Continuous Outlier Detection on Uncertain Data Streams

Salman Ahmed Shaikh and Hiroyuki Kitagawa
Graduate School of Systems and Information Engineering, University of Tsukuba
Tennodai, Tsukuba, Ibaraki 305-8573, Japan
Email: salman@kde.cs.tsukuba.ac.jp and kitagawa@cs.tsukuba.ac.jp

***Abstract*—Time series data streams are common due to the increasing usage of wireless sensor networks. Such data are often accompanied with uncertainty due to the limitations of data collection equipment. Outlier detection on uncertain static data is a challenging research problem in data mining. Moreover, the continuous arrival of data makes it more challenging. Hence, in this paper, the problem of outlier detection on uncertain time series data streams is studied. In particular, we propose a continuous distance-based outlier detection approach on a set of uncertain objects' states that are originated synchronously from a group of data sources (e.g., sensors in WSN). A set of objects' states at a timestamp is called a state set. Generally, the duration between two consecutive timestamps is very short and the state of all the objects may not change much in this duration. Therefore, we propose an incremental approach of outlier detection, which makes use of the results obtained from the previous state set to efficiently detect outliers in the current state set. In addition, an approximate incremental outlier detection approach is proposed to further reduce the cost of incremental outlier detection. Finally, an extensive empirical study on synthetic and real datasets is presented, which shows the efficiency of the proposed approaches.**

I. Introduction

Outlier detection is a fundamental problem in data mining. It has applications in many domains including credit card fraud detection, network intrusion detection, environment monitoring, medical sciences, etc. Several definitions of outlier have been given in past, but there exists no universally agreed definition. Hawkins [1] defined an outlier as an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.

Recently, with the advancement in data collection technologies, e.g., wireless sensor networks (WSN), data arrive continuously and contain certain degree of inherent uncertainty [2]. The causes of uncertainty may include but are not limited to limitation of equipments, inconsistent supply voltage and delay or loss of data in transfer [2]. Detection of outliers from uncertain static data is a challenging research problem in data mining and on top of that, continuous arrival of data makes it more challenging. Hence in this paper, we propose an incremental approach of outlier detection on uncertain time series data streams.

Motivating example (identification of malfunctioning sensors): In WSNs (e.g., WSN for the monitoring of atmospheric pollution), sensors' observations are obtained continuously. These observations contain certain degree of inherent uncertainty. Assuming that the observations are being generated synchronously at a timestamp by all the sensors in the WSN, a set of observations is obtained at every timestamp. An observation is outlier if it deviates markedly from other observations in the set and the sensor generating outlying observations in the majority of timestamps (say $> 50\%$) is identified as malfunctioning.■

In this paper, we propose a continuous outlier detection approach for uncertain time series data streams. Namely, a distance-based approach is proposed to detect outliers continuously from a set of uncertain objects' states that are originated synchronously from a group of data sources (e.g., sensors in WSN). A set of objects' states at a timestamp is called a state set. Usually, the duration between two consecutive timestamps is very short and the state of all the objects may not change much in this duration. Therefore, to avoid the unnecessary computation at every timestamp, an incremental approach of outlier detection is proposed which makes use of outlier detection results obtained from previous timestamp to detect outliers in current timestamp (Sec. V). Moreover, an approximate continuous outlier detection approach is proposed to further reduce the cost of incremental outlier detection (Sec. VI). A cell-based approach, similar to our previous work [3], is utilized to reduce the cost of distance-based outlier detection from a state set (Sec. IV).

II. Related Work

Distance-based outlier detection approach on static data was introduced by Knorr et al. in [4]. They defined a data object o to be an outlier if at most threshold θ objects are within D-distance of o. [5] formulated distance-based outliers as the data objects whose distance to their κ^{th} nearest neighbour is largest. Angiulli et al. in [6] gave a slightly different definition of outliers than [5] by considering the average distance to their k nearest neighbors. Beside these, there are some works on the detection of distance-based outliers over data streams including [7], [8]. These works are based on the Knorr et al. [4] definition of outliers. Among these works, the incremental algorithm proposed in [8] is closest to our work. However, all these works deal with deterministic data and cannot handle uncertain data.

Recently, a lot of research has focused on managing, querying and mining of uncertain datasets [9], [10]. The

problem of outlier detection on uncertain datasets was first studied by Aggarwal et al. [9]. According to them, an uncertain object o is a density-based (δ, η) outlier, if the probability of existence of o in some subspace with density at least η is less than δ. However, their work was given for static data and cannot handle continuous data. In [10], Wang et al. proposed an outlier detection approach for probabilistic data streams. However, their work focuses on the tuple-level uncertainty. In contrast, in this paper, attribute level uncertainty is considered, i.e., the uncertainty lies in the measurements obtained from sensors and this uncertainty is given by the Gaussian PDF, with an assumption that sensor measurements may deviate from true values.

III. Background and Definitions

In this paper, the focus is distance-based outlier detection approach because the distance-based approach is the simplest and the most commonly used. Moreover, it can be used as preprocessing before applying more sophisticated application dependent outlier detection techniques. In [3], we defined distance-based outliers for uncertain static datasets as follows.

Definition 1: An uncertain object o_i in a database $\mathcal{G}DB$ is a distance-based outlier, if the expected number of objects lying within D-distance of o_i are less than or equal to threshold $\theta = N(1-p)$, where N is the number of uncertain objects in $\mathcal{G}DB$, and p is the fraction of $\mathcal{G}DB$ objects that lie farther than D-distance of o_i.

