DYNAMIC DATA ASSIGNING ASSESSMENT CLUSTERING OF STREAMING DATA

O. Georgieva[1], F. Klawonn[2]

[1] *Institute of Control and System Research - Bulgarian Academy of Sciences, P.O. Box 79, 1113 Sofia, Bulgaria*
phone: +359 2 979 2052, e-mail: ogeorgieva@icsr.bas.bg

[2] *Department of Computer Science, University of Applied Sciences Braunschweig/Wolfenbuettel, Salzdahlumer Str. 46/48, D-38302 Wolfenbuettel, Germany*
phone: +49 5331 939 611, email: f.klawonn@fh-wolfenbuettel.de

Key Words: Objective function-based clustering, incremental clustering, noise clustering

Abstract: Discovering interesting patterns or substructures in data streams is an important challenge in data mining. Clustering algorithms are very often applied to identify single substructures although they are designed to partition a data set. Another problem of clustering algorithms is that most of them are not designed for data streams. This paper discusses a recently introduced procedure that deals with both problems. The procedure explores ideas from cluster analysis, but was designed to identify single clusters without the necessity to partition the whole data set into clusters. The new extended version of the algorithm is an incremental clustering approach applicable to stream data. It identifies new clusters formed by the incoming data and updates the data space partition. Clustering of artificial and real data sets illustrates the abilities of the proposed method.

1. Introduction

One of the main issues in data mining is the discovery of interesting structures, patterns or rules in data. Depending on the type of data and the intention of the analysis, different models and algorithms are available for these problems. For instance, association rule and frequent item set mining was originally designed for categorical data. The techniques can be extended to numerical data, but only for the price of higher computational costs. In this paper we focus on exploratory data analysis of numerical data. Cluster analysis is a very common technique to find structures in such data spaces [2,10,14,16,18]. However, the purpose of cluster analysis is to

partition the entire data set into “meaningful” substructures, whereas in many data mining problems there is no need for a partition of the whole data space. It is often sufficient to find a few interesting substructures that cover only a small proportion of the data set. Nevertheless, cluster analysis has become a popular tool for discovering substructures in the sense that most of the clusters are ignored and only the best ones are considered as important. For instance, in customer relationship management, one interesting question is customer segmentation, fitting perfectly to the concept of cluster analysis [5,15]. The significant task is to identify single groups of very typical customers without the necessity to assign all customers to clusters. Another typical example is the analysis of gene expression data where the biologist might not be interested in partitioning the whole set of genes of the considered organism. It is more important to find a few subgroups of genes with similar expression profiles within each group.

Statistical methods, as they are for instance proposed in [26], provide one possible strategy for the identification of interesting groups in data. However, apart from their high computational costs and that they are not well-suited for handling a continuous stream of incoming data, they follow a slightly different philosophy purely based on density considerations compared to the idealised concept of well-separated clusters.

Recently, a prototype-based clustering algorithm called Dynamic Data Assigning Assessment (DDAA) was proposed [11,18] whose intention is to identify single clusters step by step. It is based on the noise clustering technique [6,7] and finds single good clusters one by one and at the same time it separates the noise data. Two algorithm versions – hard and fuzzy clustering – are realisable according to the applied distance metric. The method can be used for two purposes: either in the sense of standard cluster analysis to determine the number of clusters automatically or in order to identify one or a few clusters that might cover only a part of the data set.

In classical data analysis it is usually assumed that a data set is first collected completely and then the analysis is carried out. However, in data mining it is very common that we do not have a fixed data set, but a constantly growing amount of data coming in as a more or less constant stream. A possible way to analyse such data is to restart the corresponding algorithm completely, each time new data is arriving. However, this approach is neither very efficient nor suited to detect changes in the data. There is no explicit indication, when for instance a new cluster is built while the data set is updated. In data mining there are various newer approaches suited to analyse a stream of incoming data directly [5,15,20]. Single pass clustering, as it is for

instance described in [12,19,25], comprises techniques that find clusters by a single pass through the data set, in contrast to iterative strategies like k-means clustering. In this way, single pass clustering can be applied to large data sets due to the low computational costs as well as to streaming data. It should be noted that we are interested in finding interesting patterns in one data stream. We do not want to cluster data streams as it is proposed in [25].

In this paper, we extend the original DDAA algorithm to detect single clusters in streaming data. In contrast to single pass clustering our aim is not to partition the data stream into clusters, but to discover interesting patterns in terms of single clusters that might cover only a small proportion of the full data stream. In the DDAA algorithm it is possible that a large part of the data is considered as noise in the sense that it is not assigned to any cluster. The new evolving DDAA algorithm assigns every new data point to an already determined good cluster or alternatively to the noise cluster. In this way, it checks whether the new data collection provides one or more new good clusters and thus changes the data structure. The assignment can be done in a hard or fuzzy sense. It should be emphasized that the identification of single clusters, even if the majority of the data is considered as noise, is the main purpose of our approach. Other algorithms, like DBSCAN [8] or its incremental version [9] take noise into account as well. However, DBSCAN defines clusters in terms of a homogeneous minimum density, i.e. the user must specify two parameters *Eps* and *MinPts*. For each point in a cluster the *Eps*-neighbourhood must contain at least *MinPts* points, otherwise the point is considered as noise. This means that the definition of noise is homogeneous in the whole data space and the decision which points are marked as noise or outliers strongly depends on the setting of these parameters. In our approach we use a dynamic noise distance, which has a different interpretation than the parameter *Eps* in DBSCAN, but allows for more flexibility in the detection of noise or outliers. In [5] an approach for clustering streaming data is proposed based on fuzzy cluster analysis. The focus there is again to partition the data set into (fuzzy) cluster, not as an approach to identify only a few interesting clusters among a large amount of noise data.

The paper is organised as follows. The second section briefly reviews the necessary background on objective function-based clustering, the concept of noise clustering that we exploit in our approach and the underlying idea of the DDAA algorithm itself. At the end of the section, the well-known wine recognition data set is considered for illustration purposes. The evolving version of the DDAA algorithm is presented in detail in Section 3. In Section 4 we discuss a case study based on weather data. Comparison analysis is provided in section 5. Final conclusions are

provided in the sixth section.

