres of the clusters are initialised by either randomly selecting from the data or by fixing particular data points; then the dataset is clustered in the process of assigning each point to the nearest centre. When the dataset has been identified, the average position of the data points within each cluster is calculated and the cluster's centre is then moved to the average position. This process is repeated until a condition of stopping is reached.

More precisely, the algorithm contains the following steps:

Step 1 Place k points into the space represented by the objects that are being clustered. These points represent initial group centroids.

Step 2 Assign each object to the group that has the closest centroid.

Step 3 When all objects have been assigned, recalculate the positions of the k centroids.

Step 4 Repeat steps 2 and 3 until the centroids no longer move or another stopping rule is achieved. This produces a separation of the objects into groups from which the metric to be minimised can be calculated.

The K-means algorithm requires three user-specified parameters: number of clusters k, cluster initialisation, and distance metric. The most critical choice is k.

K-means is typically used with the Euclidean metric for computing the distance between points and cluster centres. As a result, K-means finds spherical or ball-shaped clusters in the data. The K-means when combined with Mahalanobis distance metric has been used to detect hyperellipsoidal clusters (see Mao and Jain, 1996); howether this comes at the expense of a higher computational cost.

3.2 DBSCAN and MDBSCAN algorithms

In this section, we will consider clustering methods based on the notion of density. These methods regard clusters as dense regions of units that are separated by regions of low density (representing noise); moreover, they may be used to discover clusters of arbitrary shape (see, e.g., El-Sonbaty et al., 2004; Ester et al., 1996; Mumtaz and Duraiswamy, 2010; Schoier and Borruso, 2004; Schoier and Bato, 2007).

Among these the *DBSCAN* algorithm (Ester et al., 1996) is a locality-based algorithm that relies on a density-based notion of clustering. Density based methods can be used to filter out noise (outliers) and discover clusters of arbitrary shape. This algorithm judges the density around the neighbourhood of a unit to be sufficiently dense if the number of points within a distance *EpsCoord* of the unit is greater than *MinPts*, in this case the unit is a *core point* otherwise it is a *border point*. This algorithm has been generalised in different papers (see, e.g., Sander et al., 1998).

The key idea is that each point of a cluster in the neighbourhood of a given radius must contain a minimum number of points, i.e., the density in the *neighbourhood* must exceed some threshold. The shape of a *neighbourhood* is determined by the choice of a

distance function for two points p and q, denoted by $dist(p, q)$. The *Epscoord* neighbourhood of a point q is defined by

$$N_{\varepsilon}(q) = \{q \in D \,|\, dist(p, q) \leq \varepsilon\}$$

where D is a dataset of points.

A naive approach could require for each point in a cluster that there are at least a minimum number (*MinPts*) of points in an *Eps-neighbourhood* of that point. However, this approach fails because there are two kinds of points in a cluster, points inside of the cluster (*core points*) and points on the border of the cluster (*border points*).

In general, an *Eps-neighbourhood* of a border point contains significantly fewer points than an *Eps-neighbourhood* of a core point. Therefore, one would have to set the minimum number of points to a relatively low value in order to include all points belonging to the same cluster. This value, however, would not be characteristic of the respective cluster particularly in the presence of noise. Therefore, one has to require that for every point p in a cluster C there is a point q in C so that p is inside of the *Eps-neighbourhood* of q and $N_{\varepsilon}(q)$ contains at least *MinPts* points.

This definition is elaborated on the following: a point p is density reachable from a point q if there is a chain of points $p_1, p_2 \ldots\ldots, p_{n-1}, p_n$ where $p_1 = p$ and $p_n = q$ such that p_i is direct density reachable from p_{i+1}.

Moreover, a point p is *directly density reachable* from a point q if p belongs to the neighbourhood of q and q is a core point.

The clustering formed by the DBSCAN follows the rules below:

1 a point can only belong to a cluster if and only if it lies within the *Epscoord*-neighbourhood of some core point in the cluster