Def. 1 was given for static datasets. However, the focus of this work is uncertain time series data streams, where streams are the sequences of objects' states generated over time. This paper assumes that the states of all the objects are generated synchronously at every timestamp and the set of states at a timestamp is called a state set. It is further assumed that the objects' uncertainty follows the Gaussian distribution. The Gaussian distribution is chosen for representing uncertainty, because in statistics the Gaussian distribution is the most important and the most commonly used.

Formally in this paper, k-dimensional uncertain objects o_i are considered, with attributes vector $\overrightarrow{\mathcal{A}_i} = (x_{i1}, ..., x_{ik})$ following the Gaussian distribution with mean $\overrightarrow{\mu_i} = (\mu_{i1}, ..., \mu_{ik})$ and co-variance matrix $\Sigma_i = diag(\sigma_{i1}^2, ..., \sigma_{ik}^2)$. Namely, $\overrightarrow{\mathcal{A}_i}$ is a random variable that follows the Gaussian distribution $\overrightarrow{\mathcal{A}_i} \sim \mathcal{N}(\overrightarrow{\mu_i}, \Sigma_i)$. Assuming that there are N objects whose states may change over time, $S^j = \{\overrightarrow{\mathcal{A}_1^j}, ..., \overrightarrow{\mathcal{A}_N^j}\}$ denotes a state set of N objects at time t_j. Note that the $\overrightarrow{\mu_i^j}$ denotes the observed coordinates (attribute values) of an object o_i at time t_j. Hence, Def. 1 can be extended naturally for uncertain time-series data streams as follows.

Definition 2: An uncertain object o_i is a distance-based outlier at time t_j, if the expected number of objects in S^j lying within D-distance of o_i are less than or equal to threshold $\theta = N(1-p)$, where p is the fraction of objects that lie farther than D-distance of $o_i \in S^j$.

The objects that lie within the D-distance of o_i are called its *D-neighbors*, and the set of the D-neighbors of o_i and the number of D-neighbours are denoted by $DN(o_i)$ and $\#D$-neighbors(o_i), respectively. In order to find the distance-based outliers from a state set S^j at time t_j, distance between uncertain objects needs to be calculated. If attributes $\overrightarrow{\mathcal{A}_p}$ and $\overrightarrow{\mathcal{A}_q}$ of objects o_p and o_q, respectively follow the Gaussian distribution, then $|\overrightarrow{\mathcal{A}_p} - \overrightarrow{\mathcal{A}_q}| = \mathcal{N}(\overrightarrow{\mu_p} - \overrightarrow{\mu_q}, \Sigma_p + \Sigma_q)$ also follows the Gaussian distribution [11]. Let $Pr(o_p, o_q, D)$ denotes the probability that $o_q \in DN(o_p)$. Then,

$$Pr(o_p, o_q, D) = \int_R \mathcal{N}(\overrightarrow{\mu_p} - \overrightarrow{\mu_q}, \Sigma_p + \Sigma_q) \mathrm{d}\overrightarrow{\mathcal{A}} \ , \qquad (1)$$

where R is a sphere with centre $(\overrightarrow{\mu_p} - \overrightarrow{\mu_q})$ and radius D. For the expression and derivation of $Pr(o_p, o_q, D)$, please refer our previous work [3]. Furthermore, $Pr(\alpha, D)$ is used to denote $Pr(o_p, o_q, D)$ when there is no confusion, where α is an ordinary Euclidean distance between the means of o_p and o_q. Computing this probability is usually very costly, and it gets more expensive with the increase in data dimensionality.

In the following part, the discussion focuses on 2-dimensional case. However, the solution can be extended to higher dimensional cases without loss of generality. In addition, this work assumes that an object's attributes are uncorrelated and its standard deviations are uniform in all dimensions, to keep the discussion simple. The proposed continuous outlier detection approaches make use of results obtained from previous state set (S^{j-1}) to detect outliers in current state set (S^j). However, to reduce the cost of distance-based outlier detection within a state set, a cell-based approach similar to our previous work [3] is utilized. Hence in Sec. IV, a quick overview of the cell-based outlier detection approach [3] is presented. The proposed continuous outlier detection and the approximate continuous outlier detection approaches are presented in Secs. V and VI, respectively.

IV. Cell-based Outlier Detection

The Naive approach of the distance-based outlier detection within a state set S^j, is the use of nested loop to find the $\#D$-neighbors of each $o_i \in S^j$. This approach is very expensive and on average it requires $O(N^2)$ expensive probability $(Pr(o_i, o_j, D))$ evaluations. To reduce this costly computation, a cell-based approach [3] is used in this paper.

The cell-based approach is aimed at reducing the number of costly probability evaluations. It maps a dataset objects to a cell-grid and identify the cells containing only inliers or outliers based on the bounds on the $\#D$-neighbors. This approach will be utilized in Secs. V and VI for the proposed continuous outlier detection approaches.

A. Grid ($\mathcal{G}$) Structure

Lemma 1: Let o_p and o_q be two k-dimensional uncertain objects following the Gaussian distribution. Let α denotes an ordinary Euclidean distance between the means of o_p and o_q. Then for $w \in \mathcal{R}$, denoting the number of standard deviations required to enclose a large probability (say $> 99\%$) of a k-dimensional Gaussian difference distribution, following statements hold.

1) If $\alpha \leq D - w\sigma'$, $Pr(o_p, o_q, D) \approx 1$.
2) If $\alpha \geq D + w\sigma'$, $Pr(o_p, o_q, D) \approx 0$.

