A Framework for Estimating Complex Probability Density Structures in Data Streams

Arnold P. Boedihardjo
Department of Computer Science
Virginia Tech, USA
aboediha@vt.edu

Chang-Tien Lu
Department of Computer Science
Virginia Tech, USA
ctlu@vt.edu

Feng Chen
Department of Computer Science
Virginia Tech, USA
chenf@vt.edu

ABSTRACT
Probability density function estimation is a fundamental component in several stream mining tasks such as outlier detection and classification. The nonparametric adaptive kernel density estimate (AKDE) provides a robust and asymptotically consistent estimate for an arbitrary distribution. However, its extensive computational requirements make it difficult to apply this technique to the stream environment. This paper tackles the issue of developing efficient and asymptotically consistent AKDE over data streams while heeding the stringent constraints imposed by the stream environment. We propose the concept of local regions to effectively synopsize local density features, design a suite of algorithms to maintain the AKDE under a time-based sliding window, and analyze the estimates' asymptotic consistency and computational costs. In addition, extensive experiments were conducted with real-world and synthetic data sets to demonstrate the effectiveness and efficiency of our approach.

Categories and Subject Descriptors
G.3 [**Probability and Statistics**]: Nonparametric statistics

General Terms
Algorithms, Performance

Keywords
Data Mining, Data Streams, Kernel Density Estimation

1. INTRODUCTION
Advances in hardware and software technologies have caused a surge in the growth and availability of voluminous information. Data streams are realizations of vast information that are fast, continuous, mutable, ordered, and unbounded [5]. Numerous data streams stem from the ubiquitous time series and span a wide range of applications such as finance, medicine, and sensor networks. Some well-known streams are traded stock prices, measures of brain electrical impulses (e.g., electroencephalograms), and roadway performance metrics (e.g., speed). Applying analysis and mining techniques can help deepen knowledge in these fields and enhance their applications. Therefore, the development of stream analysis tools can provide far-reaching impacts to the general discipline of stream mining.

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.

Underpinning many stream mining tasks is the use of the *probability density function (PDF)* for the most recent data [3, 5, 11, 22]. However, in real-world situations, the PDFs are usually unknown and therefore must be estimated. Examples of stream mining tasks that employ estimated PDFs include: outlier detection by modeling a sensor's sample distribution and selecting data points of low probability [21]; concept drift detection via comparison of current and past data streams' probability density estimates [2]; and pattern discovery in Internet traffic by visualizing the estimated probability density function of arriving packets [22]. One could further the efficacy of probability density estimates by enabling queries for an explicit time range [3]. The extension would be able to respond to questions such as, "What is the distribution of telnet connections within the last hour?" "How have the roadway's speed and volume distributional patterns changed in the past two hours since the snow began?" The extension of an explicit time range is naturally modeled by a time-based window. Hence, it can be seen that providing probability density estimates over a time-based sliding window will serve as a valuable tool in the study of data streams. Emphasis on the most recent data implies the need for the stream mining techniques to furnish results in real-time. Additionally, practical constraints dictate that the techniques possess finite memory and perform at most a linear pass on the data elements [5, 16]. Meeting and balancing these requirements is a key goal in stream research.

An effective technique to estimate an unknown probability density function is the nonparametric kernel density estimate (KDE). KDE possesses several advantages that include: rigorous mathematical foundation; generalization to other density estimators such as orthogonal series and histograms; asymptotic consistency; and inheritance of the kernel function's continuity and differentiability properties [19, 20]. The standard formulation of the univariate KDE is given as follows: for n independent and identically distributed sample points (i.e., kernel centers) $x_1..x_n$ with corresponding weights $w_1..w_n$, bandwidth h, and a kernel function $K(\cdot)$, the kernel density estimate is

$$p_{KDE}(x) = \frac{1}{\sum_1^n w_i}\sum_{i=1}^{n}\frac{w_i}{h}K\left(\frac{x-x_i}{h}\right)$$

The accuracy of the KDE does not significantly depend on the choice of kernel function $K(\cdot)$, but rather on the selection of h [19]. The bandwidth h is regarded as a smoothing parameter: a high h can generate a smooth shape density whereas a low h tends to provide an undersmoothed but potentially more accurate estimate. The drawback of the KDE formulation is the requirement of assigning a global bandwidth. Due to the existence of local features, a single global bandwidth may not be sufficient to model complex density structures (e.g., multimodal distributions). When the global bandwidth KDE technique is used as the core step in mining tasks (e.g., outlier detection), the

generated results can be misleading and potentially disastrous for mission critical applications (e.g., military sensor surveillance). To overcome this problem, this paper proposes the use of an adaptive KDE (AKDE) to improve the estimation accuracy of local features within data streams. The AKDE is a variable bandwidth technique, which has been shown to be effective in capturing local density features [18-20]. The general formulation of the AKDE is as follows:

$$p_{AKDE}(x) = \frac{1}{\sum_1^n w_i}\sum_{i=1}^{n}\frac{w_i}{H(x_i)}K\left(\frac{x-x_i}{H(x_i)}\right), \quad H(x_i) \propto f(x_i)^{-1} \text{ (eq. 1)}$$

where $H(x_i)$ is the bandwidth function that is inversely proportional to the true density $f(x_i)$.

In essence, the AKDE increases its learning capacity (via smaller bandwidth) in regions of high density where the local features are likely to originate from the true distribution. This AKDE characteristic is consistent with the real-world observation: inherent local features tend to occur in regions of high probabilities whereas noise-induced local features occur in areas of low probabilities. This adaptive scheme enables the AKDE to provide superior estimation accuracy over the classical KDE.

Although the AKDE can produce superior estimation quality to the classical KDE, its computational cost ($O(n^2)$) far exceeds the traditional KDE ($O(n)$). In the AKDE, the bandwidth, $H(\cdot)$, is computed from the true density, $f(\cdot)$. Because the true density is unknown, a pilot function is defined to provide an estimate for $f(\cdot)$. Generally, the pilot function is modeled by the classical KDE. This choice implies that evaluating $H(\cdot)$ is an $O(n)$ process. Therefore, computing a query under the AKDE is an $O(n^2)$ operation because $H(\cdot)$ is computed at least once for each sample point. This evaluation approach cannot be applied to data streams since it clearly infringes upon the linear pass restriction. Due to AKDE's extensive computational cost, there are currently no works that provide adaptive kernel density estimates for the data stream environment.

A crucial property provided by the KDE is asymptotic consistency, that is, as the sample size approaches infinity, the KDE converges to the true density. This KDE property is conducive to the data stream environment because the unbounded sample size provided by the data stream can improve the KDE's accuracy. Therefore, to enable AKDE to be a viable tool in stream analysis, a new online technique must be developed that is both efficient and asymptotically consistent.

This paper tackles the issue of developing efficient and asymptotically consistent AKDE over data streams. To the best of our knowledge, this paper is the first attempt that addresses the issue of applying AKDE to the data stream setting. To that end, we propose, the online Local Region KDE (LR-KDE), an adaptive kernel density estimation framework for processing univariate data streams. The major components of the proposed framework are (1) a new partition-based variable bandwidth strategy to capture local density features and to improve estimation quality, (2) a suite of *linear pass* algorithms to maintain and compute kernel density estimates, and (3) a fixed-size memory time-based sliding window. We also analyze the asymptotic costs and consistency property of the proposed online LR-KDE. Extensive experiments were conducted to demonstrate the effectiveness and efficiency of the approach.

The proceeding sections are organized as follows. Section 2 surveys the related works. Section 3 provides the theoretical preliminary. Section 4 details our proposed framework. Section 5 presents the computational complexity and asymptotic consistency analysis. Section 6 demonstrates empirical results and validation. Section 7 gives the conclusion.

