Large-scale Spatiotemporal Kernel Density Visualization

Tsz Nam Chan*, Pak Lon Ip†, Bojian Zhu‡, Leong Hou U†, Dingming Wu*, Jianliang Xu‡, Christian S. Jensen§
*College of Computer Science and Software Engineering, Shenzhen University
{edisonchan, dingming}@szu.edu.cn
†Department of Computer and Information Science, University of Macau
{paklonip, ryanlhu}@um.edu.mo
‡Department of Computer Science, Hong Kong Baptist University
{csbjzhu, xujl}@comp.hkbu.edu.hk
§Department of Computer Science, Aalborg University
csj@cs.aau.dk

2025 IEEE 41st International Conference on Data Engineering (ICDE) | 979-8-3315-3603-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICDE65448.2025.00015

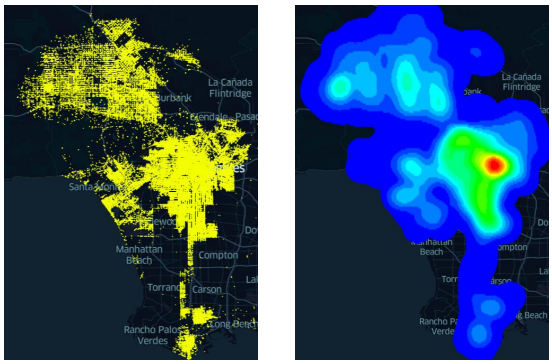
***Abstract*—Spatiotemporal kernel density visualization (STKDV) is used extensively for many geospatial analysis tasks, including traffic accident hotspot detection, crime hotspot detection, and disease outbreak detection. However, STKDV is a computationally expensive operation, which does not scale to large-scale datasets, high resolutions, and a large number of timestamps. Although a recent approach, the sliding-window-based solution (SWS), reduces the time complexity of STKDV, it (i) is unable to reduce the time complexity for supporting STKDV-based exploratory analysis, (ii) is not theoretically efficient, and (iii) does not provide optimization techniques for bandwidth tuning. To eliminate these drawbacks, we propose a prefix-set-based solution (PREFIX) that encompasses three methods, namely $\text{PREFIX}_{\text{single}}$ (addressing (i)), $\text{PREFIX}_{\text{multiple}}$ (addressing (ii)), and $\text{PREFIX}_{\text{tuning}}$ (addressing (iii)). We offer theoretical and practical evidence that PREFIX is capable of outperforming the state-of-the-art solution (SWS). In particular, PREFIX achieves at least 115x to 1,906x speedups and is the first solution that can efficiently generate multiple high-resolution STKDVs for the large-scale New York taxi dataset with 13.6 million data points.**

I. Introduction

Heatmaps (or hotspot maps) are important tools that enable domain experts to visualize and analyze geospatial data. Among the heatmap techniques, kernel density visualization (KDV) [27], [78] is a classic technique that has been used widely in many domains. For example, transportation experts [18], [35], [43], [58], [89] and criminologists [50], [56], [73], [93], [96], [100] employ KDV to discover traffic accident hotspots and crime hotspots, respectively, in geographical regions, and epidemiologists [37], [49], [52], [54] employ KDV to analyze disease outbreaks. Fig. 1 shows an example use of KDV for discovering the crime hotspots in Los Angeles.

This work was supported by the Natural Science Foundation of China under grants 23IAA00610, 62202401, and 62372308, the Guangdong Basic and Applied Basic Research Foundation under grants 2023A1515011619 and 2023B1515130002, the Science and Technology Development Fund Macau SAR (0003/2023/RIC, 0052/2023/RIA1, 0031/2022/A, 001/2024/SKL for SKL-IOTSC), the Research Grant of University of Macau (MYRG2022-00252-FST), Shenzhen-Hong Kong-Macau Science and Technology Program Category C (SGDX20230821095159012), and Wuyi University Hong Kong and Macau joint Research Fund (2021WGALH14). Dingming Wu is the corresponding author of this paper.

Note that there was one crime hotspot from 1st January 2010 to 31st December 2019. Due to the popularity of KDV, many commonly used visualization software packages, e.g., Deck.gl [3], Seaborn [13], and Leaflet [5], geographical software packages, e.g., QGIS [10] and ArcGIS [1], and scientific software packages, e.g., Scikit-learn [65] and Scipy [85], can also support KDV.



(a) Crime events in Los Angeles (b) Hotspot map

Fig. 1: A hotspot map (based on KDV) for crime events in Los Angeles from 1st January 2010 to 31st December 2019, where the red region is a hotspot.

However, KDV ignores the occurrence time of each location data point (or event) and cannot show how hotspots change over time. As pointed out by domain experts [37], [50], [53]–[55], [58], many geographical events (e.g., traffic accidents, crime events, and disease outbreaks) are time-dependent. Using crime events in Los Angeles as an example (see Fig. 1a), Fig. 2 shows that the hotspot distributions change significantly over time. Therefore, simply using KDV (as in Fig. 1b) may not reveal the real distribution of crime events. To tackle this issue, scientists in the geographical domain [18], [37], [49]–[51], [53], [55], [56], [58], [83] have recently adopted the more informative spatiotemporal kernel density visualization (STKDV) that can generate time-dependent hotspot maps (see Fig. 2). In order to generate this kind of visualization, we need to color each pixel-timestamp pair $(\mathbf{q}, t_i)$ based on computing

2375-026X/25/$31.00 ©2025 IEEE
DOI 10.1109/ICDE65448.2025.00015

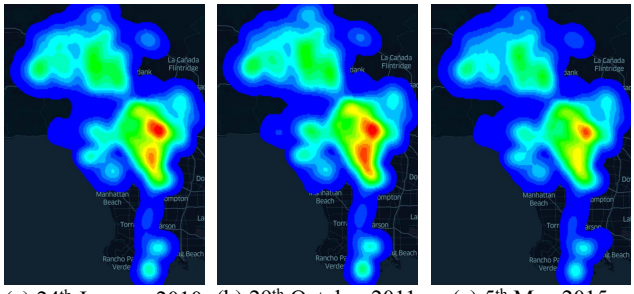
TABLE I: Some representative spatial and temporal kernel functions, where $dist$ (including $dist(\mathbf{q}, \mathbf{p})$ and $dist(t_i, t_\mathbf{p})$), b_σ, and b_τ denote the Euclidean distance, the bandwidth of the spatial kernel, and the bandwidth of the temporal kernel, respectively.

Kernel	$K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p})$	$K^{(b_\tau)}_{\text{time}}(t_i, t_\mathbf{p})$	References
Uniform	$\begin{cases} \frac{1}{b_\sigma} & \text{if } dist(\mathbf{q}, \mathbf{p}) \le b_\sigma \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} \frac{1}{b_\tau} & \text{if } dist(t_i, t_\mathbf{p}) \le b_\tau \\ 0 & \text{otherwise} \end{cases}$	[90]
Epanechnikov	$\begin{cases} 1 - \frac{1}{b_\sigma^2} dist(\mathbf{q}, \mathbf{p})^2 & \text{if } dist(\mathbf{q}, \mathbf{p}) \le b_\sigma \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} 1 - \frac{1}{b_\tau^2} dist(t_i, t_\mathbf{p})^2 & \text{if } dist(t_i, t_\mathbf{p}) \le b_\tau \\ 0 & \text{otherwise} \end{cases}$	[37], [51], [87]
Quartic	$\begin{cases} \left(1 - \frac{1}{b_\sigma^2} dist(\mathbf{q}, \mathbf{p})^2\right)^2 & \text{if } dist(\mathbf{q}, \mathbf{p}) \le b_\sigma \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} \left(1 - \frac{1}{b_\tau^2} dist(t_i, t_\mathbf{p})^2\right)^2 & \text{if } dist(t_i, t_\mathbf{p}) \le b_\tau \\ 0 & \text{otherwise} \end{cases}$	[17], [56]

the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (see Equation 1).

$$\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i) = \sum_{(\mathbf{p}, t_\mathbf{p}) \in \widehat{P}} w \cdot K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p}) \cdot K^{(b_\tau)}_{\text{time}}(t_i, t_\mathbf{p}) \quad (1)$$

where w, $\widehat{P}$, $K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p})$, and $K^{(b_\tau)}_{\text{time}}(t_i, t_\mathbf{p})$ denote the normalization (positive) constant, the set of spatiotemporal data points, the spatial kernel function, and the temporal kernel function, respectively. Some representative kernel functions are listed in Table I.



(a) 24th January 2010 (b) 29th October 2011 (c) 5th May 2015

Fig. 2: Using STKDV to visualize hotspots of crime events in Los Angeles with different timestamps.

Nevertheless, generating STKDV is very time-consuming. Given a spatial resolution $X \times Y$, T timestamps, and a set of location data points with size n, a naïve implementation of STKDV takes $O(XYTn)$ time, which does not scale to high resolutions, large-scale datasets, and a large number of timestamps. Worse still, domain experts [37], [40], [77], [95] need to generate multiple STKDVs by tuning the bandwidth parameters of spatial and temporal kernels, i.e., b_σ and b_τ, respectively, in Table I (which will be discussed in Section III), in order to obtain the time-dependent hotspot maps with the best visual quality. This further exacerbates the inefficiency issue of STKDV. In fact, many domain experts [37], [49], [54], [77] have complained about this issue with using STKDV. Using a recent work as an example, Lan et al. [54] mention that "Although temporal extensions for these methods have been proposed in the literature (such as STKDE, space-time interpolation), they are computationally demanding,...".

To address this issue, Chan et al. [21] have proposed the sliding-window-based solution (SWS) that reduces the worst-case time complexity for generating exact STKDV from $O(XYTn)$ to $O(XY(T + n))$. Despite this, the time complexity still depends on XYn, which is not yet efficient enough to support high-resolution STKDV on large datasets. For example, the SWS method cannot generate STKDVs with a 480×320 resolution on datasets with more than 500,000 data points within one hour (see Figure 12 of [21]). In addition, this method requires all timestamps of STKDV to be known in advance. In contrast, domain experts can perform exploratory analysis [61]–[63], where the timestamps are provided on the fly. Furthermore, this work [21] has not provided efficient algorithms to generate multiple STKDVs for different bandwidth parameters.

To further improve the efficiency of STKDV-based data analytics, we ask the following two research questions.