2. Basic concepts and the DDAA algorithm

2.1. Objective function-based clustering

Objective function-based clustering aims at the minimization of an objective function J that indicates a kind of fitting error of the clusters to the data. In this objective function, the number of clusters has to be fixed in advance. The underlying objective function for most of the clustering algorithms is [2]:

$$J = \sum_{i=1}^{c} \sum_{k=1}^{N} (u_{ik})^m d_{ik}^2, \tag{1}$$

where N is the number of data points; c is the number of clusters; u_{ik} and d_{ik} denote correspondingly the membership degree and the distance of the k-th data point $x_k = [x_{k1}, x_{k2}, ..., x_{kn}]$, k = 1, ..., N, to the i-th cluster prototype, i = 1, ..., c. The fuzzifier $m \in [1,\infty)$ is the weighted exponent coefficient which determines how much clusters may overlap. When we choose the fuzzifier m=1, we have $u_{ik} \in \{0,1\}$ at a minimum of the objective function (1) and the resulting partition will be crisp.

In order to avoid the trivial solution assigning no data to any cluster, i.e. setting all u_{ik} to zero, and to avoid empty clusters, the following constraints are introduced:

$$u_{ik} \in [0,1], \qquad 1 \le i \le c, \qquad 1 \le k \le N \tag{2.a}$$

$$\sum_{i=1}^{c} u_{ik} = 1, \qquad 1 \le k \le N \tag{2.b}$$

$$0 < \sum_{i=1}^{N} u_{ik} < N, \qquad 1 \le i \le c \tag{2.c}$$

The parameters to be optimized are the membership degrees u_{ik} and the cluster parameters which finally determine the distance values d_{ik}. Each cluster is represented by a cluster prototype. In the simplest case, the cluster prototype is a single vector called cluster centre $v_i = [v_{i1}, v_{i2}, ..., v_{in}]$, $i = 1, ..., c$. The distance of a data point x_k to the i-th cluster is defined by a positive definite symmetric matrix A_i and the cluster centre as follows:

$$d_{ik}^2 = \|x_k - v_i\|_{A_i}^2 = (x_k - v_i) A_i (x_k - v_i)^T. \tag{3}$$

The matrix A_i is defined differently according to the applied objective function-based clustering algorithm [1,2,10,13,14,17] and determines the cluster shape and orientation.

2.2. Noise clustering

Arbitrary noise points that do not belong to any comprehensible cluster have to be taken into account. The successful solution to deal with noise in the data set is to collect the noise points in one single cluster [6]. For this purpose a virtual noise prototype with no parameters to be adjusted is introduced. It has always the same (large) distance δ to all points in the data set.

Let cluster number *c* be the noise cluster. Then, by definition [6] we have

$$d_{ck} = \delta, \ \forall\, k. \tag{4}$$

The remaining *c*-1 clusters are assumed to be the good clusters in the data set. The objective function J_{noise} that considers the noise cluster is defined in the same manner as in the general scheme for the clustering minimization functional (1) i.e. $J_{noise} \equiv J$, with some additional specifications. The distances for each point x_k, $k = 1,...,N$ are defined by (3) for all clusters i, $i = 1,..., c-1$ and by

$$\mathrm{d}_{ck}^2 = \delta^2 \text{ for } i = c. \tag{5}$$

The objective function J_{noise} has the global minimum for a fixed noise distance δ only if:

a) for hard noise clustering (i.e. $m = 1$) the membership degrees are:

$$u_{ik} = 0 \text{ for } \forall\, i \neq j \text{ and} \tag{6}$$

$$u_{jk} = 1 \text{ for } j \text{ such that } d_{jk} = \min(d_{ik},\ i = 1,...,c).$$

b) for fuzzy noise clustering ($m > 1$) the membership degrees are

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} (d_{ik} / d_{jk})^{2/(m-1)}} \tag{7}$$

(except for the rare case that a zero distance occurs, in which case the data point is assigned to a cluster to which it has zero distance) and the cluster centres of the good clusters are defined by the weighted mean value:

$$v_i = \frac{\sum_{k=1}^{N} (u_{ik})^m x_k}{\sum_{k=1}^{N} (u_{ik})^m}, \quad \text{for } i = 1, ..., c-1. \tag{8}$$

The specification of the noise distance δ is a matter of consideration for the particular data set. If δ is chosen too small then most of the data points will be classified as noise, while for a large δ value even outliers will be assigned to good clusters.

Let us assume a special case when the data set consists of only one good cluster among a certain number of noise data considered as a noise cluster. Thus, for a certain δ value the two clusters could be separated in minimizing the objective function J_{noise}, which is simplified for the particular case of c=2 to the following form:

$$J_{noise} = \sum_{k=1}^{N} (u_{1k})^m d_k^2 + \delta^2 \sum_{k=1}^{N} (u_{2k})^m \ . \tag{9}$$

The distance d_k denotes the distance between data point x_k and the centre v_1 of the single good cluster. The membership degrees are calculated for the fixed δ as:

a) for hard noise clustering

$u_{1k} = 1$ and $u_{2k} = 0$ if the k-th point belongs to the good cluster, i.e. $d_k \le \delta$ and (10)

$u_{1k} = 0$ and $u_{2k} = 1$ if the k-th point belongs to the noise cluster, i.e. $d_k > \delta$. (11)

b) for fuzzy noise clustering the membership degree to the good cluster is defined as

$$u_{1k} = \frac{1}{1 + \left(\frac{d_k}{\delta} \right)^{2/(m-1)}} \tag{12}$$

and the membership degree to the noise cluster is correspondingly defined as:

$$u_{2k} = \frac{1}{1 + \left(\frac{\delta}{d_k} \right)^{2/(m-1)}} \ . \tag{13}$$

Since we are still in the framework of probabilistic clustering the following statement is valid for both clustering variants:

$$u_{1k} = 1 - u_{2k} \ , \quad k = 1,...,N \ . \tag{14}$$

The method presented below separates one cluster at a time based on the concept of noise clustering via a dynamic decrease of the noise distance.

2.3. DDAA algorithm

The procedure starts by choosing a large noise distance, for instance the diameter of the data set, so that all data points are assigned to the good cluster and no data are considered as noise. Then, decreasing the noise distance stepwise by a prescribed decrement $\Delta\delta$, for each $\delta = \delta_j$ value we can determine the number $N_{in}(j)$ of data belonging to the good cluster according to their membership degrees – crisp (10) or fuzzy (12) – for the crisp and fuzzy algorithm versions, respectively. The index j denotes the current step of the noise distance reduction. Obviously, decreasing the distance δ, a process of "loosing good" data, i.e. assigning data to the noise cluster will begin. Continuing to decrease the noise distance, we will start to separate points from the good cluster and add them to the noise cluster, until the good cluster will be entirely empty as all data will be assigned to the noise cluster.

