Chemometrics and Intelligent Laboratory Systems 120 (2013) 92–96

Contents lists available at SciVerse ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemolab

Revised DBSCAN algorithm to cluster data with dense adjacent clusters

Thanh N. Tran [a,*], Klaudia Drab [b], Michal Daszykowski [b]

[a] Center for Mathematical Sciences-Europe, Merck MSD, Molenstraat 110, 5342 CC Oss, PO Box 20, 5340 BH Oss, The Netherlands
[b] Department of Analytical Chemistry, Chemometric Research Group, Institute of Chemistry, The University of Silesia, 9 Szkolna Street, 40-006 Katowice, Poland

ARTICLE INFO

Article history:
Received 18 May 2012
Received in revised form 22 October 2012
Accepted 11 November 2012
Available online 20 November 2012

Keywords:
Clustering
Density-based clustering

ABSTRACT

Over the last several years, DBSCAN (Density-Based Spatial Clustering of Applications with Noise) has been widely used in many areas of science due to its simplicity and the ability to detect clusters of different sizes and shapes. However, the algorithm becomes unstable when detecting border objects of adjacent clusters as was mentioned in the article that introduced the algorithm. The final clustering result obtained from DBSCAN depends on the order in which objects are processed in the course of the algorithm run. In this article, a modified version of the DBSCAN algorithm is proposed to solve this problem. It was shown that by using the revised algorithm the clustering results are considerably improved, in particular for data sets containing dense structures with connected clusters.

1. Introduction

DBSCAN (Density-Based Spatial Clustering of Applications with Noise), introduced by Ester et al. [1], is a non-parametric, density-based clustering technique. Along with partitioning methods and hierarchical clustering, DBSCAN belongs to the third category of clustering methods and assumes that a cluster is a region in the data space with a high density.

Compared to non-density based clustering methods, the DBSCAN algorithm has unique and advanced features that are useful when detecting objects/class/patterns/structures of different shapes and sizes. DBSCAN is a good candidate to find 'natural' clusters and their arrangement within the data space when they have a comparable density without any preliminary information about the groups present in a data set [12].

Initially, the DBSCAN algorithm was proposed for clustering spatial data. Owing to its unique features, the algorithm rapidly became popular and was applied as a clustering approach for other data types. Some examples of its applications include applications in such fields of science as civil engineering (grouping spatial civil infrastructure networks [2]), chemistry [3], spectroscopy (grouping of single particle mass spectra [4] and aerosol time-of-flight mass spectrometry [5]), social sciences (clustering of insects based on chemical pheromone data [6,7]), and medical diagnostics based on medical images (for detection of brain atrophy patterns [8] and to detect skin lesions [9,10]). DBSCAN can also be applied in the field of remote sensing to perform segmentation of three-dimensional images (hyperspectral images) [11].

In addition to the possibility of detecting clusters of arbitrary shapes, DBSCAN is relatively fast when clustering small and medium data sets. The complexity of DBSCAN is at least $O(n \cdot \log(n))$. The most time-consuming step of clustering is the calculation of the similarity measure between data objects, while the clustering itself requires only a single scan through the data objects. If necessary, the computational performance of DBSCAN can be accelerated by using a faster computer and/or modifying the code to enable parallel computing (using a multiprocessor computer [14–16] or a distributed computing network [17]).

In spite of its practical features, the original DBSCAN algorithm fails when the border objects of two clusters are relatively close. For spatial types of data, groups of objects are relatively well separated, but for other types of data, this is not always the case. When data are dense, e.g. X-ray 3D tomography data of a dense material [13], almost every cluster is adjacent to several other ones. Due to the increased number of border objects in contact areas between clusters, the result of clustering depends on the order in which the data objects are processed during the so-called expansion step of the algorithm [1]. As a consequence, the original algorithm is less robust. Ester et al. acknowledged this issue as a drawback of DBSCAN. On the other hand, for spatial types of data, for which the DBSCAN approach was initially designed, the results of clustering very rarely indicate its decreased performance.