1) *Can we further reduce the time complexity for computing exact STKDV, without increasing the space complexity, in the following two settings?*
   (a) Every timestamp is given on the fly (i.e., $T = 1$).
   (b) The T timestamps are known in advance.
2) *Can we reduce the time complexity for computing multiple exact STKDVs with different spatial and temporal bandwidths, without increasing the space complexity?*

In this paper, we answer the above questions by proposing a prefix-set-based solution (PREFIX) that encompasses three methods. To tackle the first question, we develop $\text{PREFIX}_{\text{single}}$ and $\text{PREFIX}_{\text{multiple}}$ that reduce the time complexity for generating exact STKDV with an on-the-fly timestamp (i.e., $T = 1$) and T known timestamps, respectively. To tackle the second question, we develop $\text{PREFIX}_{\text{tuning}}$ that reduces the time complexity of generating exact STKDVs with multiple spatial and temporal bandwidths. The proposed methods retain the same space complexity for solving these problems. Table II summarizes the worst-case complexity results of our solution, PREFIX, and the state-of-the-art solution, SWS [21], where $X \times Y$, T, n, M, and N denote the spatial resolution, the number of timestamps, the dataset size, the number of spatial bandwidths, and the number of temporal bandwidths, respectively.

TABLE II: Complexity results for computing exact STKDVs.

Problem	Method	Time complexity	Space complexity
STKDV (on-the-fly timestamp)	SWS [21]	$O(XYn)$	$O(XY + n)$
	$\text{PREFIX}_{\text{single}}$ (Section IV-B)	$O(Y(X + n))$ (Theorem 1)	$O(XY + n)$ (Theorem 4)
STKDV (T known timestamps)	SWS [21]	$O(XY(T + n))$	$O(XYT + n)$
	$\text{PREFIX}_{\text{multiple}}$ (Section IV-C)	$O(XYT + Yn)$ (Theorem 2)	$O(XYT + n)$ (Theorem 5)
Bandwidth tuning	SWS [21]	$O(MNXY(T + n))$	$O(MNXYT + n)$
	$\text{PREFIX}_{\text{tuning}}$ (Section IV-D)	$O(M(XYTN + YN))$ (Theorem 3)	$O(MNXYT + n)$ (Theorem 6)

Our experiments on five large-scale location datasets (with up to 5 million data points) show that PREFIX can achieve **speedups of 115x to 1,906x** over the state-of-the-art solution, SWS, in all problem settings. In addition, we conduct a case study to generate 25 high-resolution STKDVs (with $X = 1280$, $Y = 960$, and $T = 32$) for the large-scale New York taxi dataset [8] (with $n = 13.6$ million data points). Compared with the state-of-the-art solution, SWS [21], which takes more than one day to generate a single STKDV, **our solution only takes 23 minutes to generate these 25 high-resolution STKDVs**, which, to the best of our knowledge, is the first study to achieve this scalability in a single-machine setting. Furthermore, we provide a case study that illustrates how to support STKDV-based exploratory operations on the Hong Kong COVID-19 dataset [2]. With the lower time complexity of PREFIX, we show that PREFIX enables real-time performance (< 0.5 seconds) of STKDV-based exploratory operations on this dataset, while SWS needs roughly 20 seconds to generate a single hotspot map.

The rest of the paper is organized as follows. We first review the related work in Section II. Then, we formally define the problems and overview the state-of-the-art solution, SWS, in Section III. Next, we illustrate our solution, PREFIX, in Section IV. After that, we discuss the experimental results in Section V. Lastly, we conclude our paper in Section VI.

II. Related Work

In this section, we review four camps of studies, which are closely related to this work.

Sliding-window-based methods: In the database and data mining communities, sliding-window-based methods have been used extensively to improve the efficiency of many query processing tasks, including aggregate queries (e.g., COUNT, MIN, MAX,...) [42], [57], [79]–[81], [84], [88], [92], frequency queries [15], [44], [45], top-k queries [86], [101], [102], and skyline queries [59], [82], [94]. Since none of these studies focuses on the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (see Equation 1), they cannot be adopted readily to improve the efficiency for supporting STKDV. Recently, Chan et al. [21] successfully developed the first sliding-window-based solution (SWS) to reduce the time complexity for generating STKDV. However, this method achieves inferior theoretical and practical performance compared with PREFIX (see Table II and Section V).

Range-query-based methods: Recall from Table I that only those data points $(\mathbf{p}, t_\mathbf{p}) \in \widehat{P}$ with $dist(\mathbf{q}, \mathbf{p}) \leq b_\sigma$ and $dist(t_i, t_\mathbf{p}) \leq b_\tau$ can contribute to the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (see Equation 1). Therefore, one possible approach is to first find the range query set $R(\mathbf{q}, t_i) = \{(\mathbf{p}, t_\mathbf{p}) \in \widehat{P} : dist(\mathbf{q}, \mathbf{p}) \leq b_\sigma \text{ and } dist(t_i, t_\mathbf{p}) \leq b_\tau\}$ for each pixel-timestamp pair $(\mathbf{q}, t_i)$ and then evaluate $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ based on this range query set $R(\mathbf{q}, t_i)$. Since many index structures [48], [75], [76], [91], e.g., kd-tree [16] and ball-tree [64], have been developed to improve the efficiency for solving range queries, this approach can also improve the efficiency of computing STKDV. However, range-query-based methods do not reduce the time complexity for computing STKDV. In addition, Chan et al. [21] verified that this approach is consistently inferior compared with the SWS method.

Parallel methods: Various studies [21], [37], [38], [49], [77] propose parallel methods to improve the efficiency of computing STKDV. However, most of these studies [37], [38], [49], [77] mainly parallelize the naïve method (with no optimization). Recently, Chan et al. [21] parallelized the state-of-the-art method, SWS, which can significantly improve practical efficiency. Nevertheless, the theoretical improvement of the parallel methods is at most $\mathcal{T}$ times using $\mathcal{T}$ threads (e.g., at most 16 times with 16 threads). Since PREFIX achieves speedups of at least two orders of magnitude over SWS (to be discussed in Section V), the parallel version of SWS is still not competitive with PREFIX in a single machine setting (with a limited number of threads).[1] A more detailed discussion on the parallelization of PREFIX can be found in a supplementary document (see Section 3 of [24]).

Other kernel-density-visualization-related methods: Throughout the past two decades, many studies have been developed for solving other types of kernel density visualization, e.g., spatial kernel density visualization (SKDV) [19], [20], [22], [23], [26]–[28], [31], [39], [46], [47], [66]–[68], [71], [97]–[99], network kernel density visualization (NKDV) [25], [28], [32], [33], [70], and network-temporal kernel density visualization (NTKDV) [41], [72], [74]. However, SKDV does not need to consider the timestamp of a pixel, the timestamp of a data point, and the temporal kernel function (i.e., t_i, $t_\mathbf{p}$, and $K_{\text{time}}^{(b_\tau)}(t_i, t_\mathbf{p})$ in Equation 1, respectively), indicating that these studies do not consider the more complex spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (see Equation 1). Worse still, some studies [19], [22], [26], [31], [39], [46], [47], [66]–[68], [71], [97]–[99] can only provide approximate results for SKDV. Therefore, none of these studies in SKDV can be used directly to efficiently generate exact STKDV. Furthermore, the studies in NKDV and NTKDV [25], [32], [33], [41], [70], [72], [74] aim to efficiently obtain visualizations in a road network, which cannot be extended to support STKDV (as this work does not consider a road network.).

III. Preliminaries

We first formally define two problems, namely STKDV (with two settings) and bandwidth tuning, in Section III-A. Then, we overview the state-of-the-art method, namely sliding-window-based solution (SWS) [21], in Section III-B.

A. Problem Definitions

To compute STKDV, i.e., time-dependent hotspot maps (see Fig. 2), with a spatial resolution $X \times Y$ for a set of location data points (see Fig. 1a), domain experts [37], [50], [58] need to determine the color of each pixel for each timestamp based on the spatiotemporal kernel density function, which is formally stated in Problem 1.

Problem 1 (STKDV)**.** *Given a set of spatiotemporal data points* $\widehat{P} = \{(\mathbf{p}_1, t_{\mathbf{p}_1}), (\mathbf{p}_2, t_{\mathbf{p}_2}), ..., (\mathbf{p}_n, t_{\mathbf{p}_n})\}$ *with size* n*,*

[1] We focus on this setting as domain experts normally adopt the off-the-shelf software packages (e.g., QGIS, ArcGIS, and Scikit-learn) for their analytic tasks, who may not necessarily have many computational resources.

a spatial resolution $X \times Y$, T timestamps t_1, t_2,..., t_T, the bandwidth of the spatial kernel b_σ, and the bandwidth of the temporal kernel b_τ, we need to compute the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ for each pixel $\mathbf{q}$ and timestamp t_i (see Equation 1).

In this problem, we focus on two settings: (1) generate STKDV with each on-the-fly timestamp (i.e., $T = 1$) and (2) generate STKDV with T timestamps that are known in advance.

Based on Problem 1, we further define the bandwidth-tuning problem for STKDV in Problem 2.

Problem 2 (Bandwidth tuning)**.** *Given M bandwidths of the spatial kernel, b_{σ_1}, b_{σ_2},..., b_{σ_M}, and N bandwidths of the temporal kernel, b_{τ_1}, b_{τ_2},..., b_{τ_N}, we need to compute STKDV (see Problem 1) for each pair $(b_{\sigma_u}, b_{\tau_v})$ of a spatial bandwidth and a temporal bandwidth, where $1 \leq u \leq M$ and $1 \leq v \leq N$.*

As a remark, since the Epanechnikov kernel is more popular compared with other kernel functions, we adopt it as both the spatial kernel $K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p})$ and the temporal kernel $K^{(b_\tau)}_{\text{time}}(t_i, t_\mathbf{p})$ in this paper. Nevertheless, all our methods can be extended to other kernel functions in Table I.

B. Overview of SWS: the State-of-the-art Method

Recently, Chan et al. [21] propose the sliding-window-based solution (SWS), which reduces the time complexity for computing STKDV with T timestamps. Observe from Table I that only data points $(\mathbf{p}, t_\mathbf{p})$ with $dist(t_i, t_\mathbf{p}) \leq b_\tau$ can result in $K^{(b_\tau)}_{\text{time}}(t_i, t_\mathbf{p}) > 0$, which possibly contribute to the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (see Equation 1). As such, they establish a sliding window $W(t_i)$ (see Equation 2) on the time-axis (see Fig. 3).

$$W(t_i) = \{(\mathbf{p}, t_\mathbf{p}) \in \widehat{P} : dist(t_i, t_\mathbf{p}) \leq b_\tau\} \tag{2}$$

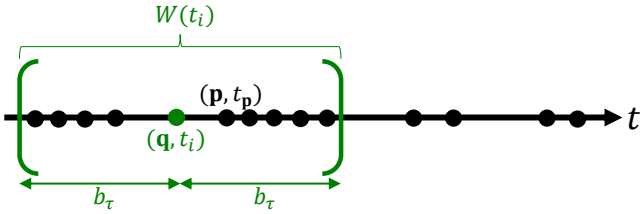


Fig. 3: Only data points $(\mathbf{p}, t_\mathbf{p})$ in the green sliding window $W(t_i)$ (i.e., $dist(t_i, t_\mathbf{p}) \leq b_\tau$) can possibly contribute to the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$.