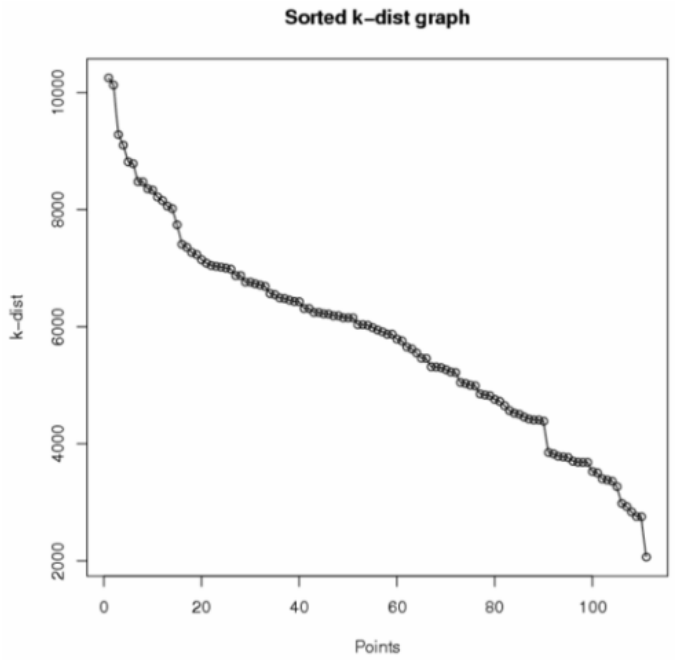
2 a *core point o* within the *Epscoord*-neighbourhood of another core point p must belong to the same cluster as *p*

3 a *border point r* within the *Epscoord*-neighbourhood of some core point must belong to the same cluster to at least one of the core points

4 a border point that does not lie within the *Epscoord*-neighbourhood of any core point is considered to be noise.

The value of the parameter *MinPts* is fixed as in Ester et al. (1996). In order to determine the parameter *EpsCoord*, regarding the spatial variables, and the parameter *Eps*, regarding the non spatial variables we consider the algorithms *SorteKdist* and *SorteKdist2* respectively. The former *SorteKdist*, which is used for the spatial variables, is implemented following Ester et al. (1996). For every point of the dataset represented by the matrix of coordinates, it evaluates the distances from the nearest k-points (belonging to the same dataset), orders them in a decreasing way and gives a graphical representation of these distances [for the choice of k see Ester et al. (1996) and see Figure 1 for an example].

Figure 1 Sorted k-distances graph for a sample dataset



The latter *SorteKdist2*, which is used for non-spatial variable is similar. For every point of the dataset represented by the matrix *SetOfPoints*, it evaluates the distances from the nearest k-points (belonging to the same dataset) and gives the mean value.

As stated previously the form of the neighbourhood is determined by the choice of the distance function between two points' x_i and x_j. The Euclidean distance generates a sphere around the point of circular shape (MDBSCAN (euc)).

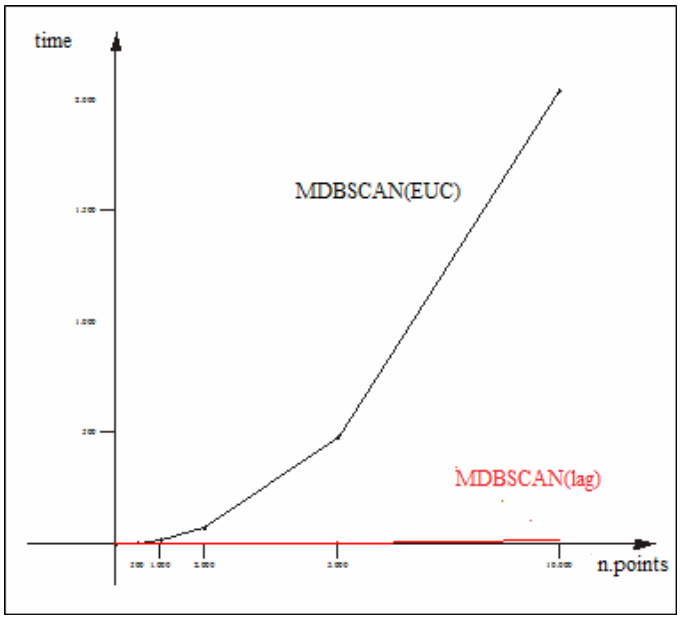
$$_2d_{ij} = \left[\sum_{s=1}^{p} \left|x_{is} - x_{js}\right|^2\right]^{1/2}$$

The Lagrange-Chebyshev distance generates a sphere around the point of square shape (MDBSCAN(lag)).

$$_\infty d_{ij} = \max_s \left|x_{is} - x_{js}\right|$$

We have verified both the validity of the MDBSCAN based on the Lagrange distance and that based on the Euclidean distance on the base of CPU times. As one can see from Figure 2, the time (in seconds) remains similar with the increase in the number of points if we use MDBSCAN(lag), it is not the same if we use the Euclidean distance.

Figure 2 MDBSCAN(lag) and MDBSCAN(euc) (see online version for colours)



4 Application: results and discussion

In order to test the MDBSCAN(lag) and the K-means algorithm we considered some applications to both synthetic datasets and a satellite image dataset.

