Statistics and Its Interface Volume 11 (2018) 183–189

Data stream clustering by fast density-peak-search*

Jinxia Su†, Yanwen Li, and Xuejing Zhao

Data stream mining has recently been studied extensively in the literature. Many clustering algorithms were proposed to handle massive streams of data. However, many of these algorithms may not be as efficient as one desires for data streams, as they typically require a number of iterations in their implementations. In this paper, we will propose a new data stream clustering algorithm, based on the fast density-peak-search method. It does not require any iterations in its implementation, and therefore is most suitable for large streams of data. The comparisons of numerical illustration as well as a real example will be made with other alternative data stream algorithms.

AMS 2000 subject classifications: Primary 62-07; secondary 68U20.
Keywords and phrases: Clustering, Data stream, Gaussian kernel density, Centrifugal distance, Density peaks.

1. INTRODUCTION

There exist many clustering algorithms in common use, e.g., k-means, k-medoids, hierarchical clustering, DBSCAN, density-based methods, just to name a few. They have been very effective in handling a variety of static data sets. These algorithms need to consider all data points and typically iterate over the data multiple times. On the other hand, data streams become more common in practical applications. Due to its dynamic nature, the speed of handling the data stream becomes an important issue, and algorithms for clustering static data sets are no longer adequate for data streams. Hence new algorithms are needed to tackle huge data streams.

There has been increasing literature on clustering data streams, see, e.g., [1, 2] and references therein. To explore massive stream data, the objects are usually summarized by k micro-clusters, see [3] and [4]. In some algorithms, micro-clusters are represented by non-empty grid cells (e.g., DStream by [5]). These clustering methods involve k-means or DBSCAN ([6]), etc. The algorithms are often modified to take into account the weighted micro-clusters. When the drift is involved, one can use the exponential fading strategy given in DenStream ([7]).

In most data stream clustering algorithms, the first step is to summarize the data stream with the help of particular data structures for dealing with space and memory constraints of stream applications. To summarize continuously arriving stream data and to give greater importance to up-to-date objects, a popular approach is to use a time window, e.g., landmark, sliding and damped windows. The offline component is used together with a variety of parameters to provide a quick understanding of the broad clusters in the data stream. Sequentially, traditional clustering algorithms (such as DBSCAN and k-means) can be used to find a data partition over the summaries. Note that the cluster shape will determine what clustering algorithm is employed. For instance, k-means generates hyper-spherical clusters, whereas DBSCAN allows the discovery of arbitrarily shaped clusters.

Rodriguez and Laio[8] proposed an effective algorithm to cluster the massive data with arbitrary shape. It works by fast finding the density peaks to decide the number of clusters, to decide the center of each cluster as well as to partition the data simultaneously. One attraction of this algorithm is that it does not need any iteration.

In this paper, we wish to develop a fast algorithm (*FStream*) in streaming data, by taking advantage of Rodriguez and Laio's algorithm ([8]) for static data. We achieve this by considering the problem of clustering a data stream in the sliding window model, in which only the most recent information from the data stream are stored in a data structure. The organization and manipulation of objects are based on the principles of queue processing. The comparison of the characters between the FStream and other three classical algorithms are illustrated in Table 1.

The rest of the paper is organized as follows. Section 2 describes the Rodriguez and Laio's algorithm (RL algorithm) and its applications in clustering of Data stream. The evaluation of data stream clustering algorithms is discussed in Section 3. The performance of the proposed algorithm is illustrated through the comparisons of numerical simulations and a real KDD Cup'99 data stream in Section 4 and Section 5, respectively.

*The project was sponsored by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, Ministry of Education of China, Supported by National Natural Science Foundation of China (No. 11571156) and by the Fundamental Research Funds for the Central Universities, Lanzhou University, P.R. China (No. lzujbky-2012-15,lzujbky-2013-178). The authors would also like to thank Edit-in-chief and the referees for their suggestions to improve the paper.
†Corresponding author.

Table 1. Analysis of four Data Stream Clustering Algorithms

Algorithm	Windows	Parameters	Algorithm	Shape
CluStream ([3])	landmark	4	k-means	hyper-sphere
DStream	damped	6	DBSCAN	arbitrary
DenStream	damped	7	DBSCAN	arbitrary
FStream	sliding	3	CFSFDP	arbitrary

2. METHODOLOGY

Our proposed *FStream algorithm* in clustering streaming data is based on Rodriguez and Laio's *fast density-peak-search algorithm* of [8], as described below.