The described dynamics of moving data from the good cluster to the noise cluster can be characterised by a curve $N_{in}(j)$ showing the number of data points assigned to the good cluster over the varying noise distance. The velocity $\Delta N_{in}(j)$ via the noise distance alteration is also evaluated:

$$\Delta N_{in}(j) = N_{in}(j-1) - N_{in}(j). \tag{15}$$

Note that not the time is used for the δ-axis and so these two curves should be viewed from right to left, when we want to observe the behaviour over time. By slightly decreasing the noise distance with decrement $\Delta\delta$=0.003 two curves are obtained (Figure 1.b) for the artificial data set given in Figure 1.a.

It is clear that if we loose actual noise points (i.e. from a region of low data density) the curve $N_{in}(j)$ will almost remain in a plateau, whereas a strong slope should be observed when data from an actual cluster (with higher density than the noise data) are moved to the noise cluster. In the general case of a complex data set, we will have a number of plateaus and a number of strong slopes in the curve. The peaks obtained from the curve ΔN_{in} correspond to the slopes in the N_{in} curve. The area of every peak is proportional to the number of points that are separated to the noise cluster within the current slope. Only significant peaks whose area is larger than a predefined threshold *tol* are of real interest:

$$S(s) = \sum_{j:\delta_{\min}(s) \le \delta_j \le \delta_{\max}(s)} \Delta N_{\text{in}}(j) > tol\,, \tag{16}$$

where $\delta_{\min}(s)$ and $\delta_{\max}(s)$ are correspondingly the left and right base of the peak *s*.

The last significant peak (the left-most one in the $\Delta N_{\text{in}}(j)$ curve) occurs when the data points of the last data cluster are moved to the noise cluster. These remaining data points define a good cluster that we remove from the whole data set. The other significant peaks also correspond to phases where at least one comprehensible cluster is shifted to the noise cluster. Data corresponding to insignificant peaks should be considered as noisy data. The whole procedure could be applied again over the reduced data set and repeated until no more significant peaks are identified. For instance, the three significant peaks in the bottom assignment curve in Figure 1.b correspond to the three good clusters (Figure 1.c) that are selected one by one in the respective three passes of the clustering procedure.

In the context of fuzzy noise clustering the membership degree is calculated by equation (12) and the data point is assumed to belong to the good cluster if its membership degree is larger than a predefined value μ_{tol} . However, being stricter in the identification of proper clusters, the prescribed threshold membership degree should be chosen larger and for a more tolerant identification, it should be chosen smaller.

The Dynamic Data Assigning Assessment cluster identification algorithm is summarised as follows:

Step 1 Specify the decrement step $\Delta\delta$ and threshold *tol*. If fuzzy DDAA clustering is applied specify μ_{tol} .

Step 2 Compute the curves N_{in} and ΔN_{in} decreasing δ by the prescribed decrement $\Delta\delta$.

Step 3 Find all peaks of ΔN_{in} curve and select the significant peaks according to (16).

Step 4 Separate one good cluster determined by the last significant peak.

Step 5 Subtract the separated points from the data set and repeat the procedure from ***Step 2*** for the remaining data points until no significant peaks could be found.

As in most clustering algorithms we normalise the data set in advance in order to let each feature have approximately the same influence on the distance used for clustering. The choice of the decrement parameter $\Delta\delta$ depends on the density of the particular data set. It should be varied in a certain range and an appropriate value is chosen such that at least two significant peaks in the ΔN_{in} curve are provided. Usually (our experience covers several real world data sets) successful separation can be found for values from 0.01 to 0.001. Commonly, a more dense data set needs a

lower decrement value. The other important parameter – the threshold *tol* –can be determined based on assumptions on the minimal amount of data that form a cluster or analysing the two dynamics curves. It is chosen approximately equal to the number of points that form the last significant peak. In the fuzzy DDAA version μ_{tol} is set by default to 0.5 which gives satisfactory results. However, application of specific knowledge can improve the selection results.

2.4. Clustering of wine data

In this subsection, clustering of the wine recognition data [23] is applied in order to illustrate the abilities of the DDAA algorithm. The wine data set is the result of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined quantities for 178 instances characterized by 13 features and found in each of the three types of wines.

The parameters of the hard DDAA algorithm were set by analysing the ΔN_{in} curve obtained in the first algorithm pass for the normalised data set. The value of the decrement parameter was defined as $\Delta\delta$=0.01, resulting in three significant peaks in the ΔN_{in} curve. The threshold *tol*=18 allows to select the three clusters in three algorithm passes (Figure 2). The first cluster is separated at δ=0.46 (Figure 2.a), the second – at δ=0.52 (Figure 2.b) and the third one at δ=0.48 (Figure 2.c). All points clustered by the DDAA method except one of the third cluster were successfully assigned to the corresponding classes (Table 1). This means that the obtained clusters and calculated cluster centres provide a reliable data partition. A relatively large amount of 69 non-clustered points is assigned to the noise cluster. As we do not have information about the cluster shape we are searching for hyper-spherical clusters according to the applied Euclidean distance metric. However, the real cluster shape is more complex which is "invisible" for our algorithm. There is an indication [22] that the clusters (classes) in the wine data set are of hyper-ellipsoidal shape. The DDAA algorithm can be extended in a straight forward manner to other cluster shapes, simply by replacing the cluster prototype update scheme (8) by the scheme for the corresponding cluster shape.

3. Evolving DDAA

Let us suppose that the DDAA algorithm was applied to the data set $X=\{x_k\}$, $k=1, \ldots, N$ as it was described in the previous section. Thus, c-1 good clusters and their cluster centres have been found. The noise data are collected in one additional noise cluster. Let us also suppose that a new input data stream is currently added to the data set. The task is to find the proper assignment

of the new incoming points, entering them one by one, to the already determined data structure or alternatively to account for a new structure as a result of the incoming information.

In general, there are three possibilities for every new coming data point x_s: first, the new data point could belong to a good cluster or secondly, it can be a noise data point. In case of fuzzy clustering the point belongs to all clusters – good and fuzzy, but with different membership degrees. Finally, the new stream data can change the structure by forming additional clusters. As we restrict the clustering to the case when the characteristics of the data stream will not change significantly over time, we do not consider the case of elimination of clusters due to disappearing concepts, since we had to rebuild them again later on.