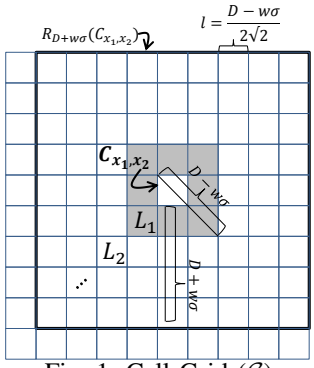


Fig. 1: Cell Grid ($\mathcal{G}$)

where σ' is the standard deviation of the Gaussian difference distribution in any one dimension (assuming that the standard deviation is uniform in all the dimensions).

Proof. Please refer to our previous work [12].

To identify distance-based outliers using the cell-based approach, the objects in the state set S^j are mapped to a 2-dimensional space that is partitioned into cells of length $l = \frac{D-w\sigma}{2\sqrt{2}}$ as shown in Fig. 1. Let C_{x_1,x_2} be a cell at the intersection of row x_1 and column x_2. The Layer 1 (L_1) neighbors of C_{x_1,x_2} are the immediate neighbouring cells of C_{x_1,x_2}, defined as follows and satisfies property 1.

$$L_1(C_{x_1,x_2}) = \{C_{u_1,u_2} | u_1 = x_1 \pm 1, u_2 = x_2 \pm 1, C_{u_1,u_2} \neq C_{x_1,x_2}\}.$$

Property 1: If $C_{u_1,u_2} \in L_1(C_{x_1,x_2})$, then $o_p \in C_{x_1,x_2}$ and $o_q \in C_{u_1,u_2}$ are at most $D - w\sigma$ apart.

From Property 1 and Lemma 1, $Pr(o_p, o_q, D) \approx 1$. The Layer 2 ($L_2$) neighbors of C_{x_1,x_2} are the immediate neighbouring cells of $L_1(C_{x_1,x_2})$ and are defined as follows.

$$L_2(C_{x_1,x_2}) = \{C_{u_1,u_2} | u_1 = x_1 \pm 2, u_2 = x_2 \pm 2, C_{u_1,u_2} \notin L_1(C_{x_1,x_2}), C_{u_1,u_2} \neq C_{x_1,x_2}\}.$$

$L_3(C_{x_1,x_2})$ and higher layers are defined in a similar way. Let $n_{D+w\sigma} = \left\lceil \frac{D+w\sigma}{l} \right\rceil$, then the region $R_{D+w\sigma}$ of C_{x_1,x_2} is defined as follows and satisfies property 2.

$$R_{D+w\sigma}(C_{x_1,x_2}) = \{C_{u_1,u_2} | u_1 = x_1 \pm n_{D+w\sigma}, u_2 = x_2 \pm n_{D+w\sigma}, C_{u_1,u_2} \notin L_1(C_{x_1,x_2}), C_{u_1,u_2} \neq C_{x_1,x_2}\}.$$

Property 2: If C_{u_1,u_2} is neither an L_1 nor an $R_{D+w\sigma}$ neighbour of C_{x_1,x_2} and $C_{u_1,u_2} \neq C_{x_1,x_2}$, then $o_p \in C_{x_1,x_2}$ and $o_q \in C_{u_1,u_2}$ are at least $D + w\sigma$ apart.

From Property 2 and Lemma 1, $Pr(o_p, o_q, D) \approx 0$.

B. Cell Bounds

Cell bounds on $\#D$-neighbors are computed to prune a cell as inlier or outlier, without expensive object-wise distance computation. The upper bound of C_{x_1,x_2}, $UB(C_{x_1,x_2})$, binds the maximum $\#D$-neighbors in the $\mathcal{G}$ for any object in C_{x_1,x_2}, and is defined as follows.

$$UB(C_{x_1,x_2}) = N(C_{x_1,x_2}) + \sum_{m=1}^{n_{D+w\sigma}} N(L_m(C_{x_1,x_2})) \times Pr((m-1)l, D) + (N - N(C_{x_1,x_2}) - \sum_{m=1}^{n_{D+w\sigma}} N(L_m(C_{x_1,x_2}))) \times Pr(n_{D+w\sigma} l, D),$$

where $N(.)$ denotes the number of objects. On the other hand, the lower bound of C_{x_1,x_2}, $LB(C_{x_1,x_2})$, binds the minimum $\#D$-neighbors in the $\mathcal{G}$ for any object in C_{x_1,x_2} and is defined as follows.

$$LB(C_{x_1,x_2}) = 1 + (N(C_{x_1,x_2}) - 1) \times Pr(\sqrt{2}l, D) + \sum_{m=1}^{n_{D+w\sigma}} N(L_m(C_{x_1,x_2})) \times Pr((m+1)\sqrt{2}l, D).$$

Since $Pr(\alpha, D)$ value is dependent on α and is independent from the location of C_{x_1,x_2}, the $Pr(\alpha, D)$ values required for the bounds computation are pre-computed to reduce the cost of the bounds computation. The $Pr(\alpha, D)$ values need to be computed only for $\alpha = m\sqrt{2}l$ $(1 \leq m \leq n_{D+w\sigma} + 1)$ and $\alpha = ml$ $(0 \leq m \leq n_{D+w\sigma} - 1)$. The pre-computed values are stored in a lookup table to be used for the bounds computation.

C. Cell Pruning

Let $\theta' = \left\lceil \frac{\theta}{Pr(D-w\sigma, D)} \right\rceil$, where θ is threshold and is dependent on parameter p, then the grid cells can be pruned using the following property.