2.1 Rodriguez and Laio (RL algorithm) to static data

Given a data set $X_1, ..., X_N$ in R^m, the dissimilarity between X_i and X_j is measured by Euclidean distance $d_{ij} = \|X_i - X_j\|_2$. From this, we can define two useful quantities.

- The Gaussian kernel density ρ_i of data point X_i:

$$\rho_i = \sum_j \exp(-(d_{ij}/d_c)^2), \tag{1}$$

  where $d_c = d_{(s)}$is the s-th smallest value of all distances d_{ij} $(i \neq j)$. The density ρ_i is useful in finding the centers of the clusters.
- The centrifugal distance δ_i from points of higher density:

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}). \tag{2}$$

  Note that δ_i is much larger than the nearest neighbor distance only for points that are local or global maxima in the density.

Cluster centers are those points for which both values of δ_i and ρ_i are relatively large.

Algorithm 1 describes the details of the RL algorithm. First calculate ρ and δ from (1) and (2), and select K cluster centers by choosing largest values of $W = \rho\delta$, and then assign other points to the cluster of nearest cluster center. The algorithm can be implemented by R package 'densityClust'. Figure 1 illustrates the accuracy of Algorithm 1 with an example.

2.2 FStream algorithm to streaming data

Some of the advantages of the RL algorithm include: it is very fast, and it is suitable for the data type of arbitrary shape. But it is only designed for static data.

Here, we wish to develop a fast algorithm (*FStream*) in streaming data, by taking advantage of RL algorithm ([8]) for static data. We achieve this by considering the problem of clustering a data stream in the sliding window model ([9]), in which only the most recent information from the data stream are stored in a data structure. The organization and manipulation of objects are based on the principles of queue processing. The comparison of the characters between the FStream and other three classical algorithms are illustrated in Table 1.

Algorithm 1 Fast Clustering Algorithm

Input:
D: data set $D \subset \mathbb{R}^N$;
K: number of clusters $K < N$;
Output:
c: clusters of the points;
centers: centers of clusters.
Algorithmic:

1: Calculate Euclidean distance matrix of D, find the cutoff distance d_c;
2: Calculate Gaussian kernel density ρ_i, centrifugal distance δ_i and $W_i = \delta_i \cdot \rho_i$ for $i = 1, 2, \ldots, n$;
3: Choose the largest K values of W: $W_{(1)} \geq W_{(2)} \geq \cdots \geq W_{(K)}$;
4: Find the corresponding K points from D as the centers of clusters;
5: **Initialize:** Label the centers form 1 to K;
6: **for each** X in $W_{K+1} \cdots W_N$ corresponding the points of D **do**
Assigning X to the cluster of nearest point X who has been labeled.
$c(X) \leftarrow$ The label of X
$c(X') \leftarrow$ The label of nearest point X' who has been labeled.
$c(X) = c(X')$
end for

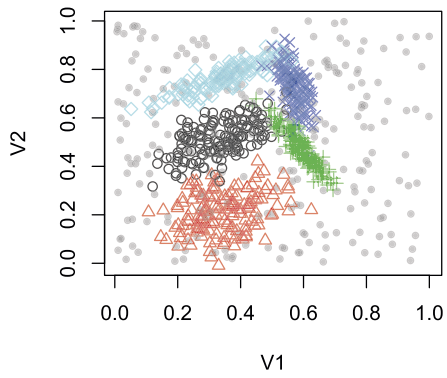
Algorithm 2 applies the fast density-peak-search clustering algorithm, i.e., RL algorithm, to the data stream-*FStream*. *FStream* uses the sliding windows model to realize the stream process. Algorithm 1 is utilized to find the clusters of the training set. Then, it starts to receive data streams, and assign the stream to the closest cluster or just a noise upon Algorithm 2. Finally, the certain amount of outliers (or the earliest points be recorded) are chosen and removed from the data set to realize the stream process.

In algorithm 2, a new input data X can be a data point as well as a collection of data points. When X is a set of data points, we have the unitary assignment and removement of the same number of points.