The below developed evolving procedures enable us to detect the following possible changes in the data structure:

- Movement of existing clusters;
- Creation of new clusters;
- Merging of clusters;

3.1. Hard evolving DDAA

First, the solution will be discussed in the hard clustering variant. Let us suppose that we have a fixed threshold δ_h that is a distance limit value under which a given point is specified as a part of the good cluster. We propose the following procedure except in case when it is context-determined. The threshold is calculated equal to the minimal cluster radius:

$$\delta_h = \min_{i=1,\ldots,c-1}(r_i), \tag{17}$$

where r_i is the cluster radius of the i-th cluster defined as the maximal distance between the cluster centre v_i and the points belonging to this cluster:

$$r_i = \max_{x_j \in i-\text{th cluster}} \left\| v_i - x_j \right\|_{A_i} . \tag{18}$$

In order to evaluate the membership of the current point to a cluster we need to assess the distance $d_s(i)$ between the new coming data point x_s and each cluster centre v_i, $i = 1,\ldots, c$-1 calculated in the sense of the distance measure applied in the DDAA algorithm:

$$d_s(i) = \sqrt{(v_i - x_s)^2_{A_i}} \quad i = 1,\ldots,c-1. \tag{19}$$

The data point x_s is assigned to the closest good cluster, for which the distance (19) is minimal

and simultaneously less than the threshold δ_h. Otherwise, x_s cannot be assigned to any good cluster and then it is added to the noise cluster. In this case we should presume that the new data point could change the data structure. In order to check whether a new good cluster is formed within the data classified as noise, we run the DDAA algorithm for the new-formed noise cluster. Additionally we apply a procedure of cluster merging in case some of the clusters in the updated partition are closer than the prescribed threshold *dv_tol*. For this purpose we calculate the distances between all cluster centres and compare them to *dv_tol*. The threshold *dv_tol* could be determined by different strategies as:

- according to some particular knowledge of the considered data set;

- $dv\_tol = 2\delta_h$ i.e. we presume that the threshold is restricted by the diameter of the smallest cluster provided in the original data set;

The assignment of each new stream point x_s to the already known data partition is accomplished according to the following procedure:

Step 1 Insert x_s to the data set, $N=N+1$. Calculate $d_s(i)$ by (19).

Step 2 Find $d_p = \min_{i=1,...c-1}(d_s(i))$, i.e. the closest cluster p.

Step 3 If $d_p \le \delta_h$

- Assign x_s to the cluster p.
- Update the position of the existing cluster centres applying formula (8).

else $d_p > \delta_h$

- Assign x_s to the noise cluster.
- Apply the DDAA algorithm to the newly obtained noise cluster.
- If a new good cluster is separated, update the data structure:
  - Increase the number of clusters $c=c+1$;
  - Update the position of the existing cluster centres applying formula (8).

Step 4 Calculate the distances between cluster centres of all determined good clusters:

$$dv(i,j) = \sqrt{(v_i - v_j)^2}, \quad i,j = 1,...,c-1.$$

Step 5 If $dv(i,j) < dv\_tol$ then merge the i-th and j-th cluster, $i,j = 1,...,c-1$:

- Decrease the number of clusters $c=c-1$;
- Update the position of the newly defined cluster centre applying formula (8) for all points belonging to the merged clusters.

3.2. Fuzzy evolving DDAA

If fuzzy clustering is applied then the procedure is carried out according to the membership degree $u_s(i)$, $i = 1, \ldots, c$ of x_s to every cluster present in the data set. Let us suppose that a threshold membership degree u_h is given in advance. It could be chosen equal to the predefined value μ_{tol} used in the fuzzy DDAA algorithm. The assignment procedure is rewritten as follows:

Step 1 Insert x_s to the data set, $N=N+1$. Calculate $d_s(i)$ by (19);

Step 2 Find $d_p = \min_{i=1,\ldots,c-1}(d_s(i))$, i.e. the closest cluster p.

Step 3 Find u_p according to (7) and for the fixed $\delta=\delta_h$.

Step 4 If $u_p \geq u_h$

- Assign x_s to the cluster p.
- Update the position of the existing cluster centres applying formula (8).

else $u_p < u_h$

- Assign x_s to the noise cluster.
- Apply the DDAA algorithm to the newly obtained noise cluster.
- If a new good cluster is separated update the data structure:
  - Increase the number of clusters $c=c+1$;
  - Update the position of the existing cluster centres applying formula (8).

Step 5 Calculate the distances between cluster centres of all determined good clusters:

$$dv(i,j) = \sqrt{(v_i - v_j)^2}, \quad j = 1,\ldots,c-1.$$

Step 6 If $dv(i,j) < dv\_tol$ then merge the i-th and j-th cluster:

- Decrease the number of clusters $c=c-1$;
- Update the position of the new defined cluster centre applying formula (8) for all points belonging to the merged clusters.

If the data are grouped into complex clusters that have different shape, orientation and density then the assignment thresholds δ_h, u_h and the respective value *dv_tol* will vary in wide ranges for the distinct clusters. It will not be easy to set a proper assignment threshold neither for

hard, nor for the fuzzy clustering variant. In such case the evolving algorithm is organised so that the DDAA algorithm applied in the respective steps 3 and 4 in the evolving DDAA procedures is spread and applied over the current input data point x_s, the data of the closest good cluster p and all noise data. The separated good cluster(s) update(s) the data structure. However, this will increase the computational costs.

The clustering capabilities of the evolving extension of the DDAA algorithm are demonstrated on the artificial data set processed in Section 2 (Figure 1.a). The new data stream is added point by point to the data set (Figure 3.a). The first seven new points do not change the existing data partition and are added to the noise cluster according to the set threshold δ_h =1.5 obtained by equations (17) and (18). After the 8^{th} data point a new cluster – fourth for the considered data set – is separated by the DDAA algorithm. The partition is updated by adding the new cluster (Figure 3.b).

4. Application to weather data

The case study data set comes from a weather database containing three specific air and wind features that were measured hourly at the small German island Helgoland during the first months of the year 1997. The features are air pressure, wind direction and wind strength. The wind direction is measured in 10 degrees steps. In order to obtain the angle of the wind direction, we have to multiply the corresponding value of the data set by 10. The wind with angle direction 0 (or equivalently 360) degrees corresponds to wind from the north direction. The angle is measured counter-clockwise.