2. RELATED WORKS

Early efforts in computationally tractable KDEs can be found in the domain of *offline* analysis. Zhang *et al.* proposed a method to maintain a space efficient KDE by using the CF-tree [23] to cluster a set of kernels into a single kernel known as the CF-Kernel [24]. The CF-Kernel employs a global bandwidth which can lead to oversmoothing and loss of local density information. Gray *et al.* proposed a kernel space partitioning technique by utilizing a KD-Tree and bounded support kernels to offline data sets [9]. The KD-Tree reduces computations by effectively pruning kernels, which do not contribute to the density query.

Several recent works have been proposed for the online management of KDE. These online techniques can be classified into the following three categories:

1. **Grid-based KDE** – provides a uniform and discretized representation of the kernel points
2. **Sample-based KDE** – employs a sampling methodology on the data stream to reduce the total kernel size
3. **Cluster-based KDE** – utilizes kernel merging techniques to maintain a fixed storage and minimize the kernel merge error

Grid-based and sample-based: Aggarwal proposed a framework to capture the structural evolution of data streams by using a grid-based KDE [2]. The kernels are summarized in a multidimensional grid where a common bandwidth is employed for each dimension. Concepts of forward and reverse density profiles are introduced to discover the occurrences of concept drifts. Subramaniam *et al*. proposed an approach for outlier detection in sensor networks by modeling the densities of node observations [21]. Their scheme employs a uniformly sampled sequence-based sliding window to summarize the current set of kernels. A global bandwidth is applied to the sampled set of kernels. Wegman *et al.* introduced an online KDE for the analysis of Internet traffic [22]. They suggested the use of a sequence-based and exponentially aging sliding window to accommodate a fixed storage environment. To derive estimates, a single bandwidth KDE is employed on the sliding window.

Cluster-based: Zhou *et al.* introduced the M-Kernel, a cluster-based KDE maintenance strategy which performs numerically-based kernel mergers under a fading window [25]. The merging algorithm combines two kernels which produces the minimum integrated absolute error between the original pair and the merged result. This scheme allows for a fixed memory representation of the kernels. However, under the M-Kernel, the consistency of the estimate is not guaranteed and the approach can exhibit high update costs due to its numerical-based merging strategy. In a similar vein, Heinz *et al.* proposed a constant time pair-wise merging technique that guarantees consistency [12]. Their kernel method employs a single bandwidth that has been shown to be effective in approximating the classical KDE.

Table 1. Summary of stream-based KDE techniques. M is the number of maintained kernels.

Technique		Query Cost	Update Cost	Bandwidth Strategy	Asymp. Consis.	Time-based Window
Grid-based	[2]	$O(M)$	$O(M)$	Single Bandwidth	Yes	Yes
Sample-based	[21], [22]	$O(M)$	$O(1)$	Single Bandwidth	Yes	Yes
Cluster-based	Heinz [12]	$O(M)$	$O(M)$	Single Bandwidth	Yes	No
Cluster-based	M-Kernel [25]	$O(M)$	$O(M)$	Variable Bandwidth	No	No
Cluster-based	LR-KDE	$O(M)$	$O(M)$	Variable Bandwidth	Yes	Yes

Table 1 provides a summary of the key characteristics of current stream-based KDEs. Most KDE approaches employ single a bandwidth strategy hence they cannot accurately estimate the stream's local features. The M-Kernel, although it applies a variable bandwidth approach, it is not assured to be asymptotically consistent. As a consequence, the M-Kernel is not guaranteed to converge as the sample size increases. The proposed LR-KDE differentiates itself from existing works in the following aspects:

1. **Local feature estimation** – models local density features via partition-based bandwidth to improve estimation quality
2. **Consistency** – assures asymptotic consistency under the proposed variable bandwidth strategy
3. **Linear pass processing** – employs $O(M)$ algorithms to process kernel updates and density queries
4. **Time-based window** – provides a fixed-size time-based sliding window

Online histograms have also been proposed in the field of database optimization to provide query selectivity estimates and approximate queries [13]. Some online histograms include dynamic quantiles [8], equidepth histograms [7], and V-optimal histograms [10]. Due to the histogram's inherent discontinuities and slower convergence, the histogram may not be suited for the tasks of stream analysis [20].

3. THEORETICAL PRELIMINARY

A formal description of the density estimation problem is given as follows: *Given a data stream,* $S = \{x^{(1)}, x^{(i)}, .., x^{(n)}\}$*, where* $x^{(i)}$ *is a real-valued scalar (i.e., sample point) and each* $x^{(i)}$ *is associated with a timestamp,* $\tau(x^{(i)}) \in [\tau_{min}, \tau_{max}]$*, generate and maintain an adaptive kernel density estimator,* $\hat{f}_{AKDE}(\cdot)$*, of* S*. The storage, update, and query costs of* $\hat{f}_{AKDE}(\cdot)$ *should be at most* $O(M)$*, where* M *is a constant and* $M \ll \|S\| = n$*.*

With respect to stream applications, one of the fundamental issues of the AKDE is its high computational cost for determining the bandwidth, $H(\cdot)$ (eq. 1). To reduce this computational requirement, a bandwidth approximation technique is proposed for the AKDE. Let $T = \{x_i : x_i \leq x_{i+1}, 1 \leq i \leq n, x_i \in S\}$ be an ordered representation of the kernels in S. Define the *relative density variance*, $R(k,l)$, as the sample variance of the set of estimated densities at $x_k..x_l \in T$, where $1 \leq k \leq l \leq n$. The bandwidth approximation procedure is given as follows:

1. Partition T into Q continuous and disjoint *local regions* (i.e., intervals) such that each local region's $R(\cdot,\cdot)$ value is minimized
2. For each local region, assign a bandwidth that is unique to its constituent kernels

The above scheme serves to capture the local densities within the total span of the distribution. The obtained approximation is consistent with the structure of AKDE's bandwidth i.e., similar bandwidth values are assigned to kernels of similar densities. Hence, the local regions can be seen as a piecewise constant representation of the structure of $H(\cdot)$.

Two design challenges exist in applying the above approximation approach: (1) the efficient derivation of the relative density variance; and (2) the development of a technique for local region identification. In the following, the Pair-wise Adjacent Distance Uniformity theorem is introduced to provide an efficient venue for estimating the density variance. The theorem is followed by the definition of an optimization problem for the task of identifying the local regions.

3.1. Derivation of Relative Density Variance

As previously defined, local regions provide a total and disjoint partitioning of the kernel domain which minimizes the intra-variance of the density estimates. A unique bandwidth is assigned to each local region based on the regional kernel characteristics. If each kernel is given its own unique local region, then the local region based KDE (with the appropriate bandwidth function) is a reformulation of the traditional AKDE. However, if the number of the local regions is less than the number of kernels, then the local region KDE is an approximation of the AKDE. The problem now is to derive a method which can efficiently deduce the density estimate variance for a given range of kernels. One possible approach is to estimate the densities via the traditional KDE approach. However, this poses the same computational issue as the AKDE. An alternative and more viable solution is to employ the pair-wise and adjacent kernel distances to derive relevant properties of the density variance. To that endeavor, the Pair-wise Adjacent Distance Uniformity theorem is proposed.