With the definition of $W(t_i)$, the next step is to represent the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (Equation 1 with the Epanechnikov kernel) as follows.

$$\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i) = \sum_{(\mathbf{p}, t_\mathbf{p}) \in W(t_i)} w \cdot K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p}) \cdot \left(1 - \frac{1}{b_\tau^2} dist(t_i, t_\mathbf{p})^2\right) \tag{3}$$

Based on the simple mathematical derivations, they have:

$$\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i) = w\left(1 - \frac{t_i^2}{b_\tau^2}\right) \cdot S^{(0)}_{W(t_i)}(\mathbf{q}) + \frac{2wt_i}{b_\tau^2} \cdot S^{(1)}_{W(t_i)}(\mathbf{q}) - \frac{w}{b_\tau^2} \cdot S^{(2)}_{W(t_i)}(\mathbf{q}) \tag{4}$$

where:

$$S^{(u)}_{W(t_i)}(\mathbf{q}) = \sum_{(\mathbf{p}, t_\mathbf{p}) \in W(t_i)} t_\mathbf{p}^u \cdot K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p}) \tag{5}$$

Then, once it is possible to efficiently maintain the statistical terms $S^{(u)}_{W(t_i)}(\mathbf{q})$, $0 \leq u \leq 2$, in the sliding window, the density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ can be computed in $O(1)$ time using Equation 4.

By moving this sliding window to the next timestamp t_{i+1} on the time-axis (see Fig. 4), some computations of $S^{(u)}_{W(t_i)}(\mathbf{q})$ can be further shared to $S^{(u)}_{W(t_{i+1})}(\mathbf{q})$ since two consecutive sliding windows, $W(t_i)$ and $W(t_{i+1})$, may overlap with each other (see the yellow region in Fig. 4). With this concept and Equation 4, Chan et al. [21] show that, for a given spatial pixel $\mathbf{q}$, they can compute the density values for all T timestamps in $O(T + n)$ time. Since there are $X \times Y$ pixels, the time complexity of this method becomes $O(XY(T + n))$ for computing STKDV. With M spatial bandwidths and N temporal bandwidths, this method needs $O(MNXY(T + n))$ time to support bandwidth tuning (see Problem 2).

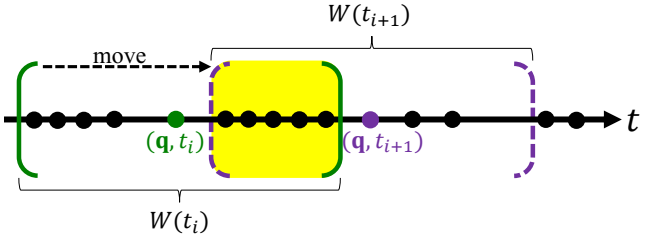


Fig. 4: Moving from the green sliding window $W(t_i)$ to the purple dashed sliding window $W(t_{i+1})$ in order to compute $S^{(u)}_{W(t_{i+1})}(\mathbf{q})$ for the next timestamp t_{i+1}, where the computations for the yellow region in $S^{(u)}_{W(t_i)}(\mathbf{q})$ can be shared to $S^{(u)}_{W(t_{i+1})}(\mathbf{q})$.

Although the SWS method reduces the worst-case time complexity for computing STKDV, the time complexity still depends on XYn. Therefore, this method does not scale to fine-grained spatial resolutions (i.e., large $X \times Y$) and large numbers of data points (i.e., large n). Furthermore, the SWS method adopts the sharing approach for computing the statistical terms between consecutive sliding windows (see Fig. 4), which correspond to consecutive timestamps, in order to efficiently evaluate the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ across T timestamps. Therefore, if domain experts provide each timestamp on-the-fly (i.e., $T = 1$) for conducting the exploratory analysis, this approach cannot be utilized to improve the efficiency for computing STKDV. In addition, SWS does not offer optimization techniques for handling multiple bandwidths.

IV. Our Solution: PREFIX

To advance the state of the art, we propose PREFIX in this section. Specifically, we first summarize the core ideas of PREFIX in Section IV-A. Based on these ideas, we then discuss the methods, PREFIX$_{\text{single}}$, PREFIX$_{\text{multiple}}$, and PREFIX$_{\text{tuning}}$, in Section IV-B, Section IV-C, and Section IV-D, respectively, which reduce the time complexity for computing STKDV (i.e., Problem 1 with two settings) and supporting bandwidth tuning

(i.e., Problem 2). Lastly, we discuss the space complexity of PREFIX in Section IV-E.

A. Core Ideas of PREFIX

Here, we first extend the concept of sliding window $W(t_i)$ (see Equation 2) to the general one $W_{\mathcal{I}}$ (see Equation 6), where $\mathcal{I}$ denotes any time interval in the time-axis.

$$W_{\mathcal{I}} = \{(\mathbf{p}, t_{\mathbf{p}}) \in \widehat{P} : t_{\mathbf{p}} \in \mathcal{I}\} \tag{6}$$

Then, we further define the statistical terms for the window $W_{\mathcal{I}}$ in Equation 7 ($u = 0, 1, 2$ for the Epanechnikov function as the temporal kernel).

$$S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q}) = \sum_{(\mathbf{p},t_{\mathbf{p}})\in W_{\mathcal{I}}} t^u_{\mathbf{p}} \cdot K^{(b_\sigma)}_{\text{space}}(\mathbf{q}, \mathbf{p}) \tag{7}$$

Based on these concepts, we illustrate three core ideas of PREFIX, namely (1) an $O(Y(X + |W_{\mathcal{I}}|))$-algorithm for computing $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ with all pixels $\mathbf{q}$, (2) prefix set, and (3) computation of the statistical terms for multiple prefix sets.

Core idea 1 (An $O(Y(X + |W_{\mathcal{I}}|))$-algorithm for computing $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ with all pixels $\mathbf{q}$): In Fig. 5, note that all grey data points in $W_{\mathcal{I}}$ (see Equation 6) do not depend on the spatial position of each red pixel. Therefore, $W_{\mathcal{I}}$ can be shared across all pixels $\mathbf{q}$.

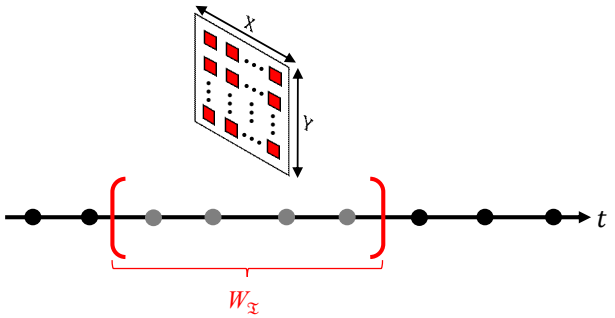


Fig. 5: All data points in the sliding window $W_{\mathcal{I}}$ (i.e., grey circles on the time-axis) can be used to compute the statistical terms $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ for all pixels $\mathbf{q}$ (i.e., red rectangles in the plane).

Based on the above observation, we aim to compute the statistical terms $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ efficiently for all pixels $\mathbf{q}$. To achieve this goal, we consider another problem, called spatial kernel density visualization (SKDV), which is formally stated in Problem 3.

Problem 3. *(SKDV [27]) Given a set of spatial data points $P = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\}$ with size m and a spatial resolution $X \times Y$, we need to compute the kernel density value $D_P(\mathbf{q})$ for each pixel $\mathbf{q}$, where*

$$D_P(\mathbf{q}) = \sum_{\mathbf{p}\in P} w \cdot K^{(b_\sigma)}_{space}(\mathbf{q}, \mathbf{p}) \tag{8}$$

Recently, Chan et al. [27] propose a bucket-based algorithm, SLAM$_{\text{BUCKET}}$, that takes $O(Y(X + m))$ time to solve this problem.

Observe that $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ (see Equation 7) is close to $D_P(\mathbf{q})$ (see Equation 8), by replacing the positive constant w and the set P with $t^u_{\mathbf{p}}$ and $W_{\mathcal{I}}$, respectively. As such, we can modify SLAM$_{\text{BUCKET}}$ to compute $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ in $O(Y(X + |W_{\mathcal{I}}|))$ time with any spatial kernel function in Table I.

As a remark, this is the first work that (1) observes the possibility of sharing $W_{\mathcal{I}}$ across all pixels $\mathbf{q}$ (see Fig. 5) and (2) observes the linkage between $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ (see Equation 7) and $D_P(\mathbf{q})$ (see Equation 8). Without these two observations, SLAM$_{\text{BUCKET}}$ [27] cannot be directly used for solving our problems.

Core idea 2 (Prefix set): We first define the prefix set of the timestamp t, which includes the data points $(\mathbf{p}, t_{\mathbf{p}})$ with $t_{\mathbf{p}} \le t$ in the dataset $\widehat{P}$, where:

$$\widehat{P}(t) = \{(\mathbf{p}, t_{\mathbf{p}}) \in \widehat{P} : t_{\mathbf{p}} \le t\} \tag{9}$$

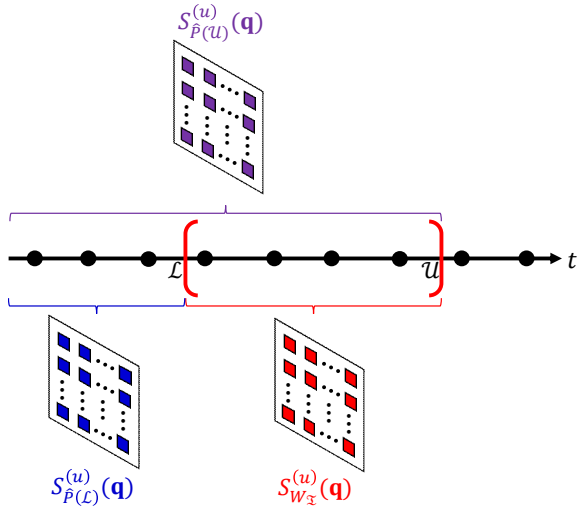


Fig. 6: The statistical terms $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ for the window $W_{\mathcal{I}}$ (i.e., all the red pixels) with the interval $\mathcal{I} = [\mathcal{L}, \mathcal{U}]$ can be computed by subtracting $S^{(u)}_{\widehat{P}(\mathcal{U})}(\mathbf{q})$ (i.e., all the purple pixels) by $S^{(u)}_{\widehat{P}(\mathcal{L})}(\mathbf{q})$ (all the blue pixels), where $\widehat{P}(\mathcal{L})$ and $\widehat{P}(\mathcal{U})$ represent the data points $(\mathbf{p}, t_{\mathbf{p}})$ with $t_{\mathbf{p}} \le \mathcal{L}$ and $t_{\mathbf{p}} \le \mathcal{U}$, respectively.