Although the data set is already complete, in order to demonstrate how our method can be applied, we assume that only the data from the first month (January) are available, altogether 744 data points. For this relatively small data set the DDAA algorithm parameters are determined easily. The decrement parameter $\Delta\delta = 0.003$ was defined in such a way that significant peaks in the ΔN_{in} curve are provided: it was varied in the range from 0.01, when only one cluster is selected, to 0.002 with many small peaks and one large peak, not leading to a meaningful clustering result. The other important parameter – threshold *tol* – was determined according to the N_{in} curve (Figure 4). The threshold value $tol = 25$ was chosen approximately equal to the number of points that are assigned to the noise cluster through the last significant peak (the left most in the ΔN_{in} curve) whose right base corresponds to $\delta = 0.115$. Six clusters that separate 627 data

points and one noise cluster of 117 data points were selected. The coordinates of the cluster centres are given in Table 2 and the distribution of the assigned number of data in Table 3.

According to the clustering results, the weather in January at the considered area is characterized by 6 typical weather types that corresponds to the six good clusters. Bearing in mind that the minimal and maximal measured air pressure of the whole data set are 9711 and 10403, respectively, the clusters indicate medium (clusters N^o 1,2,3,6) and high (clusters N^o 4 and 5) pressure (Table 2). The wind strength in January was relatively small and does not change significantly throughout the clusters. Just in one group (cluster 6) it is in the middle of its interval of possible values for the whole data set. The wind direction is the parameter with the highest variation: the first and second cluster comprise data that are characterised by southern (S) wind, the third and fourth by eastern (E) wind, whereas for the fifth cluster the wind is western-northern (WN) and for the sixth cluster eastern-southern (ES) wind.

The 132 data points measured subsequently have been treated as a stream data set. The evolving DDAA algorithm is applied to them in order to determine the data partition. The threshold $\delta_h = 40$ value is set (almost) equal to the minimal distance equal to 41.916 between the already defined cluster centres. As a result of the clustering a seventh cluster with cluster centre $v_7 = (10279, 25, 43)$ is separated (Figure 5). The new cluster comprises data of the first few days of February as well as some noise data that had not been assigned to any good cluster in the data from January. Despite the new found cluster and cluster 5 are close to each other, we keep them separated as the new cluster recognizes weather that is characterised by high pressure, western wind and relatively low wind strength. At the same time, the already selected clusters have been extended by new data, which can be seen in Table 3.

These results fit well to the real meteorological conditions of the considered area. This is a zone where some cold winter periods exist in winter caused by either eastern or northern winds. The northern winds usually come along with high pressure – cold, but sunny, no strong wind. They are more typical for late winter (February). Thus, the seventh cluster found in the new data stream presents a transitional period to the late winter weather conditions.

5. Comparison analysis

Most of the objective function-based clustering algorithms seeking for all clusters at once are based on validity measures to assess the quality of the partition. They depend on the algorithm initialisation. The algorithms will converge to a local minimum, in the worst case to a saddle point. For the global minimum or a good local minimum, the computed clusters should

correspond to the good or true clusters in the data set [1,2,3,7,21]. The main difference of our algorithm lies in the strategy of searching for a good cluster accomplished at every algorithm pass. We do not try to minimize the standard objective function. We have a more local view on clusters separated at a certain level of the noise distance. Thus, we do not choose a constant δ, but vary δ from an initial large value to a small (usually zero) value.

Another comparison could be done over a clustering approach based on the evolving, distance-based partitioning methods [4,20,24]. One of the quite frequently applied clustering methods in the last decade is the subtractive clustering proposed by Chiu [4], which is an improved version of the mountain clustering method introduced by Yager and Filev [24]. In subtractive clustering the clusters are selected one by one according to the estimated potential of a point to be a cluster centre. A data point with many neighbouring data points will have a high potential value. Four clustering parameters need to be properly adjusted in advance in order to obtain a reliable data partition – cluster radius, squash factor, accepting and rejecting rate. Although the preferable values of the clustering parameters are mentioned in [4], experience shows that they should be fine tuned according to the particular data set.

In contrast to subtractive clustering the adjusted parameters in the hard DDAA algorithm are only two and in case of fuzzy DDAA one more parameter μ_{tol}. The most important parameter is the decrement value $\Delta\delta$ that specifies the reduction rate of the noise distance and strongly depends on the density of the searched clusters. The second parameter is the threshold *tol* that determines the number of points above which a defined peak is selected as a significant one. An important advantage of the DDAA algorithm is the possibility to visualise the dynamics of the assigned data points to the good cluster. Both curves – the acceptance rate and its velocity – can be plotted independently of the data dimensionality. One can easily adjust the proper threshold value *tol* simply by looking at the significant steepness of the slopes. In this sense, we do not claim that the DDAA algorithm performs better than others, but its few control parameters can be handled in a very intuitive way using simple visualisations.

The key of subtractive and DDAA clustering methods is that they do not involve any iterative optimisation and thus, the computation grows only linearly with the dimension of the problem (number of data as well as attributes). It should be mentioned that the DDAA algorithm incorporates relatively simple mathematical formulae. We are also free in the choice of the cluster prototype for every distinct good cluster to be identified. We can also use more complicated prototypes for more flexible cluster shapes as they are very often found in objective

function-based clustering.

6. Conclusions

The cluster identification method presented here is based on the assessment of the dynamics of the number of points that are assigned to only one cluster through noise clustering of the data set by slowly changing the noise distance from a reasonably large to a sufficiently small value. It successfully assigns the new input data stream to the already known data structure or discovers new interesting groups of the data set that currently appeared. Two algorithm variants – hard and fuzzy – are presented in parallel. Additionally two alternatives – when there is or there is no preliminary information about the assignment thresholds – are considered.