Property 3:

1) If $N(C_{x_1,x_2}) > \theta'$, all the objects in C_{x_1,x_2} and $L_1(C_{x_1,x_2})$ are inliers.
2) If $N(C_{x_1,x_2}) + N(L_1(C_{x_1,x_2})) > \theta'$, all the objects in C_{x_1,x_2} are inliers.
3) If $LB(C_{x_1,x_2}) > \theta$, all the objects in C_{x_1,x_2} are inliers.
4) If $UB(C_{x_1,x_2}) \leq \theta$, all the objects in C_{x_1,x_2} are outliers.

For all the un-pruned objects in the un-pruned cells, the Naive approach is used to find their $\#D$-neighbors, to determine whether the un-pruned objects are inliers or outliers.

V. Continuous Outlier Detection

This section presents the proposed continuous outlier detection approach for time series data steams. Streams are the sequence of objects' states generated over time. We assume that the states of all the objects are generated synchronously and the set of states at a timestamp t_j is called a state set S^j, as shown in Fig. 2. The straightforward approach to detect outliers from each state set is to use the cell-based approach discussed in Sec. IV for every timestamp. However, the duration between two consecutive timestamps is usually very short and the state of all the objects may not change much in this duration. Hence, we propose an incremental approach of outlier detection, which makes use of outlier detection results obtained from state set S^{j-1} at t_{j-1} to detect outliers in state set S^j at t_j. This eliminates the need to process all the objects' states at every timestamp and saves a lot of computation time.

Time	State set	o_1	o_2	...	o_N
t_1	S^1	$\overrightarrow{\mathcal{A}_1^1}$	$\overrightarrow{\mathcal{A}_2^1}$	...	$\overrightarrow{\mathcal{A}_N^1}$
t_2	S^2	$\overrightarrow{\mathcal{A}_1^2}$	$\overrightarrow{\mathcal{A}_2^2}$	...	$\overrightarrow{\mathcal{A}_N^2}$
⋮	⋮	⋮	⋮	⋮	⋮
t_{j-1}	S^{j-1}	$\overrightarrow{\mathcal{A}_1^{j-1}}$	$\overrightarrow{\mathcal{A}_2^{j-1}}$	...	$\overrightarrow{\mathcal{A}_N^{j-1}}$
t_j	S^j	$\overrightarrow{\mathcal{A}_1^j}$	$\overrightarrow{\mathcal{A}_2^j}$	...	$\overrightarrow{\mathcal{A}_N^j}$

Fig. 2: State sets

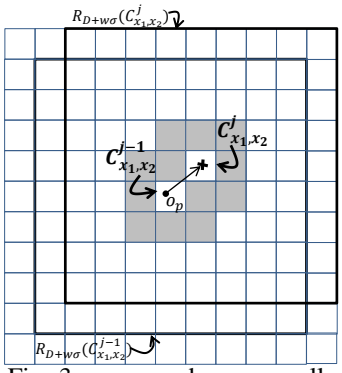


Fig. 3: o_p moved among cells

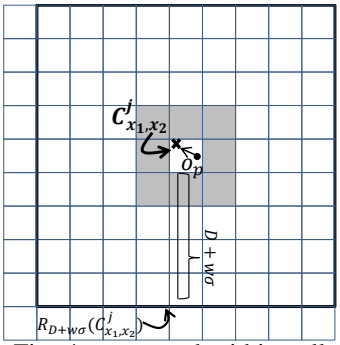


Fig. 4: o_p moved within cell

A. Incremental Outlier Processing

Let SC^j at timestamp t_j denotes a set of objects whose states change between timestamps t_{j-1} and t_j. We call such objects SC-objects (state-change objects). Note that $SC^j \subseteq S^j$. The main idea of the incremental outlier processing is to process only the objects which are either SC-objects or are affected by the SC-objects. We will utilize the cell-based algorithm discussed in Sec. IV to process only the SC-objects. The proposed incremental approach targets all state sets except the initial state set (S^1). For the S^1, no results are available from the previous state set, hence all the objects in the S^1 need to be processed using the cell-based approach. To simplify the problem, consider the case with one SC-object, o_p. Let $C^j_{x_1,x_2}$ represents a cell C_{x_1,x_2} at time t_j. As a result of state change, $o_p \in \mathcal{G}$ can move in one of the following two ways.

[Case 1] o_p moved to a different cell:

$$o_p \in C^{j-1}_{x_1,x_2}, o_p \in C^j_{x_1,x_2}, C^{j-1}_{x_1,x_2} \neq C^j_{x_1,x_2}.$$

[Case 2] o_p moved within a cell:

$$o_p \in C^{j-1}_{x_1,x_2}, o_p \in C^j_{x_1,x_2}, C^{j-1}_{x_1,x_2} = C^j_{x_1,x_2}.$$

Recall that in the cell-based approach, for the computation of cell bounds on $\#D$-neighbors, the cells within region $R_{D+w\sigma}$ are considered and for the upper bound, the cells outside the region $R_{D+w\sigma}$ are also considered. Hence, when an object moves among cells (case 1), it affects the cell bounds of all the cells within region $R_{D+w\sigma}$ of the $C^{j-1}_{x_1,x_2}$ and $C^j_{x_1,x_2}$ and the $\#D$-neighbors of all the un-pruned objects in the $\mathcal{G}$. Namely in case 1, o_p affects cells $C^{j-1}_{x_1,x_2}$, $C^j_{x_1,x_2}$, their L_1 and $R_{D+w\sigma}$ neighbors and all the objects in un-pruned cells in the $\mathcal{G}$. This movement does not affect the cell-based pruned cells outside $R_{D+w\sigma}$ region, because the number of objects outside the region $R_{D+w\sigma}$ is not affected by this movement. Fig. 3 shows the movement of o_p between $C^{j-1}_{x_1,x_2}$ and $C^j_{x_1,x_2}$ and their L_1 and $R_{D+w\sigma}$ neighbors.