With this concept, Fig. 6 illustrates how we can then obtain the statistical terms $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ for the window $W_{\mathcal{I}}$ (where $\mathcal{I} = [\mathcal{L}, \mathcal{U}]$) using Equation 10 if the statistical terms for the prefix sets $\widehat{P}(\mathcal{L})$ and $\widehat{P}(\mathcal{U})$, i.e., $S^{(u)}_{\widehat{P}(\mathcal{L})}(\mathbf{q})$ and $S^{(u)}_{\widehat{P}(\mathcal{U})}(\mathbf{q})$, respectively, are available.

$$S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q}) = S^{(u)}_{\widehat{P}(\mathcal{U})}(\mathbf{q}) - S^{(u)}_{\widehat{P}(\mathcal{L})}(\mathbf{q}) \tag{10}$$

Since there are $X \times Y$ pixels, the statistical terms $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ for the window $W_{\mathcal{I}}$ can be computed in $O(XY)$ time once we have $S^{(u)}_{\widehat{P}(\mathcal{L})}(\mathbf{q})$ and $S^{(u)}_{\widehat{P}(\mathcal{U})}(\mathbf{q})$.

Core idea 3 (Computation of the statistical terms for multiple prefix sets): Suppose that we need to compute the statistical terms for the prefix sets $\widehat{P}(t^{(1)})$, $\widehat{P}(t^{(2)})$,..., $\widehat{P}(t^{(L)})$, which correspond to arbitrary timestamps $t^{(1)}$, $t^{(2)}$,..., $t^{(L)}$ (see Fig. 7).

In Fig. 7, note that we can use the sweep line ℓ to compute the statistical terms for each prefix set. Here, we use an inductive approach to show how we can achieve this goal.

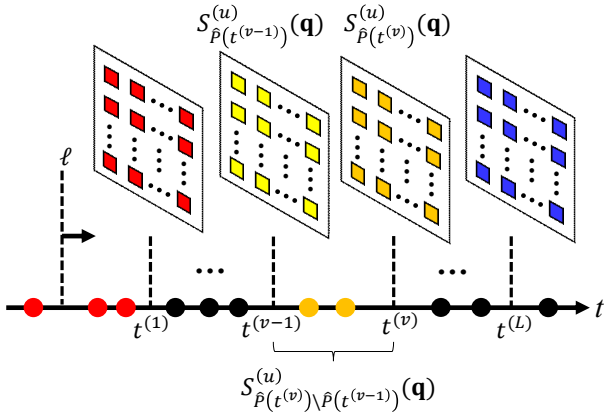


Fig. 7: Using the sweep line ℓ to compute the statistical terms $S^{(u)}_{\widehat{P}(t^{(v)})}(\mathbf{q})$ for each prefix set $\widehat{P}(t^{(v)})$, where we consider L arbitrary timestamps (i.e., $1 \le v \le L$) in this example.

Consider the timestamp $t^{(1)}$. This sweep line ℓ needs to scan and store all the red data points $(\mathbf{p}, t_{\mathbf{p}})$ with $t_{\mathbf{p}} \le t^{(1)}$. Once this line ℓ reaches $t^{(1)}$, we can compute the statistical terms $S^{(u)}_{\widehat{P}(t^{(1)})}(\mathbf{q})$ (i.e., the red pixels) in $O(Y(X+|\widehat{P}(t^{(1)})|))$ time (see the core idea 1) and remove all these red points from the sweep line ℓ (in $O(|\widehat{P}(t^{(1)})|)$ time).

Consider the timestamp $t^{(v)}$, where $v \ge 2$. Suppose that the statistical terms $S^{(u)}_{\widehat{P}(t^{(v-1)})}(\mathbf{q})$ for all yellow pixels are available, the sweep line ℓ needs to scan and store all the orange points $(\mathbf{p}, t_{\mathbf{p}})$ with $t^{(v-1)} < t_{\mathbf{p}} \le t^{(v)}$. Once this sweep line ℓ reaches $t^{(v)}$, we can compute $S^{(u)}_{\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})}(\mathbf{q})$ for all pixels $\mathbf{q}$ in $O(Y(X+|\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})|))$ time (see the core idea 1). Then, we can compute the statistical terms $S^{(u)}_{\widehat{P}(t^{(v)})}(\mathbf{q})$ for all orange pixels in $O(XY)$ time, based on the following equation.

$$S^{(u)}_{\widehat{P}(t^{(v)})}(\mathbf{q}) = S^{(u)}_{\widehat{P}(t^{(v-1)})}(\mathbf{q}) + S^{(u)}_{\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})}(\mathbf{q}) \quad (11)$$

Lastly, we can remove all the orange points from the sweep line ℓ (in $O(|\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})|)$ time).

Algorithm 1 details how to compute the statistical terms for multiple prefix sets. Based on the above discussion, the time complexity of this algorithm is:

$$O\Big(Y(X+|\widehat{P}(t^{(1)})|) + \sum_{v=2}^{L} Y(X+|\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})|)\Big) \quad (12)$$

Since the sets $\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})$ with different v do not intersect (see Fig. 7), we have $\sum_{v=2}^{L}|\widehat{P}(t^{(v)})\setminus\widehat{P}(t^{(v-1)})| \le n$. In addition, $|\widehat{P}(t^{(1)})| \to n$ in the worst case. As such, we can conclude that finding the statistical terms for the prefix sets with respect to L arbitrary timestamps (based on Algorithm 1) takes $O(XYL+Yn)$ time (see Lemma 1).

Lemma 1. *Given a set of spatiotemporal data points $\widehat{P}$ with size n and L arbitrary timestamps, $t^{(1)}$, $t^{(2)}$,..., $t^{(L)}$, Algorithm 1 takes $O(XYL+Yn)$ time to find the statistical terms for the prefix sets with respect to these L timestamps.*

Algorithm 1 Algorithm for Computing Statistical terms of L Prefix Sets (PS_{STAT})

procedure PS_{STAT}(timestamps $t^{(1)}$, $t^{(2)}$,..., $t^{(L)}$, $\widehat{P} = \{(\mathbf{p}_1, t_{\mathbf{p}_1}), (\mathbf{p}_2, t_{\mathbf{p}_2}), ..., (\mathbf{p}_n, t_{\mathbf{p}_n})\}$)
  Initialize the sweep line ℓ, where
    $\ell.t \leftarrow -\infty$ and $\ell.\text{set} \leftarrow \phi$
  while $\ell.t < t^{(L)}$ **do**
    ℓ sweeps the next element e
    if e is $(\mathbf{p}_i, t_{\mathbf{p}_i})$ (where $1 \le i \le n$) **then**
      $\ell.t \leftarrow t_{\mathbf{p}_i}$
      $\ell.\text{set} \leftarrow \ell.\text{set} \cup \{(\mathbf{p}_i, t_{\mathbf{p}_i})\}$
    if e is $t^{(v)}$ (where $1 \le v \le L$) **then**
      $\ell.t \leftarrow t^{(v)}$
      // Consider all pixels $\mathbf{q}$
      if $v = 1$ **then**
        $S^{(u)}_{\widehat{P}(t^{(v)})}(\mathbf{q}) \leftarrow S^{(u)}_{\ell.\text{set}}(\mathbf{q})$ (Core idea 1)
      else
        $S^{(u)}_{\widehat{P}(t^{(v)})}(\mathbf{q}) \leftarrow S^{(u)}_{\widehat{P}(t^{(v-1)})}(\mathbf{q}) + S^{(u)}_{\ell.\text{set}}(\mathbf{q})$ (Equation 11)
      $\ell.\text{set} \leftarrow \phi$
    Return $S^{(u)}_{\widehat{P}(t^{(v)})}(\mathbf{q})$ (where $1 \le v \le L$)

B. PREFIX for Generating STKDV with a Single Timestamp (PREFIX_{single})

Recall from the core idea 1 (see Section IV-A) that the statistical terms $S^{(u)}_{W_{\mathcal{I}}}(\mathbf{q})$ for all pixels $\mathbf{q}$ can be computed in $O(Y(X+|W_{\mathcal{I}}|))$ time no matter which interval $\mathcal{I}$ we choose. Therefore, once we set $\mathcal{I} = [t_i - b_\tau, t_i + b_\tau]$, i.e., $W_{\mathcal{I}} = W(t_i)$ (see Equation 2 and Fig. 3), we conclude that $S^{(u)}_{W(t_i)}(\mathbf{q})$ for all pixels $\mathbf{q}$ can be computed in $O(Y(X+n))$ time (as $|W(t_i)| \to n$ in the worst case). With the available statistical terms $S^{(u)}_{W(t_i)}(\mathbf{q})$, we can compute the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ for all pixels $\mathbf{q}$ with a single timestamp t_i in $O(XY)$ time (by adopting simple matrix arithmetic operations in Equation 4). Theorem 1 states the time complexity of this method (named as $\text{PREFIX}_{\text{single}}$).

Theorem 1. *PREFIX_{single} can compute exact STKDV for a single timestamp (i.e., solving Problem 1 with $T = 1$) in $O(Y(X+n))$ time.*

Compared with the state-of-the-art SWS method (see Section III-B), which takes $O(XYn)$ time, our $\text{PREFIX}_{\text{single}}$ method can further reduce the time complexity to $O(Y(X+n))$ for computing STKDV with a single timestamp.