The analysis of the bit-map images are a real example of raster spatial analysis, the image is formed by a regular grid of pixel and to every cell a colour has 24 bit (16 million of colours are possible). We have considered images that have 300 pixel by side (for a total of 90,000 pixel) in a space of Red, Green, Blue (= RGB) colours to 24 bpp (bit by pixel, about 16 millions of colours) this colour format is known as true-colour.

Every pixel is a statistical unit, a point in the space of five dimensions: two relatively of the spatial attributes, the other three to the non-spatial attributes. In order to apply the algorithm a standardisation of the variables has been performed.

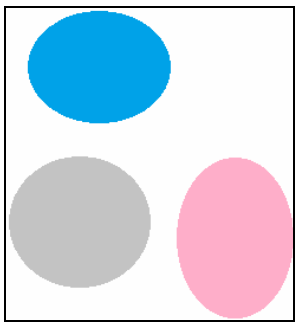
The implemented MDBSCAN(lag) algorithm involved choosing different thresholds for the coordinates and for the other indices. Regarding the spatial variables the algorithm *SorteKdist* gives the first point in the first valley. As regards the non-spatial variables the algorithm *SorteKdist2* has been implemented.

After the standardisation of the variables, the *R* language and a visualisation language have been used for the analysis.

We have applied the MDBSCAN(lag) to synthetic datasets. We performed then a visual analysis of the results of clustering.

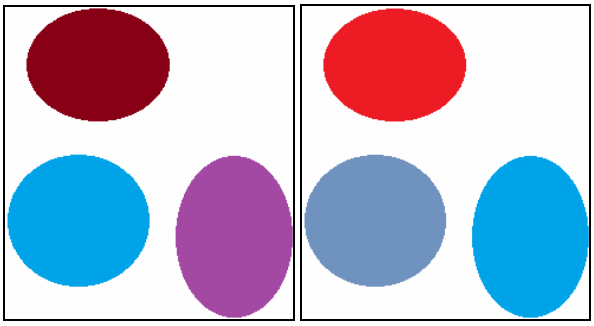
In the dataset presented in Figure 3, we have considered a simple synthetic dataset with three spherical clusters without noise.

Figure 3 A synthetic dataset (see online version for colours)



As one can see both algorithms find the clusters (see Figure 4).

Figure 4 (a) K-means (b) Clustering MDBSCAN(lag) results (see online version for colours)



(a) (b)

The K-means algorithm has difficulty in handling clusters of arbitrary shape and noise. This is not the case of the MDBSCAN(lag).

Two synthetic datasets are presented in Figure 5, while Figure 6 shows the results of image representation by using the MDBSCAN(lag).

Figure 5 Two synthetic datasets

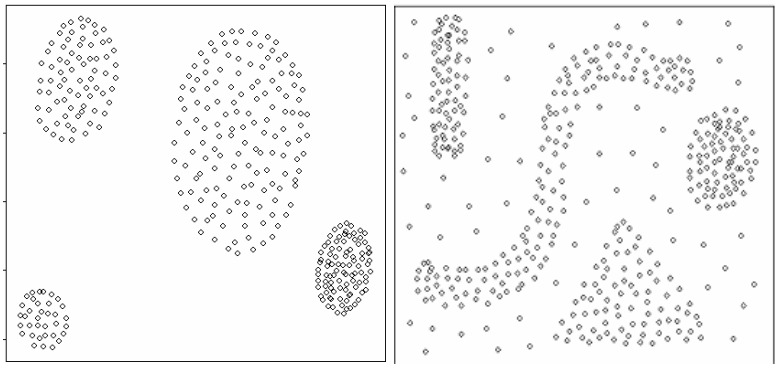
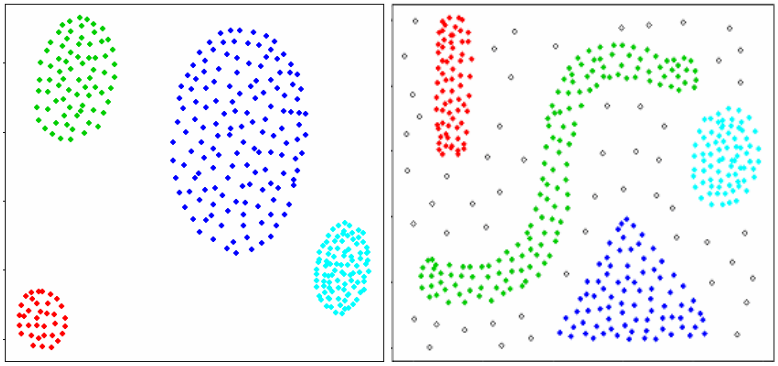


Figure 6 MDBSCAN(lag) results on the two synthetic datasets (see online version for colours)



It is easily to note that using MDBSCAN(lag) we are able to correctly identify all the clusters in each dataset. In particular, the algorithm has been able to locate clusters of concave and elongated shape, clusters of various shapes and different densities in the presence of noise.