On the other hand, when an object moves within a cell (case 2), it does not affect the bounds of any grid cell, however, this movement affects the $\#D$-neighbors of all the un-pruned objects in the $\mathcal{G}$. Fig. 4 shows the movement of o_p between $C^{j-1}_{x_1,x_2}$ and $C^j_{x_1,x_2}$ where $C^{j-1}_{x_1,x_2} = C^j_{x_1,x_2}$. We call the cells affected by the SC-objects *target cells*. Target cells require re-outlier detection with the arrival of new state set.

Algorithm 1 Incremental Outlier Detection

Input: S^j, $\mathcal{G}$, θ
Output: Set of distance-based outliers $\mathcal{O}$
/*Identifying state change objects*/
1: **for each** $o_i \in S^j$ **do**
2: **if** o_i is case-1 SC-object **then**
3: Add o_i to appropriate cell C^j, increase $N(C^j)$ by 1;
4: Delete o_i from cell C^{j-1}, decrease $N(C^{j-1})$ by 1;
5: Label C^j and C^{j-1} **A** and their L_1 and $R_{D+w\sigma}$ neighbouring cells **B**;
6: **end if**
7: **end for**
8: Label each $C^j \in \mathcal{G}$ **C**, if C^j is non-empty, un-pruned and not labelled **A** or **B**.
/*Processing cells of types A, B and C*/
9: **for each** $C \in \mathcal{G}$ **do**
10: **if** C is labelled **A**, **B** or **C**, process them using cell-based approach of Sec. IV and obtain set of outliers $\mathcal{O}$;
11: **end for**
12: **for each** o_i in un-pruned cells **do**
13: Compute $\#D$-neighbors(o_i) using the $Pr(o_i, o, D)$ values available in Hash table. ($Pr(o_i, o, D)$ computation is required if either o_i or o or both are SC-objects or $Pr(o_i, o, D)$ is not available in the Hash table)
14: **if** $\#D$-neighbors(o_i) $\leq \theta$ **then** o_i is outlier. Add o_i to $\mathcal{O}$;
15: **end for**
16: **return** $\mathcal{O}$;

1) Target Cells: In practise, there are more than one SC-objects between t_{j-1} to t_j. Therefore, we expand the idea to more than one SC-objects. Hence, the target cells can be classified into following 3 types.
Type A: Cells containing SC-objects which have moved to or from another cell at time t_j ($C^{j-1}_{x_1,x_2}$ and $C^j_{x_1,x_2}$ in Fig. 3).
Type B: L_1 and $R_{D+w\sigma}$ neighbouring cells of Type A cells, except those classified as Type A.
Type C: Un-pruned cells of the grid $\mathcal{G}$. Type C cells may include Type A and B cells.

All three cell types, i.e., A, B and C require re-outlier detection with the arrival of new state set, while rest of the cells do not need to be processed. Lines 1-11 of Algorithm 1 is used for the processing of target cells.

2) Incremental Processing of Un-pruned Objects: Due to the expensive $\#D$-neighbors computation, the main cost of our proposed algorithm lies in the processing of un-pruned objects (Type C cells). The $\#D$-neighbors computation of an object o_p requires $Pr(o_p, o_q, D)$ computation between o_p and each $o_q \in S^j$. In the incremental algorithm, this cost can be reduced by utilizing the $Pr(o_p, o_q, D)$ values computed in the processing of previous state set. Namely, a Hash table is used to store $Pr(o_p, o_q, D)$ values computed at time t_{j-1}. At time t_j, these values are retrieved from the Hash table in $O(1)$ time. Hence at time t_j, $Pr(o_p, o_q, D)$ values need to be computed only in two cases; 1) States of o_p, o_q or both have changed, 2) $Pr(o_p, o_q, D)$ is not available in the Hash table. Since un-pruned objects form a fraction of the state set, the memory required to hold the Hash table is not significant. However it saves a lot of computation time. Lines 12-15 of Algorithm 1 shows the processing of un-pruned objects.

B. Discussion: determination of values for D, p and l and outlier detection from high dimensional data

Let us begin by stating that there is no universally correct value for parameters D, p or l. Parameter D has an affect on the $\#D$-neighbors of an object and $\#D$-neighbors are computed using $Pr(o_i, o_j, D)$ function. Larger D value results in large $Pr(o_i, o_j, D)$ values and therefore large $\#D$-neighbors and vice versa. However very small or very large D value is not recommended as it results in very small or very large $\#D$-neighbors, respectively for all the state set objects and hides the difference between strong and weak outliers. Hence an appropriate D value may be decided by considering the dataset distribution by the end user. The parameter p is used in the determination of threshold, $\theta = N(1-p)$ and it affects the number of outliers returned by the proposed algorithms. Since an outlier occurs rarely, therefore it is reasonable to select a value of p very close to unity. For example, for $N = 10^3$, $p = 0.995$ may be appropriate, but for $N = 10^6$, may be far too small. For the latter case, $p = 0.99995$ may be more appropriate.