C. PREFIX for Generating STKDV with Multiple Timestamps ($\text{PREFIX}_{multiple}$)

A simple approach to generate STKDV with T known timestamps is to directly adopt $\text{PREFIX}_{\text{single}}$, which takes $O(TY(X+n))$ time (see Theorem 1). However, the time complexity remains high as TYn is a large value. Here, we describe how the core ideas of PREFIX can be adopted to further reduce the time complexity for this setting. With

the core idea 3 (see Section IV-A), we can first adopt a sweep line ℓ to maintain all statistical terms $S^{(u)}_{\widehat{P}(t_i-b_\tau)}(\mathbf{q})$ and $S^{(u)}_{\widehat{P}(t_i+b_\tau)}(\mathbf{q})$ for the endpoints of the window $W(t_i)$ with every timestamp t_i (see the pixel-planes on the lower time-axis in Fig. 8). Since there are T windows and each window contains two endpoints, we need to maintain $O(T)$ statistical terms for the prefix sets, which takes $O(XYT + Yn)$ time (by setting $L = O(T)$ in Lemma 1). Based on the core idea 2 in Section IV-A, we can then obtain the statistical terms $S^{(u)}_{W(t_i)}(\mathbf{q})$ for each window $W(t_i)$ (see the pixel-planes on the upper time-axis in Fig. 8). Since there are T timestamps (i.e., T windows) and computing $S^{(u)}_{W(t_i)}(\mathbf{q})$ for each $W(t_i)$ takes $O(XY)$ time (see the core idea 2), the time complexity of computing $S^{(u)}_{W(t_i)}(\mathbf{q})$ for all timestamps is $O(XYT)$. By adopting Equation 4, we can compute the spatiotemporal kernel density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ for all pixels $\mathbf{q}$ across T timestamps in $O(XYT)$ time. Based on the above discussion, we conclude that the time complexity of this method, named as PREFIX$_{\text{multiple}}$, for generating STKDV with T timestamps is $O(XYT + Yn)$ (see Theorem 2), which is faster than PREFIX$_{\text{single}}$.

Theorem 2. *PREFIX$_{multiple}$ can compute exact STKDV (see Problem 1) with T timestamps in $O(XYT + Yn)$ time.*

Recall that the state-of-the-art method, SWS, takes $O(XY(T + n))$ time for computing STKDV (see Section III-B). PREFIX$_{\text{multiple}}$ further improves the theoretical result to $O(XYT + Yn)$ time.

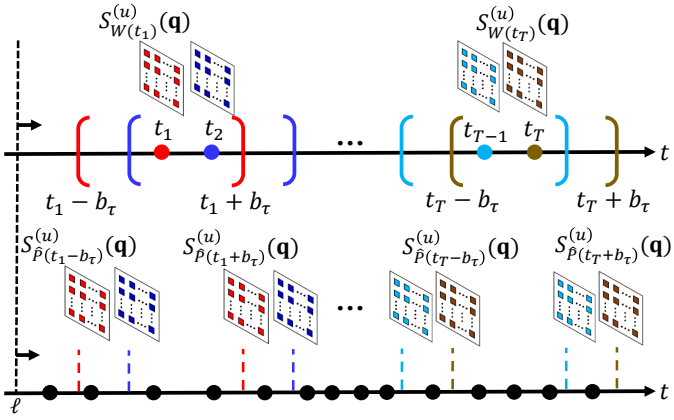


Fig. 8: Maintain the statistical terms of the prefix sets, i.e., $S^{(u)}_{\widehat{P}(t_i-b\tau)}(\mathbf{q})$ and $S^{(u)}_{\widehat{P}(t_i+b\tau)}(\mathbf{q})$, for the two endpoints of a sliding window $W(t_i)$ with each timestamp t_i (e.g., $S^{(u)}_{\widehat{P}(t_1-b\tau)}(\mathbf{q})$, $S^{(u)}_{\widehat{P}(t_1+b\tau)}(\mathbf{q})$, $S^{(u)}_{\widehat{P}(t_T-b\tau)}(\mathbf{q})$, and $S^{(u)}_{\widehat{P}(t_T+b\tau)}(\mathbf{q})$) in order to compute the statistical terms $S^{(u)}_{W(t_i)}(\mathbf{q})$ (e.g., $S^{(u)}_{W(t_1)}(\mathbf{q})$ and $S^{(u)}_{W(t_T)}(\mathbf{q})$) for all pixels $\mathbf{q}$.

D. PREFIX for Bandwidth Tuning (PREFIX$_{tuning}$)

A straightforward approach to solve the bandwidth tuning problem (see Problem 2) is to adopt the PREFIX$_{\text{multiple}}$ method, which takes $O(MN(XYT + Yn))$ time (see Theorem 2). However, PREFIX$_{\text{multiple}}$ does not provide any optimization for handling multiple bandwidths, which can lead to inferior performance in this setting. Here, we discuss how to solve this problem using the core ideas in Section IV-A. We fix a spatial bandwidth b_σ and compute STKDVs with N temporal bandwidths, i.e., computing $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ for each timestamp t_i with N sliding windows (temporal bandwidths) for all pixels $\mathbf{q}$ (see Fig. 9).

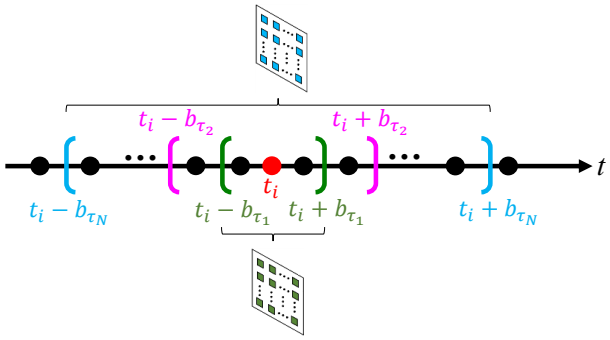


Fig. 9: Compute the density function $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ for all pixels $\mathbf{q}$ with N temporal bandwidths, b_{τ_1}, b_{τ_2},..., b_{τ_N}. Green and blue pixels denote the density values for the temporal bandwidths b_{τ_1} and b_{τ_N}, respectively.

Since each STKDV contains T timestamps, there are $2TN$ endpoints in total (see Fig. 10). Based on the core idea 3, finding the statistical terms of all prefix sets takes $O(XYTN + Yn)$ time (by setting $L = 2TN$ in Lemma 1). With these prefix sets, we can adopt the core idea 2 to obtain the statistical terms for each sliding window in $O(XY)$ time and compute the corresponding density values $\mathcal{F}_{\widehat{P}}(\mathbf{q}, t_i)$ (see Equation 4) for all pixels $\mathbf{q}$ (in $O(XY)$ time). Since there are TN sliding windows in Fig. 10, computing all density values takes $O(XYTN)$ time. Therefore, given a fixed spatial bandwidth, the time complexity for computing STKDVs with N temporal bandwidths is $O(XYTN + Yn)$.

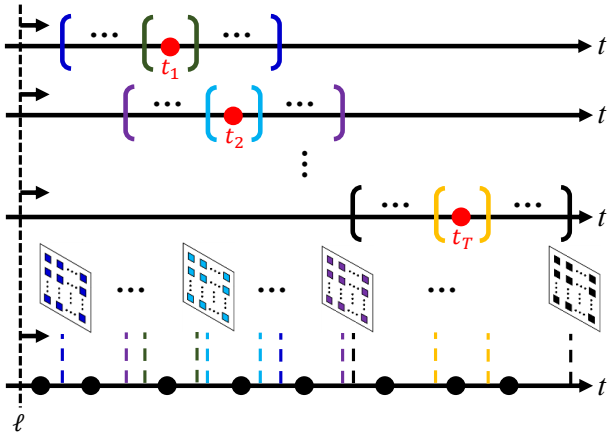


Fig. 10: Maintain the statistical terms of prefix sets for all timestamps of the endpoints of sliding windows.

Recall that there are M spatial bandwidths. Our method, named as PREFIX$_{\text{tuning}}$, can solve the bandwidth tuning problem in $O(M(XYTN + Yn))$ time (see Theorem 3), which is faster than PREFIX$_{\text{multiple}}$.

Theorem 3. *PREFIX$_{tuning}$ can solve the bandwidth tuning problem (i.e., Problem 2) in $O(M(XYTN + Yn))$ time.*

As a remark, the state-of-the-art SWS method takes $O(MNXY(T+n))$ time for solving the bandwidth tuning problem (see Section III-B). Therefore, our PREFIX$_{\text{tuning}}$ method (in $O(M(XYTN+Yn))$ time) significantly reduces the time complexity for solving this problem.

E. Space Complexity of PREFIX

We proceed to determine the space complexity of all our methods, PREFIX$_{\text{single}}$, PREFIX$_{\text{multiple}}$, and PREFIX$_{\text{tuning}}$.

PREFIX$_{\text{single}}$: Since PREFIX$_{\text{single}}$ needs to generate STKDV with a spatial resolution $X \times Y$ for a single timestamp t_i and access the data points in the sliding window $W(t_i)$, it needs at least $O(XY+n)$ space in the worst case (with $|W(t_i)| \rightarrow n$). Since PREFIX$_{\text{single}}$ is based on the simple modification of the bucket-based algorithm (see the core idea 1 in Section IV-A), SLAM$_{\text{BUCKET}}$ [27], this approach does not affect the overall space complexity (see Theorem 4 in [27]). As such, PREFIX$_{\text{single}}$ retains in $O(XY+n)$ space (see Theorem 4).

Theorem 4. *PREFIX$_{\text{single}}$ needs $O(XY+n)$ space to compute exact STKDV with a single timestamp (i.e., solving Problem 1 with $T=1$).*

PREFIX$_{\text{multiple}}$: To compute STKDV with a spatial resolution $X \times Y$ and T timestamps for a dataset $\widehat{P}$ with size n, PREFIX$_{\text{multiple}}$ needs at least $O(XYT+n)$ space for outputting the visualizations and scanning the dataset. In addition, it only needs to maintain at most $O(T)$ statistical terms for prefix sets (i.e., the pixel-planes in the lower time-axis in Fig. 8) and $O(T)$ statistical terms for the windows (i.e., the pixel-planes in the upper time-axis in Fig. 8), which consume $O(XYT)$ additional space. Therefore, the space complexity of PREFIX$_{\text{multiple}}$ remains in $O(XYT+n)$ (see Theorem 5).

Theorem 5. *PREFIX$_{\text{multiple}}$ needs $O(XYT+n)$ space to compute exact STKDV (see Problem 1) with T timestamps.*

PREFIX$_{\text{tuning}}$: Since we need to generate MN STKDVs (with M spatial bandwidths and N temporal bandwidths) for a dataset $\widehat{P}$ with size n, we need to output the visualizations (with $O(MNXYT)$ space) and scan the dataset (with $O(n)$ space). Therefore, PREFIX$_{\text{tuning}}$ needs at least $O(MNXYT+n)$ space. In addition, Fig. 10 shows that PREFIX$_{\text{tuning}}$ needs to maintain the statistical terms of prefix sets for all endpoints ($2TN$ in total), which takes $O(NXYT)$ additional space, given a fixed spatial bandwidth. Since the additional space can be cleared before we process the next spatial bandwidth, the space complexity of PREFIX$_{\text{tuning}}$ remains in $O(MNXYT+n)$ (see Theorem 6).