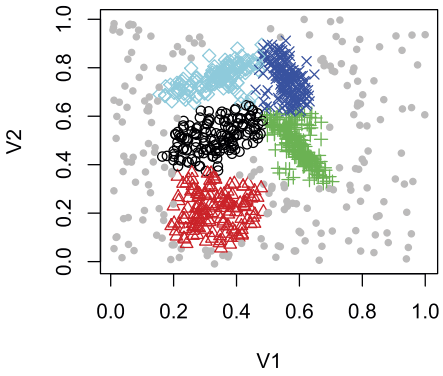
3. EVALUATION METHODS

In order to evaluate the performance of clustering on data stream mining, we need to set up some criterions. The evaluation of classical clustering for static data is extensively discussed in the literature and there are many evaluation criteria available. These evaluations only measure how well the algorithm learns static structure in the data.

For streaming data, the drift must be considered and it is important to evaluate how well the algorithm is able to



(a) Original



(b) RL

Figure 1. Original data with categories and the result of RL algorithm to the data.

adapt to these changes. The current state of the evaluation of data stream clustering can be found in [9] or [10]. In the following we discuss how to evaluate clustering algorithms in terms of learning static structures and clustering dynamic streams.

(1) The Purity ([11]) measures the ability of a clustering method for a known class (e.g. one knows the true class labels of each sample), it is applicable even when the resulted number of cluster is different from the original number. Purity is given by

$$\text{Purity} = \frac{1}{n}\sum_{q=1}^{k}\max_{1\le j\le l} n_q^j,$$

where n is the total number of samples and n_q^j is the number of samples in the cluster q that belongs to original class $j(1 \le j \le l)$. The larger the value of Purity, the better the clustering performance.

(2) The Adjusted Rand Index ([12]) is the corrected-for-chance version of the Rand Index. Given a set of n elements, and two groupings (e.g. clusterings) of these points, namely $X = \{X_1, X_2, \cdots, X_r\}$ and $Y = \{Y_1, Y_2, \cdots, Y_s\}$, the overlap between X and Y can be summarized in a contingency table $[n_{ij}]$ (See Table 2), where each entry n_{ij} denotes the number of objects in common between X_i and $Y_j : n_{ij} = |X_i \cap Y_j|$.
The Adjusted Rand Index is defined as

$$ARI = \frac{Index - ExpectedIndex}{MaxIndex - ExpectedIndex}, \tag{3}$$

more specifically,

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}, \tag{4}$$

Algorithm 2 FStream

Input:
D_{tr}: training data set;
K: number of clusters;
noise: proportion of noise
Output:
c: labels of the points;
centers: centers of clusters.
Algorithmic:

1: **while** $X \leftarrow$ We have more incoming stream **do**
2: Perform clustering on D_{tr}
3: Add the new stream point X into D_{tr}
$D_{tr} \leftarrow D_{tr} \cup \{X\}$
4: Calculate local density of each point in D_{tr}
$\rho_i = \sum_j \exp(-(d_{ij}/d_c)^2)$
5: sort the density $\rho_1 \le \rho_2 \le \cdots$, as a vector $\rho = (\rho_1, \rho_2, \cdots)$
6: Find out:
$\rho^* \leftarrow$ the value of position on the $length(\rho) * noise$
$\rho_X \leftarrow$local density of X
$X' \leftarrow$ the point nearest to X
7: **if** $\rho_X \ge \rho^*$ **then**
Assigning X to the cluster of X'
else
Assigning X to the outliers
end if
8: Find out the first point $X^* \in D_{tr}$ and its weight is assigned to be zero, i.e.,
$D_{tr} \leftarrow D_{tr} - \{X^*\}$
end while

Table 2. The contingency table

$X \setminus Y$	Y_1	Y_2	$\cdots$	Y_s	Sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
Sums	b_1	b_2	$\cdots$	b_s	

where n_{ij}, a_i, b_j are values from the contingency table.

(3) The ClassError (Classification error) is presented in the **mclust** ([13]) to calculate the error rate. The *errorRate* corresponds to a minimum error mapping between the resulted classification and the original classification. To coincide with the other criteria, hereafter we use CorrectRate = 1 − ClassError.

4. NUMERICAL STUDIES

We first generate five different simulated data streams. Each stream is produced as a mixture of data sets of $k-$cluters in $d-$dimentional space with noise, and each cluster is represented by a multivariate Gaussian distribution with a randomly chosen mean (cluster center) and covariance matrix. Noise points are generated from a $d-$dimensional uniform distribution in a bounding box. For more detailed see the **stream** package of R software. The five generated data sets are:

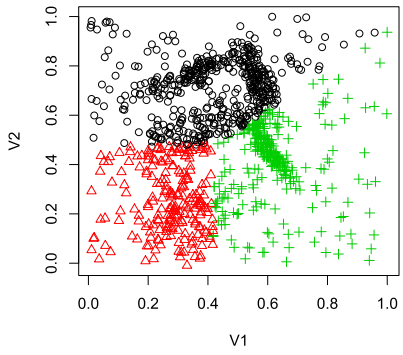
(a) five Gaussian clusters in two dimensions without noise (K5D2N00);
(b) five Gaussian clusters in two dimensions with 20% noise (K5D2N20);
(c) ten clusters in two dimensions data set without noise (K10D2N00);
(d) ten clusters in ten dimensions data set without noise (K10D10N00);
(e) ten clusters in ten dimensions data set with 20% noise (K10D10N20).