Let G be a set of ordered and identically weighted kernels whose pair-wise and adjacent distance variance is zero, then under certain conditions, it can be shown that the densities at the kernel centers (under a global bandwidth KDE) in G are uniform. The significance of this theorem is that it provides a venue of estimating the density variance through information of the kernels' pair-wise and adjacent distances (PAD). As a result, computationally tractable bandwidth approximations can be developed from the kernels' PAD information. The proof of the Pair-wise Adjacent Distance Uniformity theorem is as follows:

THEOREM 3.1 (Pair-wise Adjacent Distance Uniformity): *Let* $V = \{x_i : x_i \leq x_{i+1} \text{ and } 1 \leq i \leq n\}$ *be a set of sorted kernel centers associated to a bounded and radially-symmetric kernel function with uniform weight and bandwidth. Furthermore, let the sorted kernels be adjacently equidistant such that* $|x_{i+1} - x_i| = |x_{j+1} - x_j| \; \forall i,j$*. Suppose* $G = \{x_k : x_1 + bandwidth \leq x_k \leq x_n - bandwidth\}$*. Then for the kernel density estimate,* $\hat{f}(x)$*, the following property must hold:* $\hat{f}(x_k) = \hat{f}(x_l) \; \forall \, x_k, x_l \in G$*.*

PROOF. Let $x_k, x_l \in V$ be any two kernel centers and define a set $I_x = \{x_i : |x_i - x| < bandwidth\}$. I_x is the set of kernel centers

for which their corresponding bandwidths intersect x, thus I_x possess all the kernels that contribute non-zero values to the kernel density estimate, $\hat{f}(x)$. Consider I_{x_k} and I_{x_l} and without loss of generality choose a kernel center, $\alpha \leq x_k$, from I_{x_k}. Because all the kernel centers within G are adjacently equidistant, there must exist an element, $\beta \leq x_l$, from I_{x_l} such that $x_k - \alpha = x_l - \beta$. Since the kernels are radially-symmetric, equal bandwidth, and identically weighted, the contribution of α to $\hat{f}(x_k)$ is equal to the contribution of β to $\hat{f}(x_l)$. For any chosen α in I_{x_k}, there exists a corresponding β in I_{x_l} for which their contributions are equal. This relationship also holds for any β in I_{x_l} corresponding to an α in I_{x_k}. Hence, there is a one-to-one relationship between α in I_{x_k} and β in I_{x_l}, which implies that the sum of the kernel contributions of I_{x_k} to $\hat{f}(x_k)$ and I_{x_l} to $\hat{f}(x_l)$ are equal. Therefore, $\hat{f}(x_k) = \hat{f}(x_l)\ \forall\ x_k, x_l \in G$.

3.2. Optimization Problem for Local Region Identification

Leveraging upon Theorem 3.1, identification of the local regions can be achieved by minimizing the variance of the kernels' pair-wise and adjacent distance values. In the following, an optimization problem is established to identify local regions based on the kernels' PAD information. Assume that a local region, L, possesses a set of kernels (centers), $y_j..y_m$, where $y_j \leq y_{j+1} \leq y_m$ for $1 \leq j \leq m \leq n$, and n is the total number of kernels. Set L's i^{th} adjacent distance to be $L_d(i) = y_{j+i+1} - y_{j+i}$, and define the variance of L's kernel pair-wise and adjacent distances as follows:

$$var(L_d) = \frac{1}{\|L\|-1}\sum_{i=1}^{\|L\|-1}(L_d(i) - \frac{\sum_1^{\|L\|-1} L_d(i)}{\|L\|-1})^2 \qquad \text{(eq. 2)}$$

where $\|L\|$ is the number of kernels in L.

In practice, the density estimate makes use of arbitrarily weighted kernels. Hence, consideration of varying weights must be made in the local region formulation. Let $z_1..z_n$ be a set of sorted kernel centers assigned to a bounded, radially-symmetric, and equal bandwidth kernel function, such that $|z_{i+1} - z_i| = |z_{j+1} - z_j|\ \forall i,j$. Let $w(z_i)$ be the non-negative weight of kernel center z_i, and $\hat{f}(z_i)$ be the density estimate at z_i. From the KDE definition, $\hat{f}(z_i) \propto w(z_i) \Rightarrow var\left(\hat{f}(Z)\right) \propto var(w(Z))$. Therefore, minimizing $var\left(\hat{f}(Z)\right)$ is equivalent to minimizing $var(w(Z))$. Let $L_{kw}(i)$ be the weight of kernel y_{j+i} in L, and define L's kernel weight variance, $var(L_{kw})$, as follows:

$$var(L_{kw}) = \frac{1}{\|L\|}\sum_{i=1}^{\|L\|}(L_{kw}(i) - \frac{\sum_1^{\|L\|} L_{kw}(i)}{\|L\|})^2 \qquad \text{(eq. 3)}$$

Using the local region PAD (eq. 2) and the kernel weight variance (eq. 3) formulae, partition the entire sample points $y_1..y_n$ into Q disjoint local regions by minimizing the aggregate variance of all L_d and L_{kw}:

$$\text{Min } \sum_{i=1}^{Q}\|L\|\left(var\left(L_d^{(i)}\right) + \mu \cdot var\left(L_{kw}^{(i)}\right)\right) \qquad \text{(eq. 4)}$$

where μ is the weight assigned to L's kernel weight variance.

In summary, the solution to the above local region optimization problem generates local regions with minimum overall intra-density variation. Moreover, the problem is solved by exclusively employing the kernels' PAD and weight information which are amenable to efficient implementations.

4. PROPOSED APPROACH: ONLINE LOCAL REGION KDE

The local region identification problem can be solved via dynamic programming techniques in time $O(n^2Q)$ where Q is the number of local regions; however, this solution exceeds the constraints of the data stream problem. Therefore, an incremental local region identification technique is proposed. The technique employs a locally optimal decision strategy to reduce the identification task to $O(nQ)$. As a result, the online Local Region KDE (LR-KDE) is proposed for the efficient generation of density estimates in data streams.

4.1. Online LR-KDE Overview

The online LR-KDE is composed of the following two primary components:

1. *Local region management* – identifies and manages local regions by employing a locally optimal dynamic binning method for the entire set of kernels.
2. *Kernel maintenance* – updates the kernels with new data points in a fixed-size memory environment. Maintains and provides density evaluation of kernels whose time stamps are within $[\tau_{min}, \tau_{max}]$.

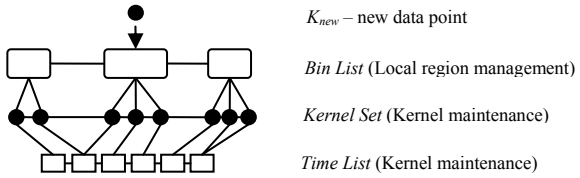

Figure 1. Online Local Region KDE

Figure 1 illustrates the LR-KDE approach. An online binning method is applied on the set of all kernels where each bin represents a local region. The bins are maintained in the data structure, *bin list*, which can store at most Q number of bins. The *kernel set* manages and organizes *all* kernels in a sorted queue ordered by their centers. A maximum of $M \geq Q$ kernels is maintained in the kernel set. The *time list* structure is a sorted queue of kernel arrival times. The objective of the *time list* is to model a time-based sliding window (using the FIFO policy) and to support efficient operations of kernel expirations and insertions. The following is a summary of the main operations for inserting a new data point K_{new}:

1. *Bin list update*: Find the bin whose interval intersects K_{new}; add a reference to K_{new} to the bin's set of kernels; and forward K_{new} to the *kernel set* list. If no intersecting bin exists, create a new bin for K_{new} and (possibly) merge two adjacent bins to maintain a maximum of Q bins.
2. *Kernel set insertion*: Insert K_{new} into position by searching the *kernel set*. If after the insert, the kernel set size is greater than M, then merge two kernels of the kernel set which minimizes the overall accuracy loss.
3. *Time list synchronization*: After K_{new} is added to the *kernel set*, its arrival time is appended to the tail of the *time list*. If a kernel merger occurs, then its corresponding timestamps will

also need to be updated in the *time list*. The head of the *time list* will also need to be verified against τ_{min} in order to remove expired kernels.