Theorem 6. *PREFIX$_{\text{tuning}}$ needs $O(MNXYT+n)$ space to solve the bandwidth tuning problem (i.e., Problem 2).*

Based on the above discussion, both PREFIX$_{\text{single}}$, PREFIX$_{\text{multiple}}$, and PREFIX$_{\text{tuning}}$ achieve the same space complexity compared with the state-of-the-art SWS method (see Table II) in different problem settings.

V. Experimental Evaluation

We first introduce the experimental settings in Section V-A. Then, we investigate the efficiency of all methods for computing STKDV with on-the-fly timestamps and known timestamps in Section V-B and Section V-C, respectively. Next, we conduct experiments for all methods to solve the bandwidth tuning problem in Section V-D. After that, we compare our solution with existing software packages in Section V-E. Lastly, we conduct case studies to analyze spatiotemporal hotspots on the New York taxi dataset (i.e., New York$_{\text{taxi}}$) and the Hong Kong COVID-19 dataset in Section V-F and Section V-G, respectively. Due to space limitations, some additional experiments are only included in the supplementary document [24].

A. Experimental Settings

We perform experiments on seven location datasets, as listed in Table III. These datasets belong to five categories, namely COVID-19 cases, crime events, traffic accidents, 311 calls, and pickup locations. By default, we set the spatial resolution to be 1,280 $\times$ 960 and the number of timestamps to be 32. Furthermore, we use the Scott's rule [21] to choose a default spatial bandwidth $b_\sigma^{(D)}$ and a default temporal bandwidth $b_\tau^{(D)}$ for each dataset.[2]

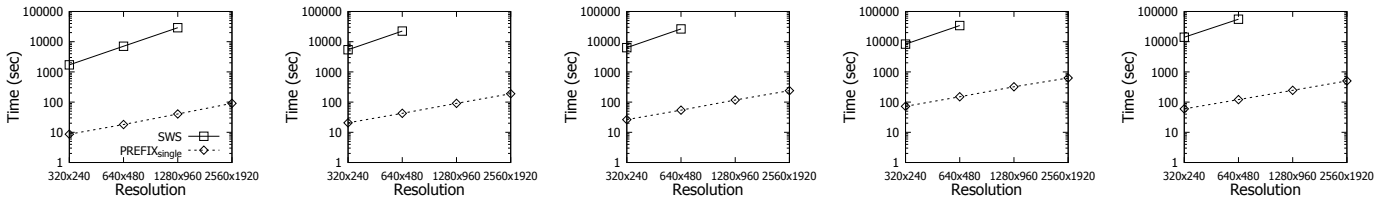
TABLE III: Datasets.

Dataset	n	Category	Ref.
Hong Kong	50,886	COVID-19 cases	[2]
Ontario	560,856	COVID-19 cases	[9]
Los Angeles	1,255,668	Crime events	[6]
New York	1,674,261	Traffic accidents	[7]
London	2,075,950	Traffic accidents	[11]
San Francisco	5,003,812	311 calls	[12]
New York$_{\text{taxi}}$	13,596,055	Pickup locations	[8]

We use five of the datasets in Table III, Ontario, Los Angeles, New York, London, and San Francisco, to study the efficiency of our methods, PREFIX$_{\text{single}}$, PREFIX$_{\text{multiple}}$, and PREFIX$_{\text{tuning}}$, and the state-of-the-art SWS method. Then, we use the two datasets, Hong Kong and New York$_{\text{taxi}}$, to conduct case studies. All methods are implemented in C++[3] and experiments are conducted on an Intel i7 3.19GHz PC with 32GB memory. In this paper, we use the response time (seconds) to measure the efficiency of each method and only report results that are within 86,400 seconds (i.e., one day). For brevity, we only show the experiment results of using the predominant Epanechnikov function (see Table I) as the spatial and temporal kernels. Note that there are also other methods, e.g., range-query-based methods (which are discussed in Section II), for computing STKDV. However, we do not include them for comparison in this paper since they have been shown to provide inferior efficiency compared with the SWS method [21] and take longer than 86,400 seconds in our experimental settings. We report on additional experiments that compare these range-query-based methods with PREFIX in the supplementary document (see Section 1 in [24]).

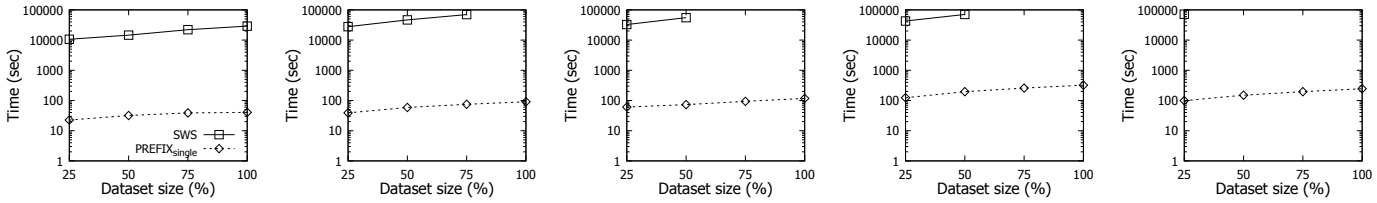
[2]For details, we can refer to https://github.com/STKDV/STKDV/blob/main/para_selection.pdf (the supplementary document of the state-of-the-art work [21]) for selecting a default spatial bandwidth and a default temporal bandwidth.

[3]The implementation of all methods can be found on Github link: https://github.com/edisonchan2013928/PREFIX.



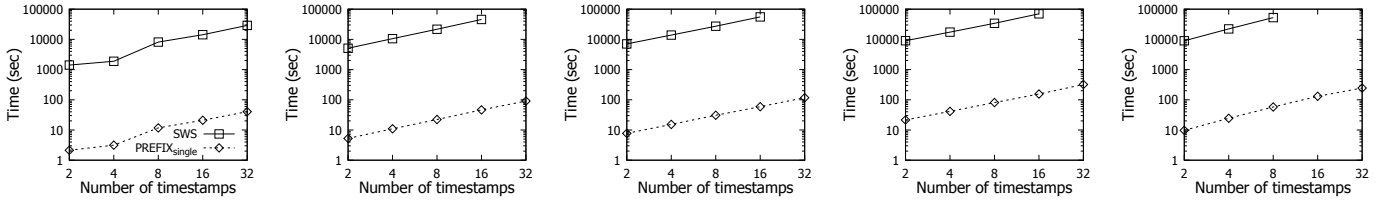
(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 11: Response time (in total) for computing STKDV with $T = 32$ on-the-fly timestamps, varying the resolution.



(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 12: Response time (in total) for computing STKDV with the default resolution 1,280 $\times$ 960 and $T = 32$ on-the-fly timestamps, varying the dataset size.



(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 13: Response time (in total) for computing STKDV with the default resolution 1,280 $\times$ 960, varying the number of (on-the-fly) timestamps T.

B. STKDV with On-the-fly Timestamps

Although PREFIX$_{\text{single}}$ has the lower time complexity for computing STKDV with on-the-fly timestamps (see Table II), we still do not know the practical improvement of PREFIX$_{\text{single}}$ compared with the state-of-the-art SWS method. Therefore, we proceed to conduct the following experiments to test the efficiency of all methods in this problem setting.

Varying the spatial resolution: We randomly generate $T = 32$ timestamps and consider the four resolutions, which are 320 $\times$ 240, 640 $\times$ 480, 1,280 $\times$ 960, and 2,560 $\times$ 1,920, for testing the efficiency of all methods. Since our method, PREFIX$_{\text{single}}$, has the lower time complexity (with $O(Y(X + n))$ time for each on-the-fly timestamp) compared with the SWS method (with $O(XYn)$ time for each on-the-fly timestamp), PREFIX$_{\text{single}}$ achieves speedups of at least 115x (see Fig. 11). In addition, the time gap between these two methods increases for each dataset when we increase the resolution.

Varying the dataset size: We proceed to investigate how the dataset size affects the response time of all methods. To conduct this experiment, we first randomly sample each dataset with four ratios, which are 25%, 50%, 75%, and 100% (no sampling), and then measure the response time of all methods on these sampled datasets. Fig. 12 shows that PREFIX$_{\text{single}}$ achieves speedups of 346x to 935x over the state-of-the-art SWS method. The main reason is that PREFIX$_{\text{single}}$ has the lower time complexity for computing STKDV with on-the-fly timestamps.

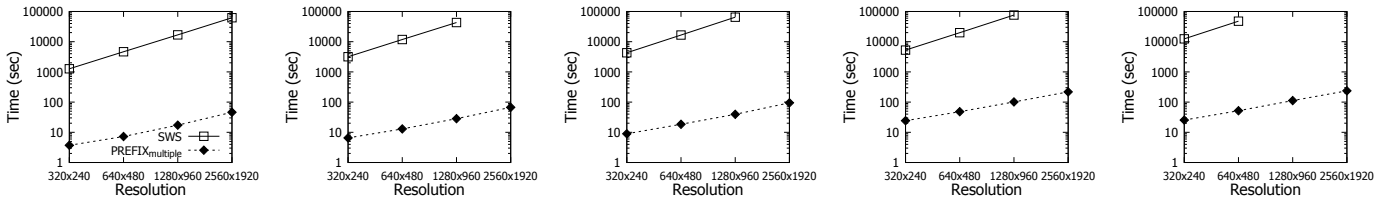
Varying the number of timestamps: We further investigate how the number of (on-the-fly) timestamps T affects the response time of all methods. Here, we first randomly generate $T = 2, 4, 8, 16, 32$ timestamps. Then, for each T, we measure the response time of all methods using those T timestamps. Due to the lower time complexity of PREFIX$_{\text{single}}$, this method consistently achieves better efficiency (with speedups of 420x to 986x) compared with the SWS method (see Fig. 13), no matter which T we use.

C. STKDV with Known Timestamps

We further investigate the time efficiency of PREFIX$_{\text{multiple}}$ and the state-of-the-art SWS method for computing STKDV with known timestamps. In this section, we discuss the results for the following experiments.