Although in this paper value of l is fixed, however it can be varied and has an affect on the performance of the proposed approaches rather than the accuracy. Smaller l values are good for cell pruning as they result in tighter bounds, however very small l increases the number of cells in the grid exponentially and the time required for the bounds computation. On the other hand, larger l values result in looser bounds and hence reduce the cell pruning capability. As the number of dimensions k increases, the number of grid cells increases exponentially and therefore larger l values are recommended for higher k.

For very high k, cell-based approach is not suitable due to the high memory and the processing time requirements to hold and process the large number of cells, respectively. Moreover, $Pr(o_i, o_j, D)$ computation becomes very expensive. In addition, for very high k, objects are very sparse and each object behaves like outlier. Therefore, distance-based outlier detection is not a feasible solution to detect outliers in full space. Hence, subspace mining and/or dimensionality reduction techniques must be employed to detect outliers from such data.

VI. Continuous Outlier Detection using the Bounded Gaussian Uncertainty

Approximating the Gaussian uncertainty by the bounded Gaussian uncertainty enables an approximate but more efficient outlier detection. According to this paper's assumption, attributes of uncertain objects follow the Gaussian distribution. Therefore according to the 3-sigma rule, there is a 95.45% chance that uncertain objects' attribute values lie within 2 standard deviations of the observed values and 99.73% chance that the values lie within 3 standard deviations of the observed values [13]. Hence the conventional Gaussian distribution can be normalized within certain boundaries to increase the efficiency of outlier detection at a small loss of accuracy.

Given a two dimensional conventional Gaussian function $g_{\overrightarrow{\mathcal{A}}}(x_1, x_2)$ with mean $\overrightarrow{\mu} = (\mu_1, \mu_2)$ and co-variance matrix $\Sigma = diag(\sigma^2, \sigma^2)$, the bounded Gaussian distribution $f_{\overrightarrow{\mathcal{A}}}(x_1, x_2)$ can be defined following the practise of [14], as follows.

$$f_{\overrightarrow{\mathcal{A}}}(x_1, x_2) = \begin{cases} \frac{g_{\overrightarrow{\mathcal{A}}}(x_1, x_2)}{\int_{(x_1, x_2) \in o.ur} g_{\overrightarrow{\mathcal{A}}}(x_1, x_2) \mathrm{d}x_1 \mathrm{d}x_2} & (x_1, x_2) \in o.ur \\ 0 & otherwise \end{cases} \tag{2}$$

where $o.ur$ denotes the uncertainty region of the bounded Gaussian distribution. This paper assumes that the uncertainty region is a sphere with centre (μ_1, μ_2) and radius r.

By bounding the Gaussian uncertainty, a cell can be pruned by simply counting the number of objects in its neighbouring layers. Moreover the major cost of outlier detection, that is, the processing of un-pruned objects also reduces significantly. This is because, with the bounded Gaussian, the outlier detection algorithm needs to consider limited number of objects for the computation of an object's $\#D$-neighbors rather than all the objects, as in the case of the conventional Gaussian uncertainty. Interested readers may refer to our previous work [3], for the details of the bounded Gaussian uncertainty.

A. Bounded Gaussian Cell-based Outlier Detection

Like the grid structure of the conventional Gaussian uncertainty in Sec. IV-A, objects in the state set S^j are mapped to a 2-dimensional space that is partitioned into cells of length $l = \frac{D-2r}{2\sqrt{2}}$. The cell layers are defined in a similar manner as that of Sec. IV-A and cell length l is chosen in such a way to satisfy the property 4.

Property 4: If $C_{u_1,u_2} \in L_1(C_{x_1,x_2})$, then $o_p \in C_{x_1,x_2}$ and $o_q \in C_{u_1,u_2}$ are at most $D - 2r$ distance apart.

From property 4, an $o_p \in C_{x_1,x_2}$ and an $o_q \in C_{u_1,u_2}$ are guaranteed to be D-neighbors mutually, with the $Pr(o_p, o_q, D) = 1$. Cells in region $R_{D+2r}(C_{x_1,x_2})$ are those which fall within $D + 2r$ distance of C_{x_1,x_2}. Let $n_{D+2r} = \left\lceil \frac{D+2r}{l} \right\rceil$, then the region R_{D+2r} of C_{x_1,x_2} is defined as follows and satisfies property 5.

$$R_{D+2r}(C_{x_1,x_2}) = \{C_{u_1,u_2} | u_1 = x_1 \pm n_{D+2r}, u_2 = x_2 \pm n_{D+2r}, C_{u_1,u_2} \notin L_1(C_{x_1,x_2}), C_{u_1,u_2} \neq C_{x_1,x_2}\}.$$

Property 5: If C_{u_1,u_2} is neither an L_1 nor an R_{D+2r} neighbour of C_{x_1,x_2} and $C_{u_1,u_2} \neq C_{x_1,x_2}$, then $o_p \in C_{x_1,x_2}$ and $o_q \in C_{u_1,u_2}$ are at least $D + 2r$ apart.

From property 5, it can be guaranteed that an $o_p \in C_{x_1,x_2}$ and an $o_q \in C_{u_1,u_2}$ are greater than $D + 2r$ distance apart, hence $Pr(o_p, o_q, D) = 0$. Using the properties 4 and 5, grid cells can be pruned as follows.

Property 6:

1) If $N(C_{x_1,x_2}) > \theta$, all the objects in C_{x_1,x_2} and $L_1(C_{x_1,x_2})$ are inliers.
2) If $N(C_{x_1,x_2}) + N(L_1(C_{x_1,x_2})) > \theta$, all the objects in C_{x_1,x_2} are inliers.
3) If $N(C_{x_1,x_2}) + N(L_1(C_{x_1,x_2})) + N(R_{D+2r}(C_{x_1,x_2})) \leq \theta$, all the objects in C_{x_1,x_2} are outliers.