4.2. Local Region Management

The following proposes an incremental bin management approach for the identification of local regions. A formal definition of the bin and its property are given as follows:

Bin Vector –a bin vector, $\mathbb{B}$, is defined as follows:

$$\mathbb{B} = [b_{ks}, b_{ss}, b_{ws}, b_{wss}, b_{wcs}, b_{wcss}, b_{sp}]$$

b_{ks} is the set of kernels in $\mathbb{B}$ sorted by their kernel centers. The following feature definitions refer to set b_{ks}. b_{ss} denotes the squared sum of the kernel centers. b_{ws} gives the sum of the kernel weights. b_{wss} yields the squared sum of the kernel weights. b_{wcs} indicates the sum of the weighted kernel centers. b_{wcss} provides the weighted squared sum of the kernel centers. Let $x_1..x_{b_c}$ be the sorted kernel centers in b_{ks}, then the sum of adjacent kernel centers product is $b_{sp} = \sum_{i=1}^{b_c-1} x_i x_{i+1}$.

The bin vector possesses the additivity property [23]. This property allows the combining of bins in constant time. Let $\mathbb{B}^{(1)}$ and $\mathbb{B}^{(2)}$ be a pair of disjoint bins, $\max\left(b_{ks}^{(1)}\right)$ = the maximum kernel in $b_{ks}^{(1)}$, and $\min\left(b_{ks}^{(2)}\right)$ = minimum kernel in $b_{ks}^{(2)}$. If $\max\left(b_{ks}^{(1)}\right) < \min\left(b_{ks}^{(1)}\right)$, then the sum of bin vectors $\mathbb{B}^{(1)} + \mathbb{B}^{(2)}$ is:

$$\mathbb{B}^{(1)} + \mathbb{B}^{(2)} = \left[b_{ks}^{(1)} \cup b_{ks}^{(2)},\ b_{ss}^{(1)} + b_{ss}^{(2)},\ b_{ws}^{(1)} + b_{ws}^{(2)},\ b_{wss}^{(1)} + b_{wss}^{(2)},\ b_{wcs}^{(1)} + b_{wcs}^{(2)},\ b_{wcss}^{(1)} + b_{wcss}^{(2)},\ b_{sp}^{(1)} + b_{sp}^{(2)} + \max\left(b_{ks}^{(1)}\right) \cdot \min\left(b_{ks}^{(2)}\right)\right] \quad \text{(eq. 5)}$$

Bin Creation and Merger: Bins are updated as new kernels enter the system. For the *bin list*, there are two cases to address when K_{new} enters: (1) the number of bins in *bin list* is less than Q; and (2) the number of existing bins is equal to Q. For both cases, if K_{new}'s center intersects any one of the bins, then a reference to K_{new} is added to the bin's kernel set, b_{ks}, and forwarded to the global *kernel set*. Suppose that the number of existing bins is less than Q, and K_{new} does not intersect any bin. Since the number of bins is less than Q, a new bin is created with K_{new} as its first and center point. The bin management algorithm continues in this manner until Q bins have formed. Now assume that there are Q bins in the *bin list* and K_{new} does not intersect any bin. In this scenario, a new bin is created for K_{new} and two bins (including one formed by K_{new}) are merged to maintain Q number of bins.

The merger of two bins is defined by its additivity property (eq. 5). Because the bins represent local regions, they must remain continuous and mutually disjoint. Therefore, the merger can only occur between adjacent bins. The merger candidates are selected based on the objective function of the local region optimization (eq. 4). The following is an expanded expression of the uniformly weighted local region PAD variance (eq. 2) in terms of kernel centers $x_1, x_i, .., x_n$:

$$var(L_a) = \frac{2}{\|L\|-1}\left(\sum_{i=1}^{\|L\|} {x_i}^2 - \sum_{i=1}^{\|L\|-1} x_i x_{i+1} - \left(\frac{x_1^2 + x_{\|L\|}^2}{2}\right)\right) - \left(\frac{x_{\|L\|} - x_1}{\|L\|-1}\right)^2$$

By utilizing the above formula, the variance of the i^{th} bin's pair-wise adjacent kernel distance is determined to be as follows (for uniformly weighted kernels):

$$\mathbb{B}_{dv}^{(i)} = \frac{2}{\left\|b_{ks}^{(i)}\right\|-1}\left(b_{ss}^{(i)} - b_{sp}^{(i)} - \left(\frac{\min\left(b_{ks}^{(i)}\right)^2 + \max\left(b_{ks}^{(i)}\right)^2}{2}\right) - \frac{(\max(b_{ks}^{(i)}) - \min(b_{ks}^{(i)}))^2}{2\left\|b_{ks}^{(i)}\right\|-2}\right)$$

Similarly, the i^{th} bin's kernel weight variance is given as follows which is a reformulation of (eq. 3) in terms of the bin vector components:

$$\mathbb{B}_{wv}^{(i)} = \frac{b_{wss}^{(i)}}{\left\|b_{ks}^{(i)}\right\|} - \left(\frac{b_{ws}^{(i)}}{\left\|b_{ks}^{(i)}\right\|}\right)^2$$

Therefore, a merger will occur for the pair of adjacent bins which minimizes the following objective function:

$$\text{Min } J = \sum_{i=1}^{Q} \left\|b_{ks}^{(i)}\right\| (\mathbb{B}_{dv}^{(i)} + \mu \cdot \mathbb{B}_{wv}^{(i)})$$
$$\text{for } i = 1..Q \text{ and } \mu = 1$$

When K_{new} is added, updating all of the bin's features (with the exception of b_{sp}) follows straight-forward algebraic calculations and incurs constant time execution. To efficiently update b_{sp}, the following operations are employed:

1. Find the position of K_{new} within b_{ks} and set r to this index position. This implies that the kernels, x_{r-1} and x_{r+1}, are K_{new}'s adjacent neighbors
2. Update b_{sp} as follows:

$${b_{sp}}^{(new)} = {b_{sp}}^{(old)} - x_{r+1}x_{r-1} + x_{r-1}x_r + x_r x_{r+1}$$

4.3. Kernel Maintenance

In order to maintain the kernels in a fixed memory environment, a kernel clustering paradigm is employed. If the size of the *kernel set* is M, then the insertion of K_{new} will cause an overflow and invoke the merger of two kernels at the corresponding bin. Let $G^{(1)}(x)$ and $G^{(2)}(x)$ be weighted kernels, then the merged kernel, $G^{(merge)}(x)$, is determined by utilizing a kernel merging approach as follows [12]:

$$G^{(1)}(x) = w^{(1)}K\left(\frac{x_1 - x}{h}\right)$$
$$G^{(2)}(x) = w^{(2)}K\left(\frac{x_2 - x}{h}\right)$$
$$G^{(merge)}(x) = (w^{(1)} + w^{(2)})K\left(\frac{x_{merge} - x}{h}\right)$$
where x_{merge} is the merged kernel center of x_1 and x_2

$G^{(1)}(x)$ and $G^{(2)}(x)$ are selected such that the following L_2 error distance is minimized:

$$L_2 = \int_{-\infty}^{\infty} \left(G^{(1)}(x) + G^{(2)}(x) - G^{(merge)}(x)\right)^2 dx$$

It can be shown that the L_2 distance increases proportionally with the kernel distance, hence the mergers only occur between adjacent kernels. It can be shown that minimizing L_2 error can be done in constant time [11].

The remainder of this section presents the time-based sliding window algorithm which ensures that all elements in *kernel set* are within the time range, $[\tau_{min}, \tau_{max}]$. The algorithm is followed by a description of the density evaluation which leverages upon the *bin list* structure for efficient computation. Lastly, the selected kernel function and bandwidth form are provided.