In this article, we revise the concept of DBSCAN and tune the algorithm in order to achieve a more robust performance. As a consequence, a solution is proposed that extends the applicability of the algorithm to many different data types and overcomes the issue of border samples belonging to adjacent clusters.

The paper is organized as follows. In Section 2 the original DBSCAN is reviewed and simplified. In Section 3, details of the revised DBSCAN algorithm along with pseudocode implementations are provided.

* Corresponding author. Tel.: +31 412 66 2690.
E-mail address: thanh.tran@merck.com (T.N. Tran).

0169-7439/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.chemolab.2012.11.006

2. Theory

2.1. DBSCAN algorithm

In this section, a number of definitions and properties of the DBSCAN algorithm are reviewed.

Given a data set $\boldsymbol{X}$ containing a total of N objects, DBSCAN is a density-based clustering algorithm which formulates a local density denoted as $density(x_i)$ in the neighborhood of the i-th object $x_i \in X$, as the total number of objects in its neighborhood: $density(x_i) = Count(N_{Eps}(x_i))$, where $N_{Eps}(x_i)$ denotes neighboring objects in the neighborhood of a radius Eps—the first parameter of DBSCAN: $N_{Eps}(x_i) = \{x_j | \forall j,\ \text{distance}(x_j, x_i) < Eps\}$.

Bearing in mind the number of objects in a neighborhood, three types of objects can be distinguished, namely *core objects*, *border objects* and *noise*. Given the minimal number of objects in the neighborhood, *MinPts*, a second parameter of the algorithm, an object x_{core} is defined as a *core object* (●) if $density(x_{core}) \geq MinPts$ (its local density is higher than *MinPts*). An object is a *border object*, x_{border} (●), if $density(x_{border}) < MinPts$ AND a *core object* exists so that $x_{border} \in N_{Eps}(x_{core})$; the *border object* x_{border} belongs to the neighborhood of x_{core} and the local density is less than *MinPts*.

Another important definition used in the course of DBSCAN clustering is related to d*ensity-reachable* objects. Two objects, x_1 and x_n, are called *density-reachable* if a chain of objects $x_1, \ldots, x_i, x_{i+1}, \ldots, x_n$ exists where $i \geq 1$ and $n \geq 2$ so that for all $i < n$, x_i is a *core object*, $density(x_i) \geq MinPts$, AND x_{i+1} is a neighbor of x_i, $x_{i+1} \in N_{Eps}(x_i)$. As properties, $x_1, \ldots, x_{n-1}$ are *core objects* and x_n is either a *core object* or *border object*.

On the basis of the definitions introduced earlier, a cluster of objects is defined as a set of objects that are *density-reachable* from an arbitrary *core object* (of a cluster) [1].

The last category of objects is *noise* objects. An object is a *noise object*, x_{noise}, if in the neighborhood of a given radius *Eps*, there are less than *MinPts* of objects, none of which is a *core object* ($density(x_{noise}) < MinPts$). In Fig. 1, key DBSCAN definitions are depicted.

Using the DBSCAN algorithm, the clustering of objects from any cluster C is a two-step procedure. At first, an arbitrary *core object* of cluster C, x_{c1}, is identified. Then, all of the *density-reachable objects* from the *core object* x_{c1} are retrieved. In the second step, every chain of objects starting from x_{c1} is detected recursively.

Two sub-routines (*dbscan* and *ExpandCluster*) of the basic DBSCAN algorithm are presented as pseudocodes in Fig. 2A and B. All *core objects* in chains are stored in the so-called *seeds list*. The *seeds list* is updated during a 'for' loop whenever a new *core object* is discovered (within a neighborhood defined by *Eps* that is being studied) and as definition is an element of a chain.