For the un-pruned objects in the un-pruned cells from the cell-based pruning, $\#D$-neighbors are computed using only the objects within $D + 2r$ distance of the target object, to find whether the un-pruned objects are inliers or outliers.

B. Bounded Gaussian Incremental Outlier Detection

As a result of state change, an object can either move within the cell or among the cells. Hence the two cases are similar to the one discussed in Sec. V-A for the conventional Gaussian uncertainty. Since the uncertainty of an object is bounded, as a result of change in their state, fewer number of neighbouring cells are affected as compared to the conventional Gaussian uncertainty, reducing the number of target cells requiring re-outlier detection. Similarly, *Type C* cells now do not contain all the un-pruned cells of the grid. However, they contain only the un-pruned cells within region R_{D+2r} of the cell containing an SC-object. As a result, the computational cost of the incremental outlier detection algorithm using the bounded Gaussian uncertainty reduces significantly than the conventional Gaussian uncertainty.

VII. Experiments

We conducted extensive experiments on synthetic and real datasets to evaluate the effectiveness of the proposed approaches. All algorithms are implemented in C++, GNU compiler. All experiments are performed on a system with an Intel Core 2 Duo CPU E8400 3.00GHz CPU and 2GB main memory running Ubuntu 12.04 OS. All programs run in main memory and no I/O cost is considered. In figures, the incremental algorithms using the conventional Gaussian and the bounded Gaussian uncertainties are denoted by DM(CG) and DM(BG), respectively. These two algorithms are compared with the simple cell-based method (CM). In the CM, the cell-based algorithm of Sec. IV is executed for all the objects in the state set at every timestamp.

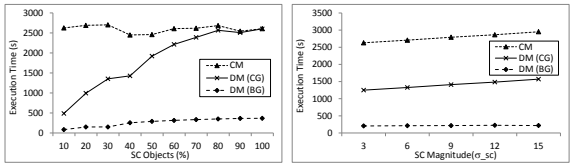
A. Datasets

In this paper a synthetic and a real datasets are used for experiments. Synthetic dataset, tri-modal Gaussian (TG) is 2-dimensional and is generated using Box Muller method [15]. This method generates pair of independent, standard, normally distributed (zero mean, unit variance) random numbers, given a source of uniformly distributed random numbers. In order to provide the stream behaviour in synthetic dataset, states of the fraction of dataset objects are modified by adding normal random numbers with zero mean and standard deviation σ_{SC} at every timestamp. The proposed incremental algorithm is executed on this modified dataset. Unless specified, the synthetic dataset consists of 5,000 tuples. As for real-world data, we have used three hourly met office weather (MOW) forecast data available at [16]. In the experiments on real data, we have used a two dimensional subset of MOW data, which consists of screen and feels like temperature forecast values for 5,802 weather stations around UK. Consecutive forecasts are used as data stream in these experiments.

Both the datasets are normalized to have a domain of [0,1000] on every dimension. For each point z in the datasets, an uncertain object o is created, whose uncertainty is given by the Gaussian distribution with mean z and standard deviation σ in both the dimensions. Unless specified, the following parameter values are used in the experiments. $D = 100$, $\sigma = 5$, $r = 3\sigma$, $p = 0.99$ for the TG dataset and $p = 0.985$ for MOW dataset, SC-objects ratio = 30% for the TG dataset (SC-objects ratio for the MOW dataset depends on the number of objects changing states between two forecasts) and $w = 3$. Pre-computation time is not included in the measurements.

B. Results

Experiments are conducted to evaluate the efficiency of the proposed algorithms. Since there are no known algorithms for distance-based outlier detection on uncertain data streams, the simple cell-based method (CM) is used as baseline.

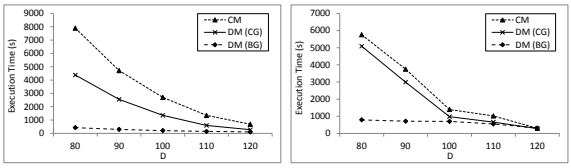


(a) Varying SC Object's % (b) Varying SC Magnitude

Fig. 5: TG Dataset

Firstly, experiments are performed on the synthetic dataset TG by varying the percentage of SC-objects. Fig. 5a shows the effect of varying the SC-objects' percentage on the execution time. In the figure, as the percentage of the SC-objects increases, the number of target cells requiring re-evaluation also increases. As a result the execution times of our proposed algorithms DM(CG) and DM(BG) also increase. The graphs of DM(CG) and CM meets when all the objects in the dataset change their states. At this point, the DM(CG) algorithm becomes similar to that of CM algorithm, that is, executing cell-based algorithm for the complete state set rather than only for SC-objects. As discussed in previous sections, the main cost of our algorithms lies in the processing of un-pruned objects. In the bounded Gaussian case, for the computation of $\#D$-neighbors of an un-pruned object, limited number of objects within certain boundary are considered. However, the conventional Gaussian algorithm needs to consider all the objects in the grid for the computation of $\#D$-neighbors of an un-pruned object. Therefore the algorithm DM(BG) is far

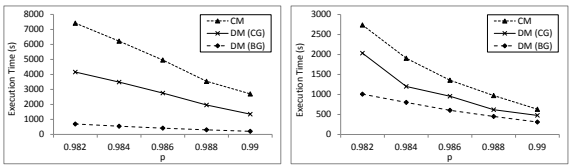
less expensive than the DM(CG). The variation in CM graph is due to the change in percentage of moving objects, which results in the variation in the state set distribution.