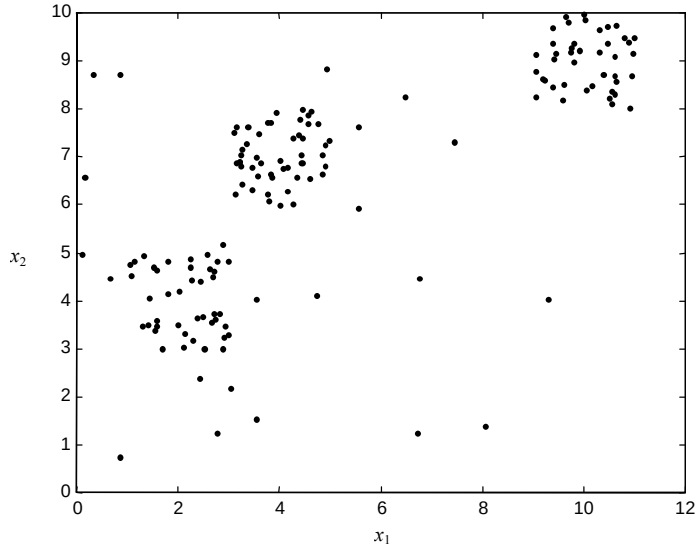
Acknowledgements

We would like to thank Deutsche Wetterdienst (DWD) for providing the weather data set for our research purposes.

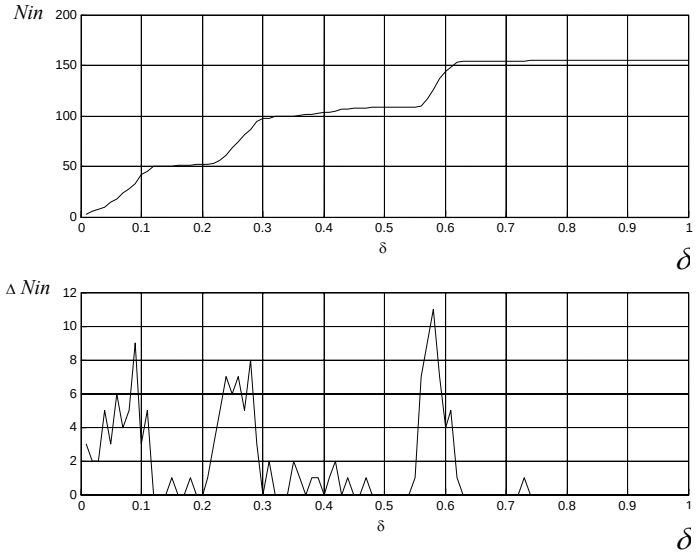
References

[1] R. Babuska, Fuzzy Modeling for Control (Kluwer Academic Publishers, Boston, 1998).
[2] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms (Plenum Press, New York, 1981).
[3] J.C. Bezdek, A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence 2 (1980) 1-8.
[4] S.L. Chiu, Fuzzy Model Identification Based on Cluster Estimation, Journal of Intelligent and Fuzzy Systems 2 (1994) 267-278.
[5] F. Crespo and R. Weber, A Methodology for Dynamic Data Mining based on Fuzzy Clustering. Fuzzy Sets and Systems 150 (2005) 267-284.
[6] R.H. Davé, Characterization and Detection of Noise in Clustering, Pattern Recognition Letters 12 (1991) 657-664.
[7] R.H. Davé and R. Krishnapuram, Robust Clustering Methods: A Unified View, IEEE Transactions on Fuzzy Systems 5 (1997) 270-293.
[8] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (Portland, OR, 1996) 226-231.
[9] M. Ester, H.-P. Kriegel, J. Sander and M. Wimmer, X. Xu, Incremental clustering for mining in a data warehousing environment, in: Proc. 24th Int. Conf. Very Large Databases, VLDB (1998) 323-333.
[10] I. Gath and A.B. Geva, Unsupervised Optimal Fuzzy Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 7 (1989) 773-781.
[11] O. Georgieva and F. Klawonn, Cluster Analysis via the Dynamic Data Assigning Assessment Algorithm, Information Technologies and Control 2/2006, 14-21.
[12] C. Gupta, R.L. Grossman, GenIc: A single pass generalized incremental algorithm for

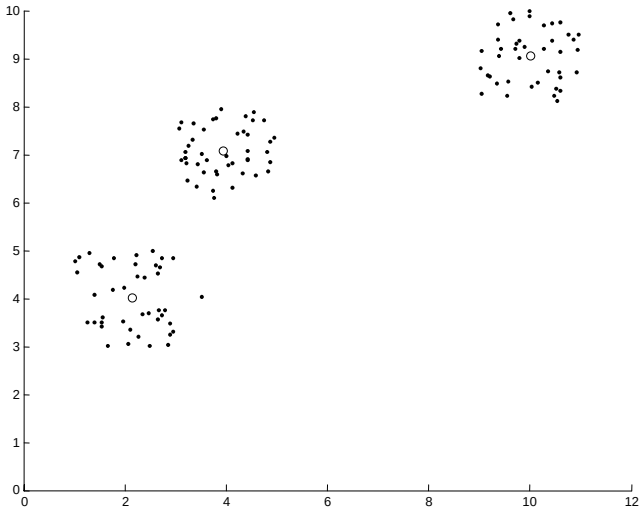
clustering, in: 2004 SIAM International Conference on Data Mining, SDM 04 (SIAM, Philadelphia, 2004) 137-153.
[13] D.E. Gustafson and W.C. Kessel, Fuzzy Clustering with a fuzzy covariance matrix, in Proc. IEEE CDC (IEEE, San Diego, 1979) 761-766.
[14] F. Höppner, F. Klawonn, R. Kruse and T. Runkler, Fuzzy Cluster Analysis (John Wiley & Sons, Chichester, 1999).
[15] G. Hulten, L, Spencer and P. Domingos, Mining time-changing data streams, in: Proc. Of KDD 2001 (ACM Press, New York, 2001) 97-106.
[16] J.-M. Jolion, P. Meer and S. Bataouche, Robust Clustering with Applications in Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (1991) 791-802.
[17] A. Keller and F. Klawonn, Adaptation of cluster sizes in objective function based fuzzy clustering, in: C.T. Leondes, ed., Intelligent Systems: Technology and Applications vol. IV: Database and Learning Systems (CRC Press, Boca Raton, 2003) 181-199.
[18] F. Klawonn and O.Georgieva, Identifying single clusters in large data sets, in: J. Wang, ed., Encyclopedia of Data Warehousing and Mining. (Idea Group, Hershey, 2006) 582-585.
[19] R. Papka and J. Allan, On-line new event detection using single pass clustering, UMASS Computer Science Technical Report 98-21 (1998).
[20] Q. Song and N. Kasabov, Dynamic evolving neuro-fuzzy inference system (DENFIS): On-line learning and application for time-series prediction, in: Proc. of the 6 th International Conference on Soft Computing (Iizuka, 2000) 696-702.
[21] C.V. Stewart, MINPRAN: A New Robust Estimator for Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 925-938.
[22] H. Timm, Fuzzy-Clusteranalyse: Methoden zur Exploration von Daten mit fehlenden Werten sowie klassifizierten Daten (in German). Ph.D. Thesis, Faculty of Computer Science, University of Madgeburg, 2002.
[23] Wine database, http://www.ics.uci.edu/~mlearn/databases/wine/
[24] R. Yager and D. Filev, Essentials of Fuzzy Modeling and Control (John Wiley & Sons, Chichester, 1994).
[25] J. Yang, Dynamic clustering of evolving streams with a single pass, in: Proc. 19th International Conference on Data Engineering, ICDE'03 (Bangalore, 2003), 695-697.
[26] Z. Zhang, D.J. Hand, Detecting groups of anomalously similar objects in large data sets, in: A.F. Famili, J.N. Kook, J.M., Peña and A. Siebes, eds., Advances in Intelligent Data Analysis VI (Springer, Berlin, 2005) 509-519.