Time-based Sliding Window: Let τ_{tl} be the length of the time window. To produce a sliding window, set τ_{max} to the current

time and set $\tau_{min} = \tau_{max} - \tau_{tl}$. The *time list* is implemented as a First-In-First-Out queue with the head node being the oldest timestamp. When K_{new} is inserted into the *kernel set*, the time handling algorithm inserts the kernel's arrival time, g_{ta}, and the kernel's extended time span, g_{ets}, (i.e., time to remain in window *after* expiration and initially set to 0) to the tail of *time list*. Assume that two kernels, $G^{(1)}$ and $G^{(2)}$, are merged to become $G^{(merge)}$. The *time list* updating process proceeds as follows:

1. Remove the corresponding *time list* nodes of $K^{(1)}$and $K^{(2)}$
2. Define $G^{(merge)}$'s arrival time and time span as follows:

$$G_{ta}^{(merge)} = \min\left\{g_{ta}^{(1)}, g_{ta}^{(2)}\right\}$$
$$G_{ts}^{(merge)} = \max\left\{g_{ta}^{(1)} + g_{ets}^{(1)}, g_{ta}^{(2)} + g_{ets}^{(2)}\right\} - g_{ta}^{(merge)}$$

3. Insert $K^{(merge)}$'s arrival time into the time list

Kernel expiration is performed by comparing the *time list*'s head node, deleting all associated kernels with $g_{ta} < \tau_{min}$ and $g_{ets} = 0$ from the *kernel set*, and updating the *bin list*. As for expiring a kernel with $g_{ets} > 0$, assume that the weight of the kernel is uniformly distributed across its time span. Therefore, the kernel will remain in the *kernel set* but its weight will be adjusted to the following:

$$g_w^{(updated)} = \left(1 - \frac{\tau_{min} - g_{ta}^{(old)}}{g_{ets}^{(old)}}\right) g_w^{(old)}$$

where $g_w^{(i)}$ is the weight of kernel i.

Density Evaluation: For a query point x, the evaluation algorithm proceeds as follows:

1. *Bin search and kernel filtering*: Find the relevant set of bins which can potentially contribute a non-zero value to x. Let $h^{(\mathbb{B})}$ be the bandwidth of kernels in bin $\mathbb{B}$, then the bins can be found by scanning the *bin list* and selecting those which fulfill the following condition: $x \cap \left[\min(\mathbb{B}) - h^{(\mathbb{B})}, \max(\mathbb{B}) + h^{(\mathbb{B})}\right] \neq \emptyset$.
2. *Kernel search*: Scan the kernels within each relevant bin and sum the density contribution of kernels whose supports intersect x. Formally stated, let K be a kernel in $\mathbb{B}$, then compute the sum of kernel density contributions where $x \cap \left[K - h^{(\mathbb{B})}, K + h^{(\mathbb{B})}\right] \neq \emptyset$.

The evaluation technique presented above capitalizes upon the bin list structure by effectively pruning kernels which do not contribute to the final estimation result. Furthermore, the bandwidth is computed from the bin features in constant time. Discussion on the exact bandwidth form is given below.

Kernel Function and Bandwidth Form Selection: The class of admissible kernel functions must satisfy the conditions of Theorem 3.1 which requires that the kernels be (1) radially-symmetric and (2) compactly supported. The compact support property also eases the burden of computing the density estimates by eliminating kernels with $\left|\frac{x}{h}\right| \geq 1$. Some kernels of the admissible class are the Bi-weight, Rectangular, and Epanechnikov kernels [20]. It has been shown that the Epanechnikov kernel minimizes the asymptotic mean integrated squared error (AMISE) and therefore it is optimal amongst all other kernels [15]. The Epanechnikov kernel is given as follows:

$$K(x) = \frac{3}{4}(1 - x^2) \text{ for } |x| < 1, \text{ and 0 otherwise}$$

Due to Epanechnikov kernel's compact support, radial-symmetry, and optimality w.r.t. the ASIME, this kernel is chosen for the proposed LR-KDE.

Each bin of the LR-KDE describes a region of similar density, hence, the captured distribution within each bin can be expected to be unimodal. Also recall that the LR-KDE assigns a unique bandwidth to each bin. Therefore, for each bin, the chosen bandwidth form is the *normal rule* which has been shown to perform well under a wide range of unimodal distributions [20]:

$$h^{(\mathbb{B})} = \sqrt{5}\sigma^{(\mathbb{B})} \sqrt[-5]{n^{(\mathbb{B})}}$$

where $\sigma^{(\mathbb{B})}$ is the kernel centers' standard deviation of bin $\mathbb{B}$ and $n^{(\mathbb{B})}$ is the number of kernels in bin $\mathbb{B}$.

Note that $\sigma^{(\mathbb{B})}$ can be directly calculated from the bin features i.e., $\sigma^{(\mathbb{B})} = \sqrt{\frac{b_{wcss}}{b_{ws}} - \left(\frac{b_{wcs}}{b_{ws}}\right)^2}$, therefore, computing the bandwidth, $h^{(\mathbb{B})}$, is achieved in $O(1)$ time.

5. ANALYSIS

This section provides the consistency results and cost complexities of the LR-KDE. An important criterion for any KDE is its consistency, that is, as the number of samples approaches infinity, the KDE converges to the true density. The time/space complexities of the online maintenance technique and density evaluation approach are analyzed to guarantee that the LR-KDE heeds the constraints of the data stream environment.

Asymptotic Consistency

Assume that all n samples are uniquely accessible and continuously persistent. To prove the consistency of the local region KDE, it suffices to show that the density estimate within a local region fulfills Parzen's condition for the asymptotic convergence of a KDE [17]. Parzen provides a sufficient condition described as follows:

If kernel $K(\cdot)$ is a bounded Borel function with

$$\int |K(t)|\, dt < \infty, \int K(t)\, dt = 1 \text{ and } |tK(t)| \to 0 \text{ as } |t| \to \infty$$

and bandwidth h_n indexed on n sample points satisfies

$$h_n \to 0 \text{ and } nh_n \to \infty \text{ as } n \to \infty,$$

then for the KDE, $\hat{f}(x)$, and true density, $f(x)$,

$$\hat{f}(x) \to f(x) \text{ in probability as } n \to \infty$$

Since a local region employs the Epanechnikov kernel, the kernel conditions given by Parzen are completely satisfied. Recall that the bandwidth selected for a local region is the normal rule: $h_n = \sqrt{5}\sigma \sqrt[-5]{n}$. Hence, it can be seen that the following holds: $h_n \to 0$ as $n \to \infty$. Therefore, Parzen's first bandwidth condition is satisfied. As for the second condition, note that a local region is continuous and has compact support, which implies

$$\sup(\sigma) = c, \text{ where } c \text{ is a constant}$$

which implies

$$\frac{h_n}{n} = \frac{\sqrt{5}\sigma \sqrt[-5]{n}}{n} = \frac{\sqrt{5}c}{n^{6/5}} \to 0 \text{ as } n \to \infty \Rightarrow nh \to \infty$$

Therefore,

$$\hat{f}(x) \to f(x) \text{ in probability as } n \to \infty$$

Online Maintenance Complexities

Let $InsertCost_{total}(K_{new})$ be the total cost of inserting K_{new}, then the total cost of the insertion procedure is:

$$InsertCost_{total}(K_{new}) = InsertCost_{binlist}(K_{new}) + InsertCost_{kernellist}(K_{new}) + InsertCost_{timelist}(K_{new})$$

Bin list cost: $InsertCost_{binlist}(K_{new})$ is composed of a sequential search over the *bin list* and merging a pair of bins. Hence the cost of inserting K_{new} into the *bin list* is:

$$InsertCost_{binlist}(K_{new}) = O(Q)$$

Kernel set cost: Similar to the bin insertion cost, the insertion to the *kernel list* is dominated by the kernel search and the L_2 error updates for each pair of adjacent kernels within the affected bin. Let R be the number of kernels in the updated bin, then the total cost of the *kernel list* insertion is:

$$InsertCost_{kernellist}(K_{new}) = O(R)$$

Time list cost: The dominant cost for inserting into the *time list* is the removal of all expired kernels. This operation involves updates on the L_2 errors and bin statistics. Let S be the number of expired kernels and T be the total number of kernels within a bin of an expired kernel, then the cost of inserting K_{new} into the *time list* is:

$$InsertCost_{timelist}(K_{new}) = Remove_{kernellist}(K_{new}) + Update_{bin} + UpdateL_{2_{kernellist}} = O(S) + O(Q) + O(T) = O(S)$$

Since $Q, R, S \leq M$, where M is the maximum size of the *kernel list*, the total cost of inserting K_{new} is:

$$InsertCost_{total}(K_{new}) = O(M)$$

Total space cost: The total space cost of the online maintenance algorithm is derived from storing the three primary structures, *bin list*, *kernel list*, and *time list* in memory:

$$SpaceCost_{total} = O(Q) + O(M) + O(M) = O(M)$$

The above analysis shows that the time and space complexities of the proposed online kernel maintenance algorithm are $O(M)$. Since M is fixed, the proposed maintenance strategy provides constant runtime and space complexities which meet the linear-pass constraint.