2.2. Implementation of DBSCAN—natural patterns approach

In [3] the concept of density-based approaches was introduced to the chemometric society. However, it should be pointed out that the general clustering framework, as stated in this paper, differs essentially from DBSCAN clustering. It is summarized as follows:

```
for object i
    if the i-th object is not a member of a given cluster yet
        create a new cluster C
        while the neighboring objects satisfy the cluster condition
            expand cluster C
        end
    end
end
```

As a consequence of the expansion step formulated above, during clustering not only *core objects* are incorporated into a given cluster but all of the neighboring objects (including non-core objects) of the *density-reachable* chains. Such a cluster expansion strategy results in longer chains of objects and thus a greater chance of the bridging of clusters. This explains why implementation of the natural patterns approach is more sensitive to changes of the *Eps* parameter that DBSCAN is. On the other hand, in some situations such a property can be regarded as an advantage.

2.3. Revised DBSCAN robust for dense adjacent clusters

In this section a revised version of DBSCAN that is suitable for dense data containing adjacent clusters is introduced. In DBSCAN, data density is described by two parameters the neighborhood, size, *Eps*, and the number of data objects in that neighborhood, *MinPts*. In order to identify clusters correctly, data objects in the area of contact should be recognized as border objects. Hence as a rule, the more contact areas in the data space, the more border objects should be detected and the larger neighborhood and the number of objects in that neighborhood. The *density-reachable* chain of objects $x_1, \ldots, x_n$ can be either in the form $[x_{core_1}, \ldots, x_{core_{n-1}}, x_{core_n}]$ with all *core objects* or $[x_{core_1}, \ldots, x_{core_{n-1}}, x_{border_n}]$ with all *core objects* except the last object being a *border object*. The *border object*, therefore, does not contribute to the expansion mechanism of *density-reachable* chains of objects—the essential step of DBSCAN. For this reason, the proposed improvement step of the algorithm aims to disconnect the last *border objects* of the *density-reachable* chain. This can be achieved by exploring a new concept, the so-called *core-density-reachable* objects, in the course of clustering. *Core-density-reachable* objects are chains of objects $x_1, \ldots, x_n$ where x_i is *core object* for all $i \leq n$.

For this reason, the expansion step of clustering is revised (see step 2 in ExpandCluster) in order to use *core-density-reachable* chains (only with *core objects*) instead of *density-reachable* chains. Since they contain a similar number of *core objects*, the same *core objects* for each cluster are identified. However, *border objects* (originally assigned during the expansion step) remain temporarily unclassified until all *core objects* of all clusters are identified. During the last step of the revised DBSCAN, each *core object* is now assigned to its best *density-reachable* chain when all *density-reachable* chains that reach to the *core object* are known. A logical alternative is the closest density reachable chain

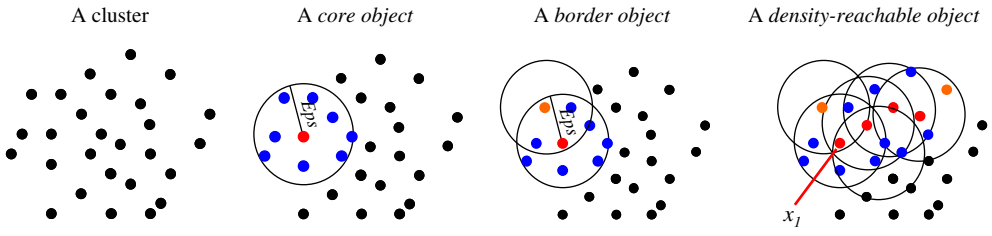

Fig. 1. Graphical presentation of key definitions explored in DBSCAN.

A

Algorithm *dbscan* (data, *Eps*, *MinPts*)

ClusterId = 1

For x_i in the data set
  If x_i is UNCLASSIFIED
    ExpandCluster (x_i , *ClusterId*)
    If *ExpandCluster* successful
      ClusterId = *ClusterId* + 1
    End
  End
End

End // *dbscan*

B

Algorithm *ExpandCluster* (x_i , *ClusterId*)

seeds = *Retrieve_Neighors* (x_i , *Eps*)