4.1 Comparisons of different algorithms

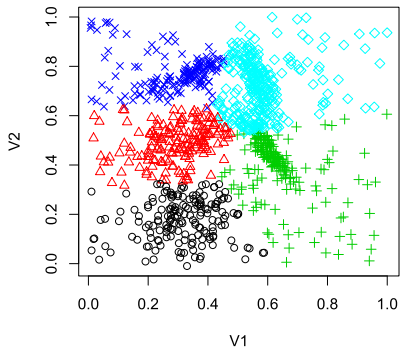
We now compare the four clustering methods. Each cluster method is applied into the five data sets above. Figure 2 gives the assignment areas for different data stream clustering algorithms, where the data is produced as the mixture of five Gaussian clusters in two dimensions with 20% noise. Each color represents a cluster.

Note that the real category is given in Figure 1(a). Clearly, our clustering algorithm (Figure 1(b)) has better clustering results. Since the clustering numbers in DStream algorithm is determined automatically, Figure 2(a) shows that the clustering number of DStream algorithm is 3 instead of 5. As shown in Figures 2(b) and 2(c), the CluStream algorithm and DenStream algorithm have poor clustering results.

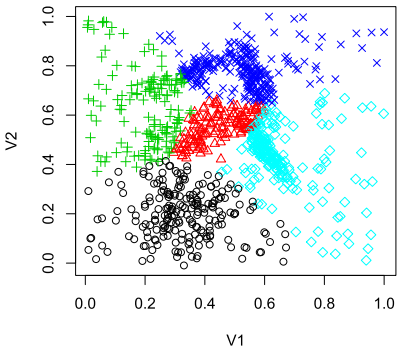
To compare numerical performances, 1000 objects are assigned and the comparison of the assignment results using the Purity, ARI and CorrectRate with respectively. From Table 3, even though the Purity is relatively high with DStream, CluStream and DenStream in all simulated cases, the Purity for FStream is still the highest among these algorithms. Therefore, the FStream improves the Purity (therefore the accuracy) significantly. Similar to the comparison of Purity, the results of CorrectRate (Table 4) and ARI (Table 5) also show the superiority of FStream algorithm.



(a) DStream



(b) CluStream



(c) DenStream

Figure 2. The assignment areas for different data stream clustering algorithms.

4.2 Numerical comparisons for streaming data

To evaluate the performance of the different clustering algorithms for streaming data, the data with two moving clusters crossing each other's path is exploited (Figure 3). First,

Table 3. Purity

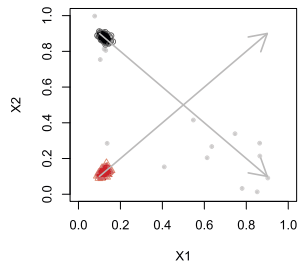
Data	*DStream*	*CluStream*	*DenStream*	*FStream*
k5d2n00	0.762	0.789	0.889	**0.911**
k5d2n20	0.831	0.860	0.754	**0.887**
k10d2n00	0.892	0.944	0.932	**0.992**
k10d10n00	0.919	0.957	0.849	**0.999**
K10d10n20	0.851	0.873	0.848	**0.996**

Table 4. CorrectRate

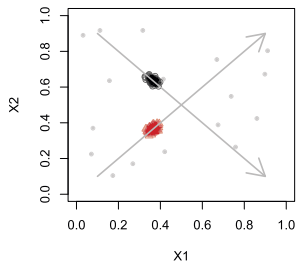
Data	*DStream*	*CluStream*	*DenStream*	*FStream*
k5d2n00	0.510	0.789	0.889	**0.911**
k5d2n20	0.404	0.668	0.554	**0.850**
k10d2n00	0.307	0.899	0.891	**0.992**
k10d10n00	0.208	0.901	0.451	**0.999**
K10d10n20	0.248	0.429	0.234	**0.991**