a) Data set

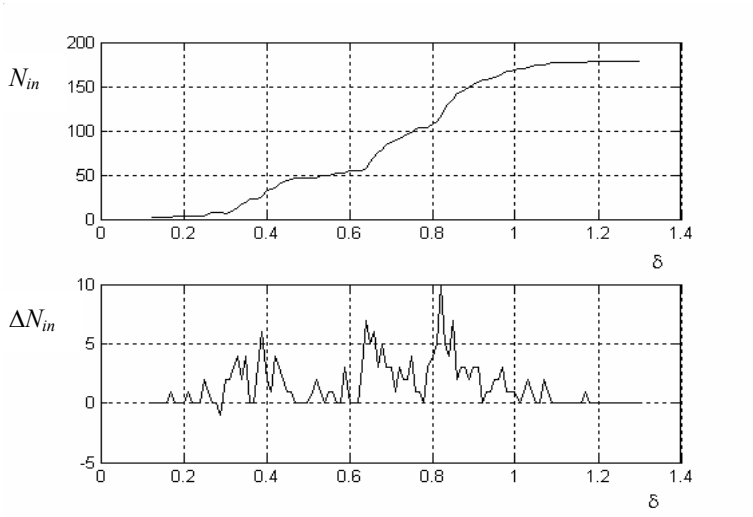


b) Data assignment dynamics in the first algorithm pass via the normalized data set

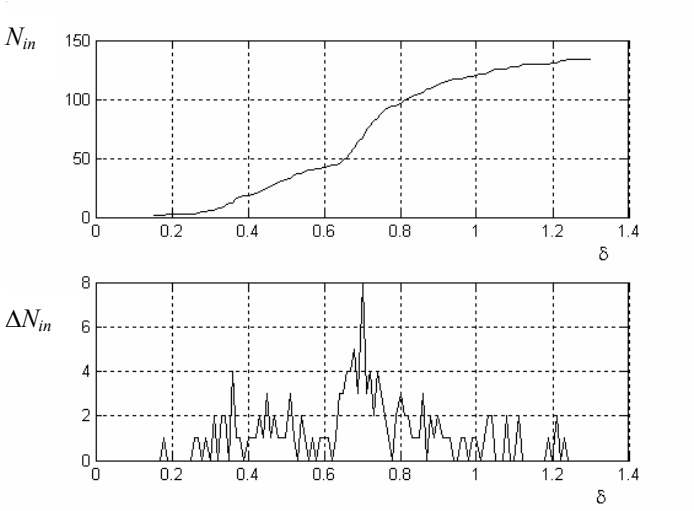


c) The three clusters determined corresponding to three significant peaks; circles are cluster centres

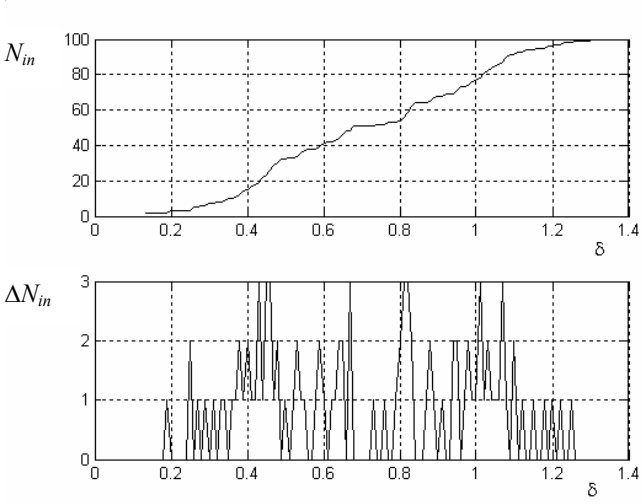
Figure 1. DDAA clustering algorithm applied to the artificial data set