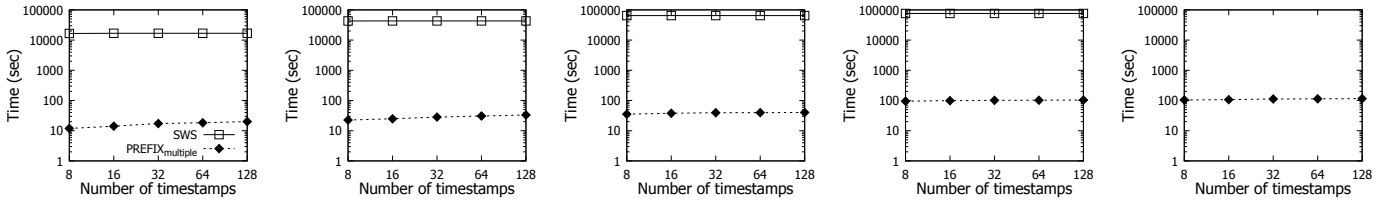
Varying the spatial resolution: In this experiment, we fix the number of timestamps to be 32 and vary the resolution from 320 $\times$ 240 to 2,560 $\times$ 1,920 for testing the response time of PREFIX$_{\text{multiple}}$ and SWS. Since PREFIX$_{\text{multiple}}$ can reduce the time complexity for computing STKDV (with known timestamps), our method achieves speedups of 217x to 1,524x over the state-of-the-art SWS method (see Fig. 14).

Varying the number of timestamps: Here, we further examine how the number of timestamps T affects the response time of PREFIX$_{\text{multiple}}$ and SWS. Since the time complexity of PREFIX$_{\text{multiple}}$ and SWS is $O(XYT + Yn)$ and $O(XY(T + n))$, respectively, and the number of data points n is much larger compared with the number of timestamps T, both



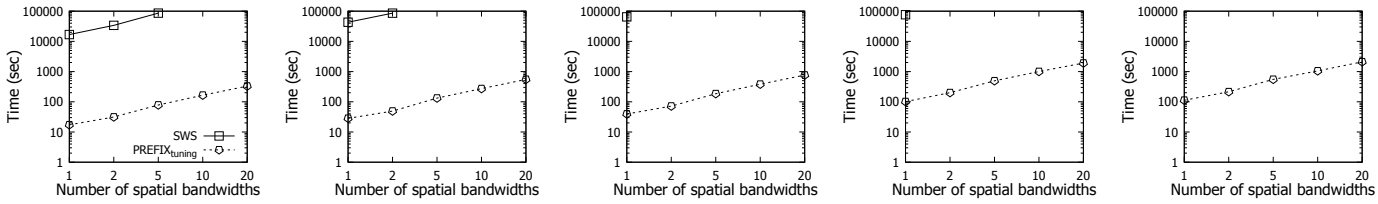
(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 14: Response time for computing STKDV with $T = 32$ known timestamps, varying the resolution.



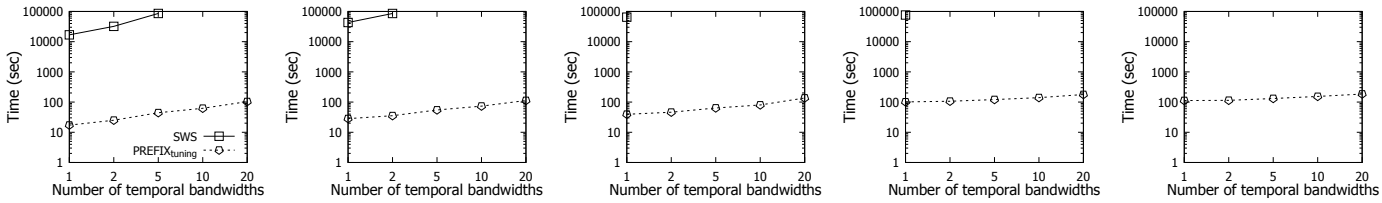
(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 15: Response time for computing STKDV with the default resolution 1,280 $\times$ 960, varying the number of (known) timestamps T.



(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 16: Response time for computing STKDVs with the default resolution 1,280 $\times$ 960 and the default number of timestamps $T = 32$, varying the number of spatial bandwidths M.



(a) Ontario (b) Los Angeles (c) New York (d) London (e) San Francisco

Fig. 17: Response time for computing STKDVs with the default resolution 1,280 $\times$ 960 and the default number of timestamps $T = 32$, varying the number of temporal bandwidths N.

methods are insensitive to this parameter T (see Fig. 15). With the lower time complexity of PREFIX$_{\text{multiple}}$, this method achieves speedups of 737x to 1,906x.

D. Bandwidth Tuning

Recall that domain experts need to tune the spatial and temporal bandwidths in order to obtain the best visual quality of STKDV. Therefore, we proceed to compare the efficiency of PREFIX$_{\text{tuning}}$ and the state-of-the-art SWS method in this problem setting.

Varying the number of spatial bandwidths: Here, we investigate how the number of spatial bandwidths M affects the efficiency of the PREFIX$_{\text{tuning}}$ and SWS methods. To conduct this experiment, we first adopt the default resolution 1,280 $\times$ 960, adopt the default number of timestamps $T = 32$, and fix the number of temporal bandwidths N to be 1 (use the default temporal bandwidth $b_\tau^{(D)}$). Then, we divide the range $[0, 2b_\sigma^{(D)}]$ equally and choose M spatial bandwidths, i.e., $\frac{2b_\sigma^{(D)}}{M}$, $\frac{4b_\sigma^{(D)}}{M}$,..., $2b_\sigma^{(D)}$, where we vary the number M from 1 to 20 for testing. Since the time complexity of our PREFIX$_{\text{tuning}}$ method and the SWS method (see Table II) is linearly proportional to the number of spatial bandwidths M, the response time increases linearly for these two methods (see Fig. 16). Benefiting from the lower time complexity of PREFIX$_{\text{tuning}}$, this method achieves nearly three-order-of-magnitude speedups over the state-of-the-art SWS method.

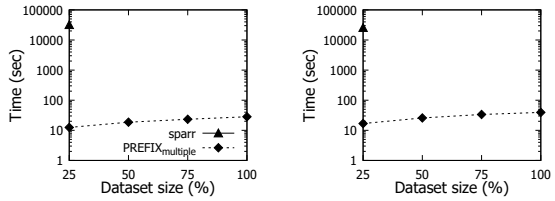
Varying the number of temporal bandwidths: We investigate how the number of temporal bandwidths N affects the efficiency of the PREFIX$_{\text{tuning}}$ and SWS methods. We follow the same settings as in the previous experiment, except that we fix the number of spatial bandwidths M to be 1 (use the default spatial bandwidth $b_\sigma^{(D)}$) and vary the number N from 1 to 20 for testing, where the N bandwidths are $\frac{2b_\tau^{(D)}}{N}$, $\frac{4b_\tau^{(D)}}{N}$,..., $2b_\tau^{(D)}$. Fig. 17 shows that PREFIX$_{\text{tuning}}$ achieves

at least 754x speedups over the SWS method. Furthermore, since $\text{PREFIX}_{\text{tuning}}$ takes $O(M(XYTN+Yn))$ time and n is normally much larger than the other parameters, this method is insensitive to the number of temporal bandwidths N.

E. Efficiency Comparisons with Software Packages

Due to the popularity of STKDV, various software packages have been recently developed for supporting this tool, including LIBKDV [23] (a python library), Fast Density Analysis [4] (a QGIS plugin), and sparr [14], [36] (an R package). Among these packages, we only compare our solution with sparr. The main reason is that LIBKDV and Fast Density Analysis are built on top of the state-of-the-art SWS method, which is evaluated extensively in Section V-B to Section V-D. As a remark, some efficiency results that are related to the parallelization of LIBKDV are included in the supplementary document (see Section 3 in [24]).

In this experiment, we adopt two datasets, Los Angeles and New York, for testing. We measure the response time of $\text{PREFIX}_{\text{multiple}}$ and sparr for generating STKDV with T known timestamps by (1) following the default settings in Section V-C and (2) varying the dataset size with four sampling ratios, which are 25%, 50%, 75%, and 100% (no sampling).



(a) Los Angeles (b) New York

Fig. 18: Response time for computing STKDV with the default resolution 1,280 $\times$ 960 and the default number of known timestamps $T = 32$, varying the dataset size.

Fig. 18 shows the results of sparr and $\text{PREFIX}_{\text{multiple}}$. Since $\text{PREFIX}_{\text{multiple}}$ can achieve low time complexity for generating STKDV with known timestamps (see Table II), these methods achieve speedups of 1,522.04x to 2,556.57x over sparr. Moreover, with the sampling ratios of 50% or above for these two datasets, sparr consumes more than 32GB memory space, causing a memory overflow error. Therefore, we also conclude that sparr suffers from excessive memory space consumption compared with $\text{PREFIX}_{\text{multiple}}$. Since sparr cannot support STKDV with on-the-fly timestamps and always takes more than 86,400 seconds for supporting bandwidth tuning, we do not compare sparr with $\text{PREFIX}_{\text{single}}$ and $\text{PREFIX}_{\text{tuning}}$, respectively.

F. Case Study: STKDV-based Hotspot Analysis on the New York_{taxi} Dataset

Since PREFIX is more scalable to large-scale datasets, we adopt this approach to analyze traffic hotspots of the largest $\text{New York}_{\text{taxi}}$ dataset (see Table III), which, to the best of our knowledge, cannot be feasibly supported by existing solutions. To conduct this case study, we choose the spatial resolution to be 1,280 $\times$ 960, set the number of timestamps T to be 32, and specify multiple spatial bandwidths, including 300, 350, 400, 450, and 500 meters, and multiple temporal bandwidths, including 0.1, 0.2, 0.3, 0.4, and 0.5 days, for tuning the best visual quality of STKDV (see Problem 2). *In this setting, generating these 25 STKDVs only takes 1,383 seconds ($\sim$23 minutes), while the state-of-the-art SWS method takes more than 86,400 seconds (i.e., one day) to generate even a single STKDV.*

For brevity, Fig. 19 only shows the STKDV results for three temporal bandwidths, i.e., 0.1 days, 0.3 days, and 0.5 days, with the fixed spatial bandwidth, i.e., 350 meters, and five timestamps. Note that the traffic hotspots are mainly in the Manhattan region. Furthermore, suppose that we choose 0.1 days as the temporal bandwidth, we cannot detect any hotspots (see the first row of Fig. 19). Once we increase the temporal bandwidth (e.g., 0.3 days and 0.5 days), we can discover meaningful hotspots in this region. In addition, we observe that different temporal bandwidths can be suitable for different timestamps. For example, using 0.5 days as the temporal bandwidth on 7/1/2014, 19/1/2014, and 24/1/2014 enables discovery of more hidden patterns (red regions). while it causes oversmoothing of hotspot maps (a large red region) on 17/1/2014 and 23/1/2014. As such, our fast solution enables domain experts to inspect more spatial and temporal bandwidths to obtain the best visualization results.