Table 5. ARI

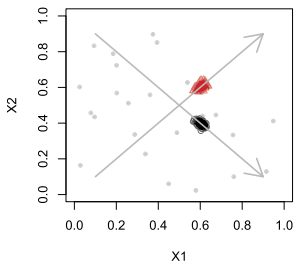
Data	*DStream*	*CluStream*	*DenStream*	*FStream*
k5d2n00	0.259	0.578	0.744	**0.790**
k5d2n20	0.323	0.675	0.539	**0.791**
k10d2n00	0.241	0.854	0.830	**0.982**
k10d10n00	0.114	0.870	0.272	**0.999**
K10d10n20	0.190	0.160	0.127	**0.991**



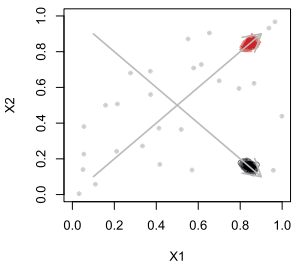
(a) Position 1



(b) Position 1650

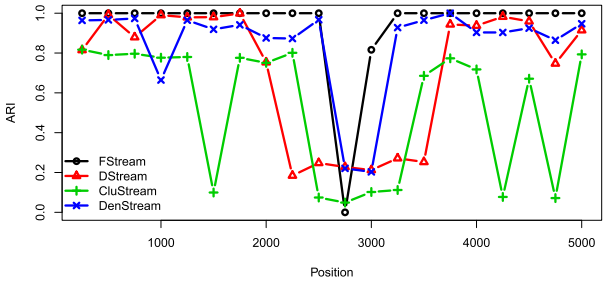


(c) Position 3300

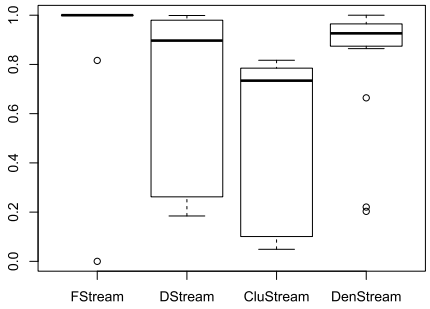


(d) Position 4950

Figure 3. Data points at different positions in the stream. The two arrows are added to highlight the direction of movement.



(a)

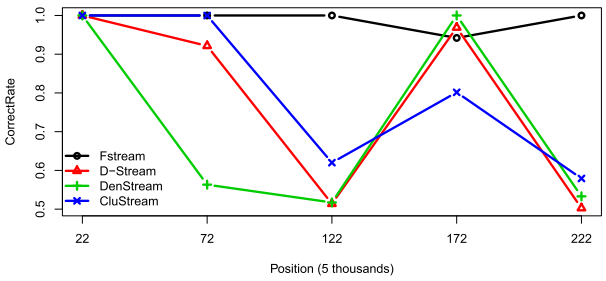


(b)

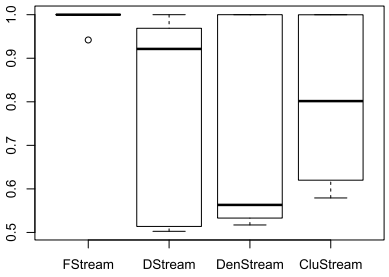
Figure 4. Evaluation of data stream clustering for simulated data stream.

a stream of 5000 data points in memory is created aligning the two directions as Figure 3, then these data points are clustered upon the different algorithms, evaluated using the ARI with a windows of 250 points, so totaly $5000/250 = 20$ times evaluations must be found for each algorithm. To plot the results we first get the positions at which the evaluation measure is calculated from the first element in the evaluation list and then extract a matrix with the ARI values. The matrix is visualized as a line plot and we add a boxplot comparing the distributions of the evaluation measure.

Figure 4(a) shows the ARI for the DStream, Clustream and Denstream as well as FStream over the dynamic data stream. Figure 4(b) is the corresponding box plot. All algorithms show that the ARI is low around position 3000, because of the inseparability of the two clusters in this position, where the two clusters are highly overlapped. From Figure 3 we can clearly see that the data structure have high separation except the epsilon neighborhood of position 3000. However, as shown in the Figure 4(a), the ARI of DStream and Denstream are extreme instability. Especially, the ARI of CluStream in position 1500, 4200 and 4700 is close to zero, the reason is that there are too many pre-defined parameters, which may not suitable for some data in the process of stream.