a) Data assignment dynamics in the first algorithm pass

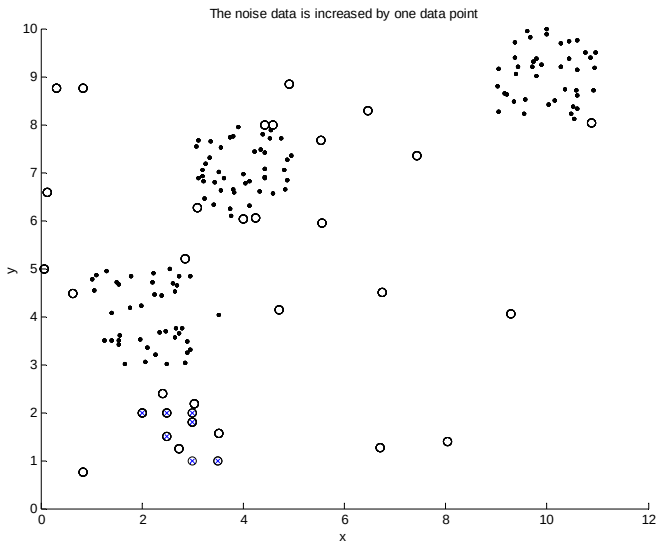


b) Data assignment dynamics in the second algorithm pass

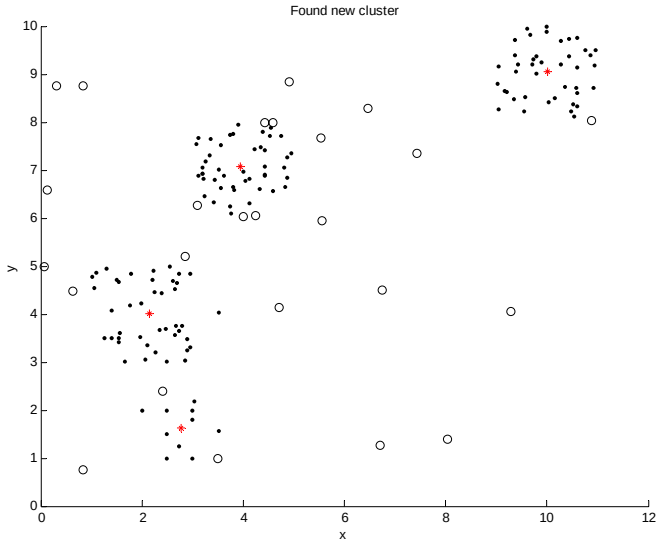


c) Data assignment dynamics in the third algorithm pass

Figure 2. The dynamics curves obtained for wine data clustering by the DDAA method



a) The already clustered data are given by dots, the noise data are presented by circles, whereas the new data stream is marked by circled ‘x’.



b) The previous clusters and the selected additional (forth) cluster of the input data stream surrounded by the left noise data. The stars are cluster centres.

Figure 3. Evolving DDAA algorithm applied to the artificial data set

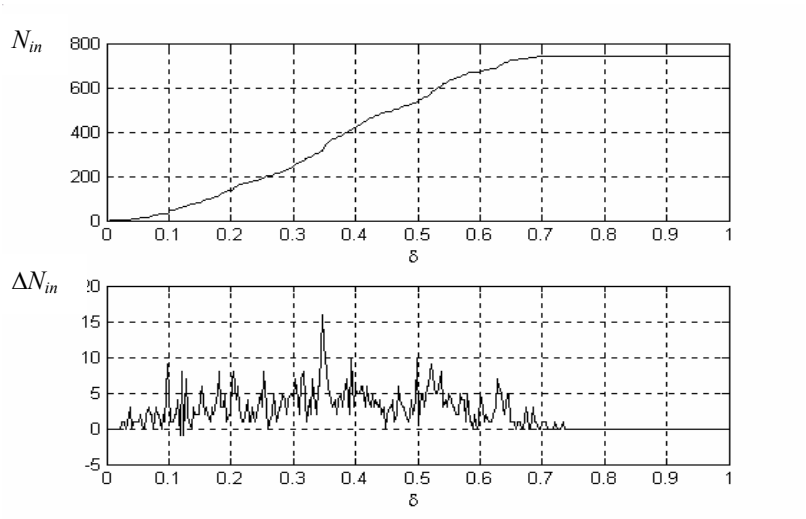


Figure 4. Data assignment dynamics in the first algorithm pass; $\Delta\delta = 0.003$ and $tol = 25$.

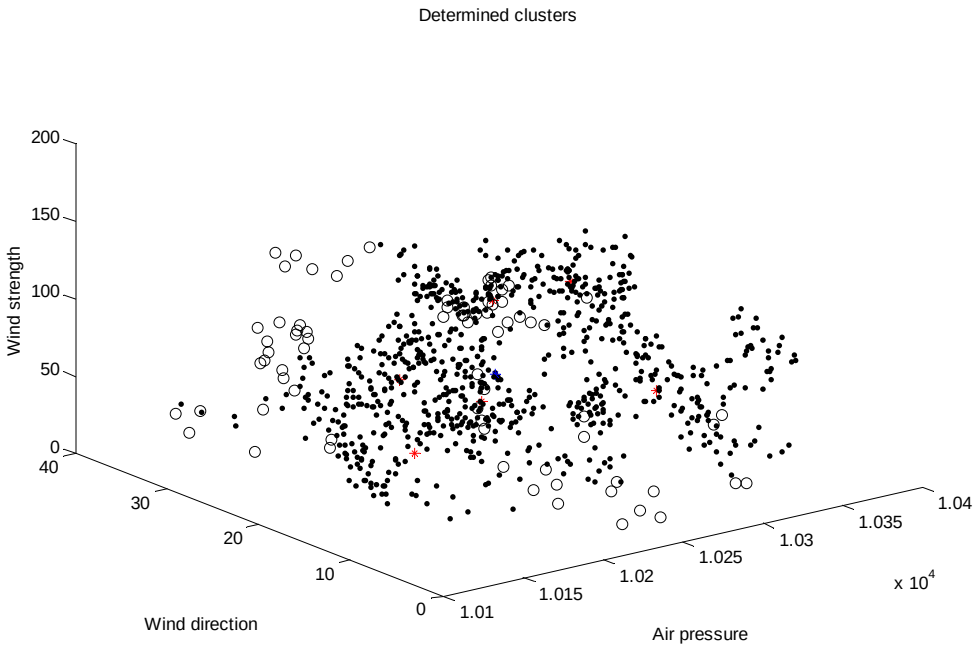


Figure 5. Clustering results: red * – cluster centres defined by DDAA algorithm, blue * – cluster centre of the new cluster determined by Evolving DDAA algorithm, o – noise data; clustered data are given by points.

Table 1. Partitioning of the wine data set

Cluster	Amount given in the data set	Amount obtained by DDAA	Amount of coinciding points
1	59	44	44
2	71	35	35
3	48	30	29

Table 2. Cluster centres of the weather data set partitioning

Cluster centre	$\mathbf{v_1}$	$\mathbf{v_2}$	$\mathbf{v_3}$	$\mathbf{v_4}$	$\mathbf{v_5}$	$\mathbf{v_6}$
Air pressure	10231	10193	10149	10297	10353	10211
Wind direction	19 (≈S)	21(≈S)	11(≈E)	11(≈E)	30(≈WN)	14(≈ES)
Wind strength	52	69	54	61	75	133

Table 3. Number of assigned data by the DDAA and evolving DDAA algorithm

Cluster	1		2		3		4		5		6		noise		7 new
	DDAA	EDDAA	DDAA	EDDAA	DDAA	EDDAA	DDAA	EDDAA	DDAA	EDDAA	DDAA	EDDAA	DDAA	EDDAA	EDDAA
Number of data	53	79	106	112	103	111	174	181	82	89	109	137	117	74	93