If |*seeds*| < *MinPts* // *density*(i)

  mark x_i as noise object
  Return without expansion success

Else

  // **STEP 1:** x_i is identified as a starting *core* object for *ClusterId*
  // **STEP 2: identify** all *density-reachable* objects

  assign all objects in *seeds list* to *ClusterId*
  delete object x_i from *seeds list*

  For all x_j in *seeds list*
    $N_{Eps}(x_j)$ = *Retrieve_Neighors* (x_j , *Eps*)
    If | $N_{Eps}(x_j)$ | ≥ *MinPts* // x_j is a *core object*

      // **further expansion only for a *core object***
      For all x_k in $N_{Eps}(x_j)$
        If x_k is UNCLASSIFIED or is NOISE
          If x_k is UNCLASSIFIED
            add x_k to *seeds list*
          End
          assign x_k to *ClusterId*
        End
      End

    End
  End

  Return with expansion success

  End

End // *ExpandCluster*

Fig. 2. Pseudocode of the DBSCAN algorithm: A) *dbscan* and B) *ExpandCoreCluster* routines.

(e.g. with the closest *core object* to the considered *border object*)—the *border object* is assigned to a cluster to which the *core-density-reachable* chain belongs.

The pseudocode routines of the revised DBSCAN are presented in Fig. 3A and B.

Matlab implementation of the revised DBSCAN algorithm is available upon request from corresponding author.

A

Algorithm *dbscan* (data, *Eps*, *MinPts*)

ClusterId = 1

For x_i in data set
  If x_i is UNCLASSIFIED
    ExpandCoreCluster (x_i , *ClusterId*)
    If *ExpandCoreCluster* successful
      ClusterId = *ClusterId* + 1
    End
  End
End

// Step 3:

For x_{border} in the *border list*
  $N_{Eps}(x_{border})$ = *Retrieve_Neighors* (x_{border} , *Eps*)
  assign x_{border} **to** *ClusterId* of the closest *core object* in $N_{Eps}(x_{border})$
End

End // *dbscan*

B

Algorithm *ExpandCoreCluster* (x_i , *ClusterId*)

seeds = *Retrieve_Neighors* (x_i , *Eps*)

If |*seeds*| < *MinPts* // *density*(i)

  mark x_i as noise
  Return without expansion success

Else

  // **STEP 1**: x_i is identified as a starting *core object* for
  // **STEP 2**: **identify** all *density-reachable* objects

  For all x_j in *seeds list*
    $N_{Eps}(x_j)$ = *Retrieve_Neighors* (x_j , *Eps*)
    If | $N_{Eps}(x_j)$ | ≥ *MinPts* // x_j is a *core object*

      // **further cluster expansion only for a *core object***
      assign x_j **to** *ClusterId* // revised code, assign only *core object* to *ClusterId*
      add all UNCLASSIFIED in $N_{Eps}(x_j)$ to *seeds list*
      label UNCLASSIFIED and NOISE objects in $N_{Eps}(x_j)$ as BORDER objects

      // Note: the noise within the neighborhood of a core object
      // is labeled as a border object

    End
  End

  Return with expansion success

  End

End // *ExpandCoreCluster*

Fig. 3. Pseudocode of the revised DBSCAN algorithm: A) *dbscan* and B) *ExpandCoreCluster* routines.

3. Results and discussion

3.1. Illustration of a problem

In [1] it was stated that an identified cluster is defined uniquely and contains objects which are *density-reachable* from any *core objects* of that cluster. However, the statement is only true for the *core objects*. The *border objects*, x_{border}, can be reached in the algorithm run (*density-reachable*) by different paths and from different *core objects* in the neighborhood $\in N_{Eps}(x_i)$ and thus many *density-reachable* chains may share the same *border object*.

In the area of between two adjacent clusters, one can expect a *border object* sharing the neighborhood *core objects* of two (or more) clusters. Such a *border object* can be reached via different *density-reachable* chains potentially originating from different clusters. Hence, the *border object* is assigned to the cluster discovered first—the object is connected via the first *density-reachable* chain of the cluster.