(a)



(b)

Figure 5. CorrectRate of the FStream comparing with DStream, DenStream and CluStream at different positions for KDD Cup'99 stream.

5. A REAL ILLUSTRATION

In this section, we use the widely used KDD Cup'99 dataset to evaluate the performance of proposed method, by comparing with DStream algorithm. The data is available from the UCI Machine Learning Repository. The data set contains 4,898,431 data points and the 34 numeric features are selected to give clustering results. We use the 1,500,000 data points to achieve our algorithm. The data set distinguishes between "bad" connections, called intrusions or attacks, and "good" normal connections. So we choose $k = 2$.

The precisions of the DStream, DenStream, CluStream and FStream algorithms are compared in Figure 5. Clearly, on average, FStream at various points is almost better than the others. The CorrectRate of FStream algorithm is close to 1. As the FStream algorithm needs fewer parameters than the other three algorithms, it is more stable. We can clearly see in Figure 5(a) that the CorrectRate of FStream algorithm is far higher than others in positions 122 and 222.

6. CONCLUSION

Based on the density-peak-search clustering (RL) algorithm in static data, this paper proposes a new data stream clustering algorithm, i.e., FStream. Its performance has been compared with those of three existing clustering methods (i.e., DStream, CluStream, DenStream) through numerical simulations and a real KDD Cup'99 data stream, under the three evaluation criterions: Purity, ARI and CorrectRate.

Under the simulated five different data sets, the Purity of FStream is much higher than the other three algorithms. Similar to the Purity, the results of CorrectRate and ARI also show the advantage of FStream algorithm. In the dynamic data stream, the proposed algorithm is compared with the DStream, the CluStream and the DenStream, the results show that our proposed algorithm has a higher accuracy of separation and gives a better performance. The result for real example of KDD Cup'99 data stream also shows that the average quality of FStream at various points are almost as good as that of DStream.

Received 31 October 2016

REFERENCES

[1] Jain, A., Murty, M. and Flynn, P. (1999). Data clustering: a review. *ACM Computer Surveys*, 31(3): 264–323.

[2] Silva, J., Faria, E., Barros, R., Hruschka, E., Carvalho, A. and Gama, J. (2013). Data stream clustering: a survey. *ACM Computer Surveys*, 46(1): 1–31.

[3] Aggarwal, C., Han, J., Wang, J. and Yu, P. S. (2003). A framework for clustering evolving data streams. *In: Proceedings of the International Conference on Very Large Data Bases (VLDB'03)*, 81–92.

[4] Zhang, T., Ramakrishnan, R. and Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2): 103–114.

[5] Chen, Y. and Tu, L. (2009). Stream data clustering based on grid density and attraction. *ACM Transactions on Knowledge Discovery from Data*, 3(3): 1–27.

[6] Ester, M., Kriegel, H., Sander, J. and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'1996)*, 226–231.

[7] Cao, F., Ester, M., Qian, W. and Zhou, A. (2006). Density-based clustering over an evolving data stream with noise. *In Proceedings of the 2006 SIAM International Conference on Data Mining.SIAM*, 328–339. MR2337946

[8] Rodriguez, A. and Laio, A. (2014). Clustering by fast search and find of density peaks. *Science*, 344(6191): 1492–1496.

[9] Aggarwal, C. (2007). *Data streams: models and algorithms, volume 31 of Advances in Database Systems.* Springer, New York, NY.

[10] Gama, J. (2010). *Knowledge discovery from data streams.* Chapman & Hall/CRC,Boca Raton, FL.

[11] Kim, H. and Park, H. (2007). Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data nalysis. *Bioinformatics*, 23(12): 1495–502.

[12] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2(1): 193–218.

[13] Fraley, C., Raftery A., Murphy, T. and Scrucca, L. (2012). Mclust version 4 for R: normal mixture modeling for model-based clustering, classification, and density estimation. *Technical Report No. 597, Department of Statistics*, University of Washington.

Jinxia Su
School of Mathematics and Statistics
Lanzhou University
Lanzhou, 730000
P.R. China
E-mail address: jinxiasu@lzu.edu.cn

Yanwen Li
College of Information Science and Engineering
Shanxi Agricultural University
Jinzhong
Shanxi, 030801
P.R. China
E-mail address: liyw13@lzu.edu.cn

Xuejing Zhao
School of Mathematics and Statistics
Lanzhou University
Lanzhou, 730000
P.R. China
E-mail address: zhaoxj@lzu.edu.cn