Another dataset that has been considered is the political map of Italy (see Figure 7). This is an example with more complex clusters: they are of various shapes and sizes, they are elongated and with surrounding elements which reflect noise. As one can see from the results (Figure 8), the MDBSCAN(lag) has been able to manage all of these elements and to correctly identify the various clusters. The edges of the various regions have been visualised with white colour (noise), while the area around Italy has been considered as a single cluster.

Figure 7 A dataset with rumours (see online version for colours)

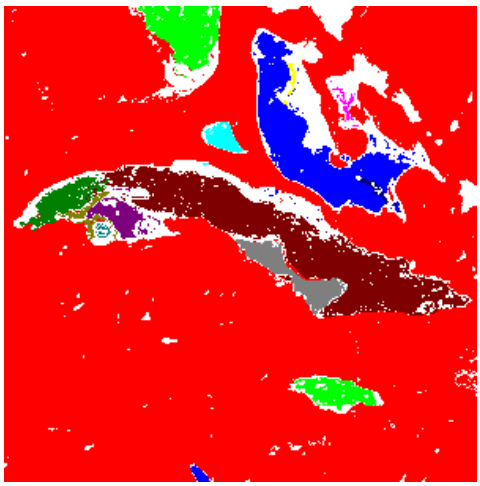
Figure 8 MDBSCAN(lag) clustering results (see online version for colours)

The next application of MDBSCAN(lag) regards a dataset of satellite image.

Figure 9 A satellite image dataset (see online version for colours)

The MDBSCAN(lag) algorithm has been used on a satellite image of the Island of Cuba in the Gulf of Mexico in order to understand the performance of the algorithm over areas in which land and sea borders are difficult to detect. In Figure 9, the starting dataset is portrayed, while Figure 10 presents in false colours the different groups or clusters identified by the procedures.

Figure 10 MDBSCAN(lag) clustering results (see online version for colours)



The more easily detectable parts of the pictures have been assigned a common colour. Therefore in Figure 10, it is possible to observe in red the water component of the sea, while other colours as green and brown highlight the island and the other components of the Florida peninsula.

The less easily detectable parts of the pictures have been assigned a white colour representing a level of 'noise'.

5 Conclusions

In this paper, the problem of dealing with large spatial databases is considered and a methodology for dealing with this type of data is presented.

A primary tool for knowledge discovery in spatial databases is clustering algorithms. In this paper two clustering methods have been applied to identify homogeneous areas: K-means, and MDBSCAN(lag).

MDBSCAN(lag) is an algorithm modification of the original DBSCAN algorithm by Ester et al. (1996).

Both DBSCAN and MDBSCAN are used to identify homogeneous areas. Moreover they can find out clusters of different shapes and sizes from large amount of data containing noise and outliers.

Our algorithm takes into consideration both spatial and non-spatial variables relevant for the phenomena that has to be analyzed. To improve computational aspects we proposed to use a Lagrange-Chebyshev metric instead of the Euclidean one so to apply the MDBSCAN(lag).

The applications regards both synthetic and real datasets. The spatial clustering analysis allowed to obtained good bit-map images and good representation of satellite images.

The main advantage of K-means algorithm is its simplicity and speed at which it can to run on large datasets. One problem of the application of the K-means is the necessity of knowing a priori the number of clusters. Other problems include the identification of noise (outliers) and the discover of clusters of arbitrary shape

The MDBSCAN(lag) algorithm is robust enough to identify clusters in noisy data, requires just a few parameters and is mostly insensitive to the ordering of points in the database. This algorithm is efficient even for very large spatial databases, discovers clusters of arbitrary shape and does not need to know the number of clusters in the data a priori, as opposed to K-means algorithm.

The tests presented in this paper increased our appreciation for the efficiency and versatility of the MDBSCAN(lag). The algorithm has produced excellent results in all situations.

The MDBSCAN(lag) can help GIS systems which are very effective to store and communicate geographic information. The representation on maps can be integrated with reports, three-dimensional views, photographic images and other media representations too.