Since the first *core object* of each cluster is any object that fulfills the properties of the *core object*, discovering the order of clusters is arbitrary. As a consequence, assignment of a *border object* located between adjacent clusters is also arbitrary and leads to a non-robust clustering result for *border objects*. This issue is even more evident in the situation when neighborhood size increases thereby resulting in the detection of more *border objects*.

To illustrate this scenario, let us consider a simple data set containing two adjacent clusters. In order to visualize and examine the clustering results easily, the data set is simulated in a two-dimensional data space and contains 237 and 234 objects in each group, respectively. The distribution of data objects with an indication of the two groups of objects (blue and red points) is presented in Fig. 4A.

As was already mentioned in the theory section, the DBSCAN algorithm is sensitive to the order in which the objects are processed and the clustering result depends on the sequence in which the clusters are constructed. This issue is illustrated in Fig. 4B and C. Using the same settings of input parameters *Eps* (=0.6) and *MinPts* (=255), two different clustering results are obtained from DBSCAN. From Fig. 4B and C, it is apparent that the assignment of objects located in the space between two clusters is problematic. After a random shuffling of the order of objects in the data set, the final assignment of objects into two groups changes proving the drawback of DBSCAN (see Fig. 4C).

Depending on the input parameters, local densities of adjacent clusters and the order in which objects are processed, a misclassification issue exists—see Fig. 4B and C. By following two different visiting paths in the course of data clustering, the original DBSCAN algorithm results in the improper expansion of a cluster towards the second one.

3.2. Performance of the revised DBSCAN algorithm

In this paper, special attention is paid to improving the DBSCAN algorithm by modifying the *density-reachable* mechanism that drives the expansion of a cluster. As was mentioned earlier, in a *density-reachable* chain of objects $x_1, \ldots, x_n$ all objects $x_1, \ldots, x_{n-1}$ except object x_n are *core objects*. Following this property, the cluster expansion list (the so-called *seeds*—in the *ExpandCluster* routine) must include only *core objects* in order to extend the *density-reachable* chains.

The DBSCAN algorithm was originally developed to cluster spatial data—a geospatial data that are usually stored as coordinates and topology maps—and therefore, by nature, groups or patterns are expected to have a similar density. For a data set with well-defined clusters (groups of objects are far from each other), the clusters obtained from the revised DBSCAN algorithm are the same as clusters identified using the original DBSCAN because there is no *border object* sharing the *core-density-reachable* chains originating from two different clusters. However, the situation is different for a dense data set when many clusters are very close or even connected to each other.

The revised DBSCAN outperforms the original algorithm and provides an improved assignment of a *border object* to the expected cluster (upon having information from all surrounding clusters). This desired property of the revised DBSCAN algorithm is pictured in Fig. 5, where the clustering results for a simulated data are depicted.

Furthermore, the results of clustering obtained from the revised DBSCAN are also independent of the order in which clusters are discovered. To illustrate this important feature, objects of a simulated data set were clustered using the revised DBSCAN. Prior to the algorithm run, the order of the objects in the data set was rearranged randomly. Projections of objects onto a plane defined by two variables with an indication of the discovered clusters are presented in Fig. 5. Although group labels can change from run to run, the content of each cluster remains unchanged, which supports the conclusion that the revised DBSCAN algorithm successfully resolves the issue of *border objects* and their assignment.