Density Evaluation Complexities

A single density evaluation composes of a sequential search of the *bin list* and a scan of all kernels which provides a non-zero contribution to the query point. Let $EvalCost_{total}(x)$ be the total cost of determining the density at x, B be the number of bins which intersects x, and J be the number of contributing kernels, then the total evaluation cost is:

$$EvalCost_{total}(x) = SearchCost_{binlist}(x) + SearchCost_{kernellist}(x) = O(B) + O(J)$$

Since $B \leq J \leq M$, the total evaluation runtime cost is:

$$EvalCost_{total}(x) = O(B) + O(J) = O(M)$$

In practical applications, only a fraction of the kernels contributes to x, which implies that $B + J \ll M$. Therefore, the total evaluation cost can be much less than the asymptotic cost described above. The space cost of the density evaluation is $O(1)$ since the evaluation algorithm makes use of a single counter to store the current density sum. Similar to the time and space complexities of the maintenance algorithm, the evaluation runtime and space costs are also constant.

6. EXPERIMENTS

A set of comprehensive experiments have been conducted to validate the effectiveness and efficiency of the proposed online LR-KDE. The experiments focused on three core metrics: estimation quality, maintenance time, and density evaluation time. Other existing online KDE techniques were included for performance comparisons. The experiments applied a battery of synthetic and real-world data sets to study the effects of various streaming conditions on the LR-KDE. The experiments were conducted on a Windows Server 2003 Enterprise Edition (32-bit) operating system. The hardware platform was a 2.0 GHz Intel Pentium Dual with 3 GB of RAM.

6.1. Experiment Design

The data sets were comprised of 2 synthetic and 4 real-world time series data. The first 25K (if available) data samples were used for the experiments. A description of the data sets is shown in the table below:

Table 2. Experimental data sets

Name	Type	Description	Size
MIX2	Synthetic	Time series randomly generated from the following mixture of Normal distributions: $F_2 = \frac{1}{2}\left(N(20,1) + N(51,1.35^2)\right)$	25K
MIX8	Synthetic	Time series randomly generated from the following mixture of Normal distributions: $F_1 = \frac{1}{8}(N(20,1.25^2) + N(24,1) + N(25,2.75^2) + N(31,1.4^2) + N(37,.95^2) + N(40,3.5^2) + N(41,.75^2) + N(44,1.5^2))$	25K
EEG	Real	EEG of a rat in wake/sleep cycle [14]	25K
POWER	Real	Power demand from a Dutch research facility [14]	25K
ROBOT	Real	Accelerometer measurements of a Sony Aibo Robot playing soccer [14]	24.5K
TRAFFIC	Real	Car volume readings from a loop detector near a California baseball stadium [1, 4]	25K

KDE Techniques and Parameters

Table 3 provides all of the evaluated techniques and parameters:

Table 3. Evaluated Online KDE Techniques

Name	Technique	Parameter
Sequence sample KDE	Sequence sample-based KDE [21, 22]	Max. # of kernels = 1000
Time sample KDE	Time sample-based KDE using the priority-sample algorithm [6]	Max. # of kernels = 1000
M-Kernel KDE	Variable bandwidth cluster KDE [25]	Max. # of kernels = 1000 Simplex max. iter. = 5000
Heinz KDE	List-based cluster KDE [12]	Max. # of kernels = 1000
LR-KDE	Proposed online Local Region KDE	Max. # of kernels = 1000 $\mu = 1, 4 \leq Q \leq 8$

For all of the evaluated techniques in Table 3 (except for time sample KDE), the time windows were set to be the total length of the data stream.

Test Methodology

The first component of the experiments was to measure the estimation quality of the online KDE techniques. This was accomplished by establishing the ground truths for all data sets. In the synthetic case, the exact density structures are given in Table 2. For the real-world data sets, the true densities are defined to be the density estimates produced by the offline AKDE. For the AKDE, the nearest neighbor KDE was used as the pilot estimate [20]. The error measure used was the Root Mean Square Error (RMSE) which is defined as follows:

$$RMSE(\rho) = \sqrt{\frac{1}{1000}\sum_{i=1}^{1000}\left(f(x_i) - \hat{f}^{(\rho)}(x_i)\right)^2}$$

where $\hat{f}^{(\rho)}(\cdot)$ is the ρ density estimation technique and $x_1 .. x_{1000}$ are query points which uniformly divide the entire span of the distribution.

The maintenance time of an online KDE is defined as the *total* amount of time required to insert and process a given set of data points. This measures the efficiency of the online KDE in updating its kernel structures to match the current stream. The density evaluation time is defined as the *average* time to evaluate and compute a single density query. The density evaluation time was measured after all of the data points were processed. In these experiments, 10 trials were conducted for each evaluation component and the averaged results were reported.

6.2. Experiment Results and Discussion

For each metric and data set, the LR-KDE was evaluated against the existing stream-based KDE techniques. The following provides the experiment results.

Estimation Quality: Figure 2 gives the estimation quality results of all data sets and KDE techniques. Each graph represents the estimation errors for a particular data set where the *x-axis* is the number of processed sample points and the *y-axis* is the RMSE of the density estimates. The experiment showed that the LR-KDE provided lower RMSE than all competing techniques in the MIX2, MIX8, ROBOT, TRAFFIC, POWER, and EEG data sets. The LR-KDE produced significant RMSE reductions in MIX2 and MIX8 with errors that were at most half of the next best performing technique. Note that the time sample and sequence sample KDEs provided almost identical performance in MIX2 and MIX8.

The LR-KDE converged as more samples were processed in the MIX2, MIX8, POWER, and EEG data sets. However, for TRAFFIC and ROBOT, the RMSE of LR-KDE increased at 20K and 24.5K points, respectively. This behavior was similarly exhibited in other techniques such as M-Kernel KDE. In the TRAFFIC data set, the sequence sample KDE produced a drastic change in its RMSE at the 20K mark. All of these observations suggest the presence of concept drifts in the POWER and EEG data sets.

Figures 3 and 4 show the plotted estimates of MIX2 and MIX8 by two of the lowest error attaining techniques, LR-KDE and Heinz KDE. The *x-axis* represents the query points and the *y-axis* shows the density. MIX2 and MIX8 possess multiple isolated modes which can be difficult to estimate with a single bandwidth KDE. For example, the Heinz KDE tended to oversmooth the distributions as indicated by the underestimated peaks and overestimated valleys. The oversmoothing can be attributed to Heinz KDE's use of the *normal rule* bandwidth which is known to oversmooth multimodal distributions [20]. In an attempt to improve the accuracy of Heinz KDE, the only available parameter, kernel size, was increased from 1K to 100K. The increased kernel size produced ≤ 1% improvement in the RMSE and showed no observable difference in the plotted estimates. When the kernel size for LR-KDE was increased to 100K, it resulted in 5.5% (MIX2) and 7.8% (MIX8) improvements in the RMSE. Although the LR-KDE employs the *normal rule* bandwidth, it restricts uniform bandwidth assignment to regions of similar densities. As a result, the LR-KDE identified all of the modes and accurately captured the peaks and valleys.