Acknowledgements

The paper derives from joint reflections of the two authors. The first author realised paragraphs 1, 2.1, 3.1, 3.2, 4, 5, while Giuseppe Borruso wrote paragraphs 2.2.

References

Bailey, T.C. and Gatrell, A.C. (1996) *Interactive Spatial Data Analysis*, Addison Longman, Edinburgh.

Bedard, Y. (2014) *Beyond GIS: Spatial On-Line Analytical Processing and Big Data*, University of Maine [online] http://umaine.edu/scis/files/2014/09/Beyond-GIS-Spatial-On-Line-Analytical-Processing-and-Big-Data-Yvan-Bedard.pdf (accessed May 2016).

Chen, Y., Suel, T. and Markowetz, A. (2006) 'Efficient query processing in geographic web search engines', Paper presented at *SIGMOD 2006*, 27–29 June 2006, Chicago, Illinois, USA [online] http://cis.poly.edu/suel/papers/geoquery.pdf (accessed May 2016).

Cressie, N.A.C. (1993) *Statistics for Spatial Data*, John Wiley & Sons, London.

El-Sonbaty, Y., Ismail, M.A. and Farouk, M. (2004) 'An efficient density-based clustering algorithm for large databases', *Proceedings of the 16th IEEE International Conference on Tods with Artificial Intelligence (ICTAI)*, pp.673–677.

Ester, M., Kriegel, H.P., Sander, J. and Xiaowei, X. (1996) 'A density-based algorithm for discovering clusters in large spatial databases with noise', *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp.94–99.

Fayyad, U., Piatesky-Shapiro, G. and Smyth, P. (1996) 'From data mining to knowledge discovery in databases' [online] http://www.kdnuggets.com/gpspubs/aimag-kdd-overview-1996Fayyad.pdf (accessed May 2016).

Han, J., Kamber, M. and Tung, A.K.H. (2001) 'Spatial cluttering methods in data mining: a survey' [online] ftp://ftp.fas.sfu.ca/pub/cs/han/pdf/gkdbk01.pdf (accessed May 2016).

Jain, A.K. (2010) 'Data clustering: 50 years beyond K-means', *Pattern Recognition Letters*, Vol. 31, pp.651–666.

Koperski, K., Han, J. and Adhikary, J. (1998) 'Mining knowledge in geographical data' [online] ftp://ftp.fas.sfu.ca/pubcs/han/pdf/geo_survey98.pdf (accessed May 2016).

Mao, J. and Jain, A.K. (1996) 'A self-organizing network for hyper-ellipsoidal clustering (HEC)', *IEEE Trans. Neural Networks*, Vol. 7, No. 1, pp.16–29.

Mumtaz, K. and Duraiswamy, K. (2010) 'An analysis on density-based clustering of multi-dimensional spatial data', *Indian Journal of Computer Science and Engineering*, Vol. 1, No. 1, pp.8–12.

Sander, J., Ester, M., Kriegel, H.P. and Xiaowei, X. (1998) 'Density-based clustering from in spatial databases: the algorithm GDBSCAN and its applications' [online] http://www.dbs.informatik.uni-muenchen.de/Publikationen/ (accessed May 2016).

Schoier, G. and Bato, B. (2007) 'A modification of the DBSCAN Algorithm in a Spatial Data Mining Approach', *Meeting of the Classification and Data Analysis Group of the SIS: CLADAG 2007*, 12–14 September 2007, pp.395–398, MACERATA: EUM, Macerata.

Schoier, G. and Borruso, G. (2004) 'A clustering method for spatial databases', in Laganà, A., Gavrilova, M.L., Kumar, V., Mun, Y., Tan, C.J.K. and Gervasi, O. (Eds.): *Proceedings of the ICCSA 2004 International Conference*, pp.1089–1095, Springer, Assisi, Italy.

Steinbach, M., Ertöz, L. and Kumar, V. (2003) 'The challenges of clustering high dimensional data' [online] http://www-users.cs.umn.edu/~kumar/papers/high_dim_clustering_19.pdf (accessed May 2016).

Worboys, M. and Duckham, M. (2004) *GIS: A Computing Perspective*, CRC Press, Boca Raton, FL.

Xu, R. and Wunsch II, D. (2005) 'Survey of clustering algorithms' [online] http://ieeexplore.ieee.org/iel5/72/30822/01427769.pdf (accessed May 2016).

Zikopoulos, P.C., Eaton, C., de Roos, D., Deutsch, T. and Lapis, G. (2012) *Understanding Big Data Analytics for Enterprise Class Hadoop and Streaming Data*, McGraw-Hill, New York.