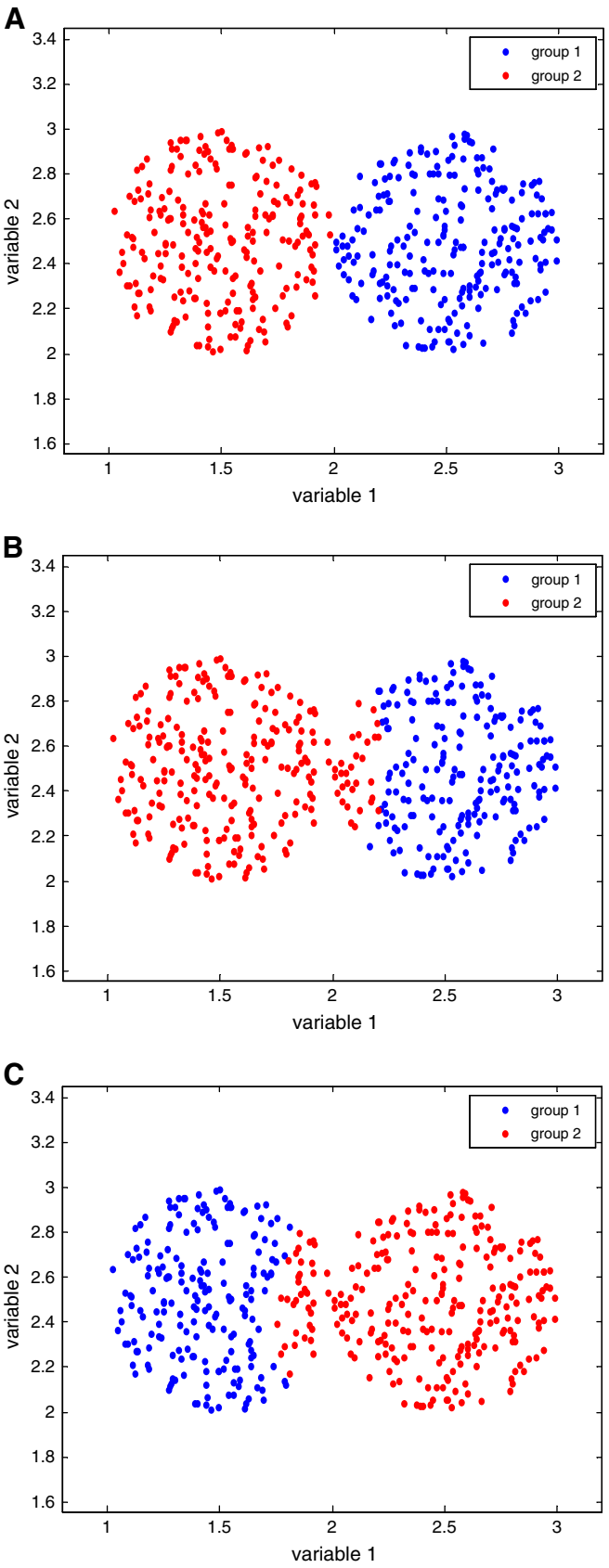

Fig. 4. A) Simulated data set containing two groups of objects; B–C) Results of DBSCAN clustering before (B) and after (C) random ordering of data objects (clusters were detected using parameters *Eps* = 0.6 and *MinPts* = 255) as input.

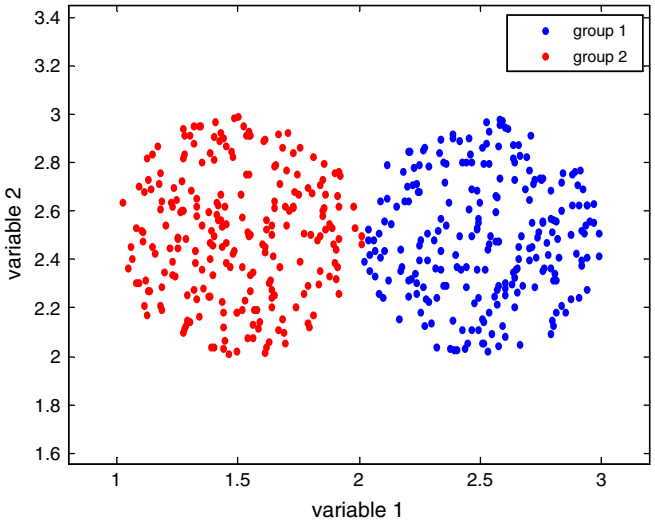

Fig. 5. Results of the revised DBSCAN simulated data set presented in Fig. 4A (*Eps* = 0.6 and *MinPts* = 255).