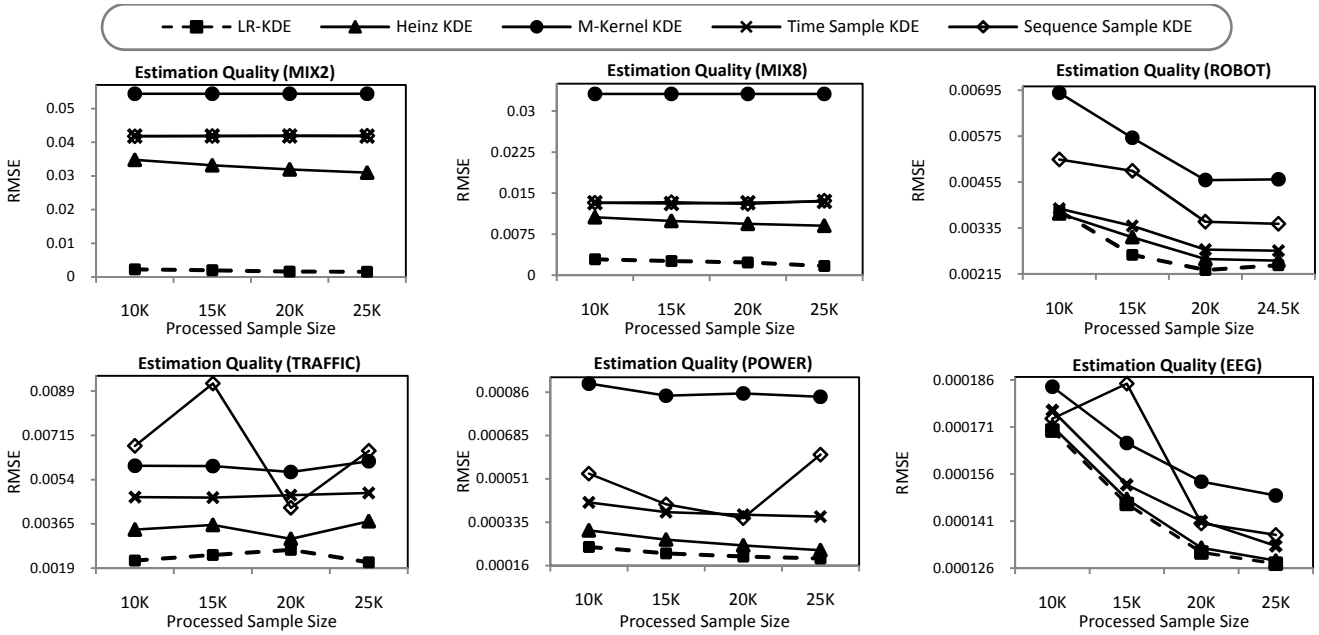

Figure 2. Estimation quality of LR-KDE with increasing sample points

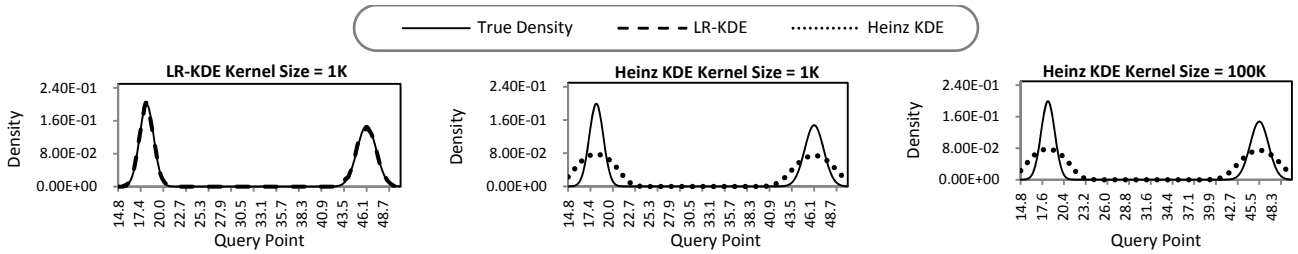

Figure 3. Plots of estimated densities by LR-KDE and Heinz KDE for MIX2

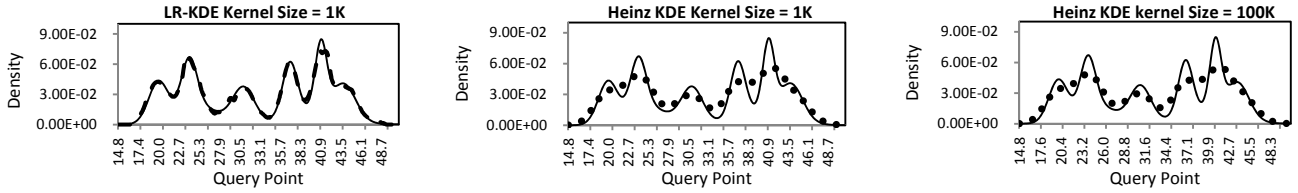

Figure 4. Plots of estimated densities by LR-KDE and Heinz KDE for MIX8

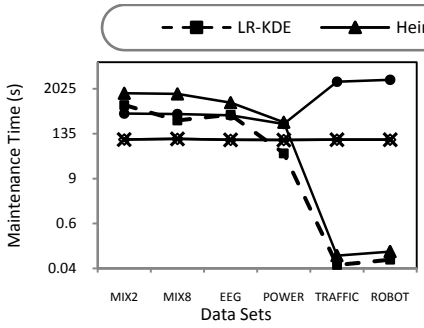

Figure 5. Log scaled maintenance time of all data sets

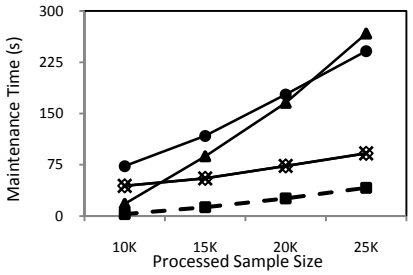

Figure 6. Maintenance times of POWER

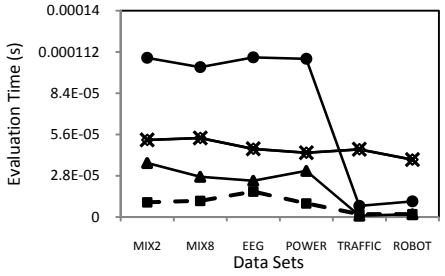

Figure 7. Avg. density evaluation time for a single query

Maintenance Time: Figure 5 illustrates the impact of the various data streams on LR-KDE's kernel maintenance times. The *x-axis* indicates the data sets and the *y-axis* measures the maintenance times for processing the entire data. The real-world POWER, TRAFFIC, and ROBOT data sets showed the most improvements for the LR-KDE. This observation is attributable to the data sets' largely ordered samples which reduced the amount of scans the LR-KDE needed to perform. In the MIX8 and EEG data sets, the LR-KDE provided lower times than all of the non-sample-based approaches. Note that the sample-based techniques produced nearly identical results within the various data sets.

Figure 6 shows the relationship between maintenance time and sample size. The *x-axis* is the number of samples processed and the *y-axis* is the maintenance time. Due to space constraint, only the POWER data set is shown but similar trends can be observed in the other data sets. All of the KDE techniques exhibited times that were linear to the sample size; however, LR-KDE and Heinz KDE provided the lowest cost rates in POWER, TRAFFIC, and ROBOT. The linear trend of LR-KDE is also consistent with the analyses of Section 5. Standard deviation for all trials was ≤ 5%.