(a) TG Dataset (b) MOW Dataset

Fig. 6: Varying D

Next, experiments are performed by varying the magnitude of movement of SC-objects. The magnitude was varied by varying the σ_{SC}. From Fig. 5b, we can observe that as σ_{SC} increases, the execution times of all the algorithms increase. This is due to the fact that the increase in σ_{SC}, results in an increase in the number of objects that move among cells. Hence the number of target cells increase, which is the cause of increase in the execution times. Next experiments are performed by varying the parameter D. As D increases, $\#D$-neighbors also increases. As a result, objects are easily pruned and we obtain fewer outliers. This results in decline in the execution times as can be observed from Fig. 6. With the increase in D, the number of cells requiring re-processing (target cells) also increases. Due to which the number of cells need to be re-processed by the DM(CG) and DM(BG) approaches become nearly equal. As a result the difference in the execution times is not so big when D reaches 120.



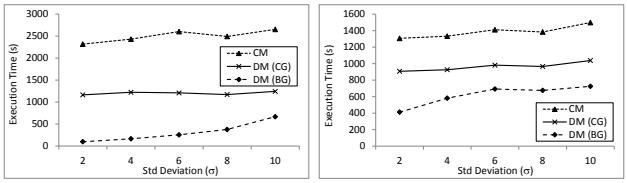
(a) TG Dataset (b) MOW Dataset

Fig. 7: Varying p

We also performed experiments by varying the parameter p. As p increases, threshold θ decreases, which results in a decline in execution times of all the algorithms for both the datasets. However, the proposed incremental approaches are still faster than the CM. This can be observed from Figs. 7a and 7b. The effectiveness of the proposed approaches is also measured with the increasing level of state set objects' uncertainty. Figs. 8a and 8b show the effect of increasing the uncertainty level. As standard deviation (σ) increases, the uncertainty level of objects increases. As a result objects are not pruned easily, which causes an increase in the execution times of the algorithms.

VIII. Conclusion and Future Work

In this work, two continuous distance-based outlier detection approaches (an exact and an approximate) are proposed for uncertain time series data streams. The proposed approaches are based on the incremental processing of the state change objects, that is, they process only those objects which are



(a) TG Dataset (b) MOW Dataset

Fig. 8: Varying Standard Deviation σ

affected by the change in objects' states. We employed a cell-based algorithm for the efficient detection of outliers within a state set, in both the incremental algorithms. An extensive empirical study on synthetic and real datasets demonstrates the efficiency of the proposed approaches. In the future, we plan to extend this work for high-dimensional data and general uncertainty model.

Acknowledgment

This work is partly supported by Grant-in-Aid for Scientific Research(A)(#24240015A).

References

[1] D. Hawkins, "Identification of outliers," ser. Monographs on Applied Probability and Statistics, 1980, pp. 1–12.

[2] A. B. Sharma, L. Golubchik, and R. Govindan, "Sensor faults: Detection methods and prevalence in real-world datasets," *ACM Trans. Sen. Netw.*, vol. 6, no. 3, pp. 23:1–23:39, 2010.

[3] S. A. Shaikh and H. Kitagawa, "Efficient distance-based outlier detection on uncertain datasets of gaussian distribution," *World Wide Web*, pp. 1–28, 2013.

[4] E. M. Knorr, R. T. Ng, and V. Tucakov, "Distance-based outliers: Algorithms and applications," *VLDB J.*, vol. 8, no. 3-4, pp. 237–253, 2000.

[5] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," *SIGMOD Rec.*, vol. 29, no. 2, 2000.

[6] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *Principles of Data Mining and Knowledge Discovery*, 2002, pp. 15–27.

[7] M. Kontaki, A. Gounaris, A. Papadopoulos, K. Tsichlas, and Y. Manolopoulos, "Continuous monitoring of distance-based outliers over data streams," in *ICDE*, 2011.

[8] K. Ishida and H. Kitagawa, "Detecting current outliers: Cont. outlier detect. over time-series data streams," in *DEXA*, 2008.

[9] C. C. Aggarwal and P. S. Yu, "Outlier Detection with Uncertain Data," in *SDM*, 2008.

[10] B. Wang, G. Xiao, H. Yu, and X. Yang, "Distance-based outlier detection on uncertain data," in *CIT*, 2009.

[11] E. W. Weisstein, "Normal difference distribution." http://mathworld.wolfram.com/, 2013, [Online; accessed 03-September-2013].

[12] S. A. Shaikh and H. Kitagawa, "Fast top-k distance-based outlier detection on uncertain data," in *WAIM*, 2013.

[13] F. Pukelsheim, "The three sigma rule," *The American Statistician*, vol. 48, no. 2, pp. 88–91, 1994.

[14] Y. Tao, X. Xiao, and R. Cheng, "Range search on multidimensional uncertain data," *ACM Trans. Database Syst.*, vol. 32, no. 3, 2007.

[15] W. J. Thistleton, J. A. Marsh, K. Nelson, and C. Tsallis, "General. box-muller method for generating q-gauss. random deviates," *IEEE Trans. Inf. Theor.*, vol. 53, no. 12, pp. 4805–4810, 2007.

[16] "Met office weather data," http://data.gov.uk/data, 2013, [Online; accessed 03-September-2013].