4. Conclusions

A modified version of the DBSCAN algorithm is proposed in this paper. It retains the key properties of the original DBSCAN algorithm, but in addition has the potential to improve clustering results by solving the issue of *border objects*. This is achieved by modifying the expansion step in which *core-density-reachable* chains, which contain only *core objects*, are used for clustering. After the detection of all clusters, *border objects* are assigned to their closest *core-density-reachable* chains that represent a given cluster. As illustrated on simulated data sets, the revised algorithm outperforms the original variant in a situation in which there is dense data with clusters close to each other. Moreover, the difference between the expansion concept used in the original and revised version of DBSCAN as well as the natural patterns approach are highlighted and discussed.

References

[1] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, 1996, pp. 226–231.
[2] D.P. de Oliveira, J.H. Garrett Jr., L. Soibelman, A density-based spatial clustering approach for defining local indicators of drinking water distribution pipe breakage, Advanced Engineering Informatics 25 (2011) 380–389.
[3] M. Daszykowski, B. Walczak, D.L. Massart, Looking for natural patterns in data: part 1. Density-based approach, Chemometrics and Intelligent Laboratory Systems 56 (2001) 83–92.
[4] L. Zhou, P.K. Hopke, P. Venkatachari, Cluster analysis of single particle mass spectra measured at Flushing, NY, Analytica Chimica Acta 555 (2006) 47–56.
[5] W. Zhao, P.K. Hopke, K.A. Prather, Comparison of two cluster analysis methods using single particle mass spectra, Atmospheric Environment 42 (2008) 881–892.
[6] S. Liu, Z.T. Dou, F. Li, Y.L. Huang, A new ant colony clustering algorithm based on DBSCAN, in: Proceedings of the 3rd International Conference on Machine Learning and Cybernetics, Shanghai, 2004, pp. 1491–1496.
[7] A. Ghosh, A. Halder, M. Kothari, S. Ghosh, Aggregation pheromone density based data clustering, Information Sciences 178 (2008) 2816–2831.
[8] C. Plant, S.J. Teipel, A. Oswald, C. Böhm, T. Meindl, J. Mourao-Miranda, A.W. Bokde, H. Hampel, M. Ewers, Automated detection of brain atrophy patterns based on MRI for the prediction of Alzheimer's disease, NeuroImage 50 (2010) 162–174.
[9] M.E. Celebi, Y.A. Aslandogan, P.R. Bergstresser, Mining biomedical images with density-based clustering, in: International Conference on Information Technology: Coding and Computing, Proceedings IEEE, 2005, pp. 163–168.
[10] M. Mete, S. Kockara, K. Aydin, Fast density-based lesion detection in dermoscopy images, Computerized Medical Imaging and Graphics 35 (2011) 128–136.
[11] J. Gong, C.H. Caldas, Data processing for real-time construction site spatial modeling, Automation in Construction 17 (2008) 526–535.
[12] T.N. Tran, R. Wehrens, L.M.C. Buydens, KNN-kernel density-based clustering for high-dimensional multivariate data, Computational Statistics and Data Analysis 51 (2006) 513–525.
[13] T. Aste, M. Saadatfar, T.J. Senden, Geometrical structure of disordered sphere packings, Physical Review E 71 (2005) 061302.
[14] X. Xu, J. Jager, H.-P. Kriegel, A fast parallel clustering algorithm for large spatial databases, Data Mining and Knowledge Discovery 3 (1999) 263–290.
[15] D. Arlia, M. Coppola, Experiments in parallel clustering with DBSCAN, in: Proc. of the 7th International Euro-Par Conference on Parallel Processing, 2001, pp. 326–331, (London, UK).
[16] R.J. Thapa, C. Trefftz, G. Wolffe, Memory-efficient implementation of a graphics processor-based cluster detection algorithm for large spatial databases, in: IEEE International Conference on Electro/Information Technology (EIT), 2010.
[17] E. Januzaj, H.-P. Kriegel, M. Pfeifle, Towards effective and efficient distributed clustering, in: Proceedings of the International Workshop on Clustering Large Data Sets, 3rd IEEE International Conference on Data Mining, 2003, pp. 49–58.