Density Evaluation Time: Figure 7 shows the density query times of all data sets. The *x-axis* represents a specific data set and the *y-axis* gives the average evaluation time for a single density query. For MIX2, MIX8, EEG, and POWER data sets, the LR-KDE consistently produced lower evaluation times than all of the competing techniques. For TRAFFIC and ROBOT, the LR-KDE performed equally to Heinz KDE but better than the other techniques. For the sample-based KDE, regardless of the kernel function employed, the entire kernel set must be scanned to generate a density estimate. This approach resulted in a consistent, but higher evaluation times than the LR-KDE and Heinz KDE. In summary, the results demonstrated that the LR-KDE evaluation performance was better than or at least equal to all of the competing techniques. Standard deviation for all trials was ≤ 10%.

Discussion

Application of the local region concept to kernel density estimates has shown to be effective in modeling the local density features in data stream. As a result, the LR-KDE provided superior estimation quality over the competing techniques in both real-world and synthetic data sets. The LR-KDE was also able to improve the estimation accuracy in data sets which did not exhibit strong localities (e.g., predominantly unimodal data sets such as ROBOT). Since real-world data streams can exhibit strong local features (e.g., multiple random processes), it can be expected that stream mining applications would benefit from the use of the LR-KDE.

A concrete mining task that would benefit from LR-KDE's improved estimation quality is the density-based clustering. In particular, those density-based approaches that employ bump hunting algorithms for determining the cluster centers [20]. For example, Figures 3 and 4 show that the LR-KDE captures and differentiates all of the modes almost exactly which would allow the bump hunting method to optimally isolate the cluster centers.

In contrast, the next best performing technique (Heinz KDE) could not capture some of these modes which would increase the likelihood for the bump hunting algorithm to dismiss some potentially vital clusters. In the context of a real-time surveillance application, such false dismissals could indicate missed emergent events.

Density estimation can also be applied to the problem of outlier detection. An outlier can be defined as a sample point whose probability of occurrence falls below a predefined threshold [21]. Suppose that density estimates are employed to perform the above outlier detection scheme, then the rates of false dismissals are dependent on the accuracy of the estimated model. For instance, if the estimate is oversmoothed, then the rate of false dismissals may be increased in regions of low probability. In these regions, the true density of the sample points may be much smaller than their estimated values. Hence, a more accurate density estimation model, such as LR-KDE, can help reduce the false dismissals and improve the outlier detection performance.

LR-KDE also improved the computational efficiency over the competing techniques. Local regions allow non-relevant kernels to be pruned from further processing which results in the simultaneous reduction of maintenance and query times. Furthermore, the LR-KDE retained the same order of space cost as the other online KDE techniques. These cost reductions are essential to stream mining tasks where results need to be furnished in real-time and processed in a fixed-size memory environment.

7. CONCLUSION

This paper addresses the issue of developing an efficient and asymptotically consistent online adaptive density estimation technique to meet the stringent constraints of the data stream environment. In that endeavor, we propose an online and local region based AKDE framework (LR-KDE) for univariate streams. The contributions of this work include (1) the first approach of its kind to provide AKDE over data streams, (2) the introduction of the local regions to effectively approximate the AKDE with guaranteed asymptotic consistency, and (3) the design of linear-pass algorithms to maintain and compute kernel density estimates over a time-based sliding window. Theoretical analyses are provided to validate the asymptotic consistency and computational complexities of the LR-KDE. Experiments demonstrated that the LR-KDE enhanced estimation quality, improved maintenance performance, and reduced density evaluation time over the existing techniques. Future work in developing a multivariate local region based KDE will be investigated.

REFERENCES

[1] "Freeway Performance Measurement System (PeMS) [http://pems.eecs.berkeley.edu]."

[2] C. Aggarwal, "A framework for diagnosing changes in evolving data streams," in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, San Diego, California, USA, pp. 575-586, 2003.

[3] C. Aggarwal and P. S. Yu, "A survey of synopsis construction in data streams," in *Data Streams: Models and Algorithms*, C. Aggarwal, Ed. New York: Springer Science and Business Media, pp. 169-202, 2007.

[4] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]," Irvine, CA: University of California, School of Information and Computer Science, 2007.

[5] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," in *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Madison, Wisconsin, USA, pp. 1-16, 2002.

[6] B. Babcock, M. Datar, and R. Motwani, "Sampling from a moving window over streaming data," in *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, California, USA, pp. 633-634, 2002.

[7] P. Gibbons, Y. Matias, and V. Poosala, "Fast incremental maintenance of approximate histograms," *ACM Transactions on Database Systems* vol. 27, pp. 261-298, 2002.

[8] A. Gilbert, Y. Kotidis, S. Muthukrishan, and M. J. Strauss, "How to summarize the universe: dynamic maintenance of quantiles," in *Proceedings of the 28th International Conference of Very Large Data Bases*, Hong Kong, China, pp. 454-465, 2002.

[9] A. Gray and A. Moore, "Rapid evaluation of multiple density models," in *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics*, Key West, Florida, USA, 2003.

[10] S. Guha, N. Koudas, and K. Shim, "Approximation and streaming algorithms for histogram construction problems," *ACM Transactions on Database Systems,* vol. 31, pp. 396-438, 2006.

[11] C. Heinz, "Density estimation over data streams," in *Mathematics*. Phd: Phillipps-University Marburg, 2007.

[12] C. Heinz and B. Seeger, "Towards kernel density estimation over streaming data," in *Proceedings of the 13th International Conference on Management of Data*, Delhi, India, pp. 91-102, 2006.

[13] Y. Ioannidis, "The history of histograms (abridged)," in *Proceedings of the 29th International Conference on Very Large Databases*, Berlin, Germany, pp. 19-30, 2003.

[14] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana, "The UCR Time Series Classification/Clustering [http://www.cs.ucr.edu/~eamonn/time_series_data]," 2008.

[15] T. Ledl, "Kernel density estimation: theory and application in discriminant analysis," *Austrian Journal of Statistics,* vol. 33, pp. 267-279, 2004.

[16] L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani, "Streaming-data algorithms for high-quality clustering," in *Proceedings of the 18th IEEE International Conference on Data Engineering*, San Jose, CA, USA, pp. 685-694, 2002.

[17] E. Parzen, "On estimation of a probability density function and mode," *Annals of Mathematical Statistics,* vol. 33, pp. 1065-1076, 1962.

[18] S. R. Sain and D. W. Scott, "On locally adaptive density estimation," *Journal of the American Statistical Association,* vol. 91, pp. 1525-1534, 1996.

[19] D. W. Scott, *Multivariate Density Estimation*. New York: Wiley & Sons, 1992.

[20] B. W. Silverman, *Density estimation for statistics and data analysis*. London: Chapman and Hall, 1986.

[21] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *Proceedings of the 32nd International Conference on Very Large Databases*, Seoul, Korea, pp. 187-198, 2006.

[22] E. J. Wegman and D. J. Marchette, "On some techniques for streaming data: a case study of internet packet headers," *Journal of Computational and Graphical Statistics,* vol. 12, pp. 1-22, 2003.

[23] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, Montreal, Canada, pp. 103-114, 1996.

[24] T. Zhang, R. Ramakrishnan, and M. Livny, "Fast density estimation using CF-kernel for very large databases," in *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, CA, USA, pp. 312-316, 1999.

[25] A. Zhou, Z. Cai, L. Wei, and W. Qian, "M-Kernel merging: towards density estimation over data streams," in *Proceedings of the 8th International Conference on Database Systems for Advanced Applications*, Kyoto, Japan, pp. 285-292, 2003.