G. Case Study: STKDV-based Hotspot Analysis on the Hong Kong COVID-19 Dataset

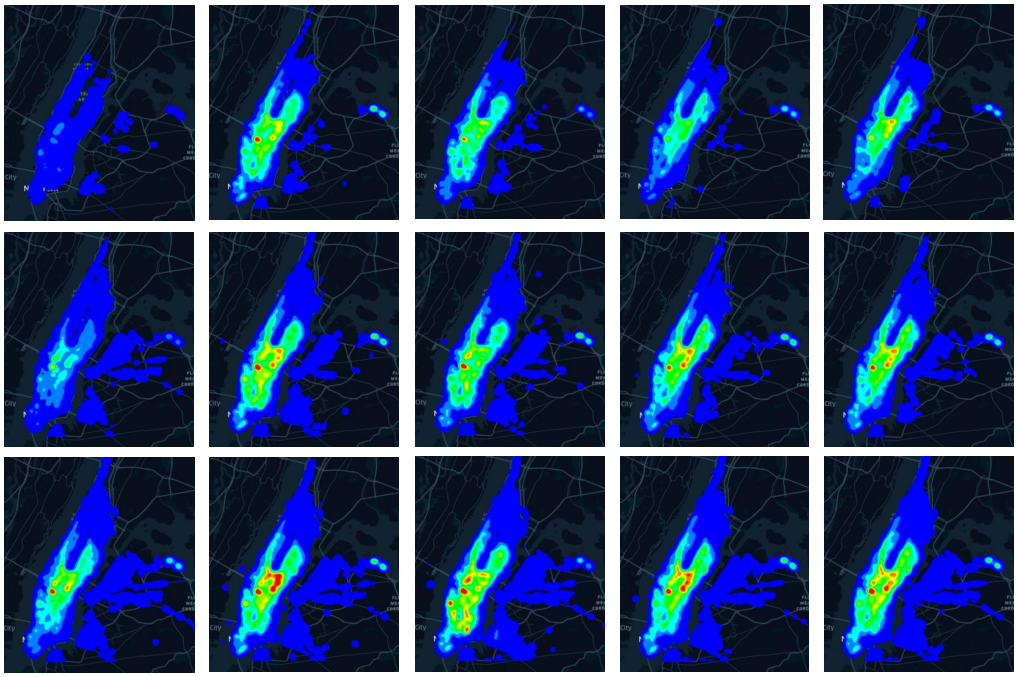
We proceed to investigate STKDV-based hotspots on the Hong Kong COVID-19 dataset by performing exploratory operations (e.g., selecting different timestamps on the fly and performing zooming and panning operations). Observe from the 1st row of the hotspot maps in Fig. 20 that the hotspot shape can change across timestamps. Once we zoom in and pan to the northwestern part of Hong Kong (the 2nd row in Fig. 20), additional hidden patterns can be detected in this region.

To support exploratory operations smoothly, a primary goal is to achieve real-time performance (< 0.5 seconds) for generating a single visualization. However, since SWS suffers from high time complexity for generating STKDV with an on-the-fly timestamp compared with PREFIX, this method takes roughly 20 seconds to generate a hotspot map in the case study. In contrast, PREFIX can support exploratory operations in real-time.[4]

VI. Conclusion

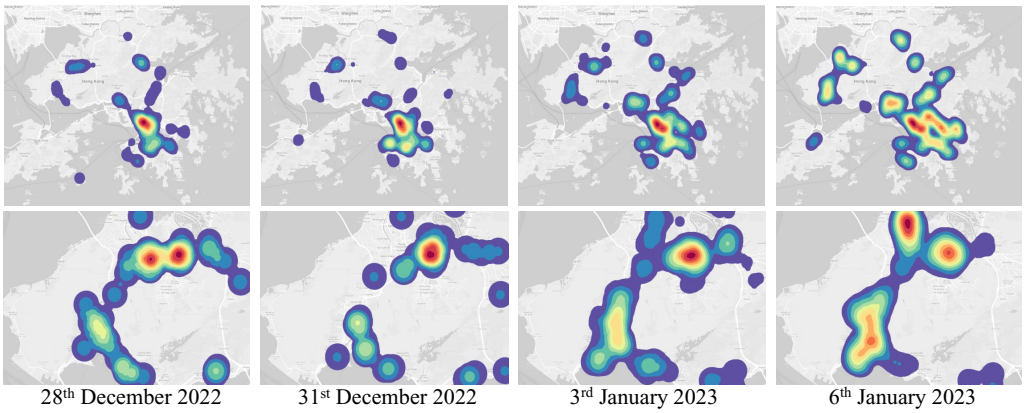
In this paper, we study spatiotemporal kernel density visualization (STKDV), which is used extensively in many applications, including traffic accident hotspot detection, crime hotspot detection, and disease outbreak detection. However, STKDV is very time-consuming and does not scale to large datasets, high resolution sizes, and a large number of timestamps. Although Chan et al. [21] have proposed a preliminary

[4]We provide the video links, which are https://www.youtube.com/watch?v=SjWcjHJd1as (Youtube) and https://www.bilibili.com/video/BV171mTYGETV (Bilibili), for illustrating this case study with SWS and PREFIX.



(a) 7/1/2014 (Tue) (b) 17/1/2014 (Fri) (c) 19/1/2014 (Sun) (d) 23/1/2014 (Thu) (e) 24/1/2014 (Fri)

Fig. 19: Generation of STKDVs for three temporal bandwidths, i.e., 0.1 days (1st row), 0.3 days (2nd row), and 0.5 days (3rd row), where we fix the spatial bandwidth to be 350 meters, choose the resolution size to be 1,280 $\times$ 960, and show five timestamps.



28th December 2022 31st December 2022 3rd January 2023 6th January 2023

Fig. 20: Generation of STKDVs with four on-the-fly timestamps on the Hong Kong COVID-19 dataset, where the hotspot maps in the 1st row and the 2nd row cover the whole region and the northwestern region of Hong Kong, respectively.

work, called SWS, that can reduce the time complexity for computing STKDV, this solution (i) is unable to reduce the time complexity for supporting exploratory analysis, (ii) still takes $O(XY(T+n))$ time, and (iii) does not provide any optimization techniques for bandwidth tuning. To address these issues, we propose the prefix-set-based solution, namely PREFIX, which consists of three methods, $\text{PREFIX}_{\text{single}}$ for handling (i), $\text{PREFIX}_{\text{multiple}}$ for handling (ii), and $\text{PREFIX}_{\text{tuning}}$ for handling (iii). Our theoretical analysis (see Table II) shows that all these methods can significantly reduce the time complexity, without increasing the space complexity, in different problem settings. Furthermore, our experimental results show that PREFIX (1) achieves speedups in the range 115x to 1,906x compared with the state-of-the-art SWS method for all problem settings and (2) is the first solution that scales to compute multiple high-resolution STKDVs (1,280 $\times$ 960) for the $\text{New York}_{\text{taxi}}$ dataset (with 13.6 million data points).

In the future, we plan to extend this solution to support other types of kernel functions, including Gaussian kernel and exponential kernel, and other geospatial analysis tasks, e.g., network-temporal kernel density visualization (NTKDV) [41], [72], [74] and network K-function [29], [30]. Furthermore, we will investigate advanced parallel/distributed/hardware-based methods (e.g., GPUs) to further improve the efficiency of our solution, since PREFIX still cannot achieve real-time performance ($<$ 0.5 seconds) for generating STKDV or supporting bandwidth tuning (see Section V) on million-scale datasets (see Table III). Moreover, we also plan to investigate the efficiency and accuracy issues of other types of data visualization tasks (e.g., [34], [60], [69]).

REFERENCES

[1] "ArcGIS," http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/how-kernel-density-works.htm.

[2] "Common Spatial Data Infrastructure (CSDI) Portal for Government," https://portal.csdi.gov.hk/csdi-webpage/.

[3] "Deck.gl," https://deck.gl/docs/api-reference/aggregation-layers/heatmap-layer.

[4] "Fast Density Analysis," https://plugins.qgis.org/plugins/fast_kernel_density_analysis/.

[5] "Leaflet - a javascript library for interactive maps," https://leafletjs.com/.

[6] "Los Angeles open data," https://data.lacity.org/A-Safe-City/Crime-Data-from-2010-to-2019/63jg-8b9z.

[7] "NYC open data," https://data.cityofnewyork.us/Public-Safety/Motor-Vehicle-Collisions-Crashes/h9gi-nx95.

[8] "NYC yellow taxi trip data," https://data.cityofnewyork.us/Transportation/2014-Yellow-Taxi-Trip-Data/gkne-dk5s.

[9] "Ontario open data," https://data.ontario.ca/dataset/confirmed-positive-cases-of-covid-19-in-ontario.

[10] "QGIS," https://docs.qgis.org/2.18/en/docs/user_manual/plugins/plugins_heatmap.html.

[11] "Road safety data," https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data.

[12] "San Francisco open data," https://data.sfgov.org/City-Infrastructure/311-Cases/vw6y-z8j6.

[13] "Seaborn," https://seaborn.pydata.org/generated/seaborn.kdeplot.html.

[14] "sparr: Spatial and spatiotemporal relative risk," https://cran.r-project.org/web/packages/sparr/index.html.

[15] R. Ben-Basat, R. Friedman, and R. Shahout, "Stream frequency over interval queries," *Proc. VLDB Endow.*, vol. 12, no. 4, pp. 433–445, 2018. [Online]. Available: http://www.vldb.org/pvldb/vol12/p433-basat.pdf.

[16] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[17] C. Brunsdon, J. Corcoran, and G. Higgs, "Visualising space and time in crime patterns: A comparison of methods," *Comput. Environ. Urban Syst.*, vol. 31, no. 1, pp. 52–75, 2007. [Online]. Available: https://doi.org/10.1016/j.compenvurbsys.2005.07.009.

[18] M. Bíl, R. Andrášik, and J. Sedoník, "A detailed spatiotemporal analysis of traffic crash hotspots," *Applied Geography*, vol. 107, pp. 82–90, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0143622818309081.

[19] T. N. Chan, R. Cheng, and M. L. Yiu, "QUAD: Quadratic-bound-based kernel density visualization," in *SIGMOD*, 2020, pp. 35–50. [Online]. Available: https://doi.org/10.1145/3318464.3380561

[20] T. N. Chan, P. L. Ip, L. H. U, B. Choi, and J. Xu, "SAFE: A share-and-aggregate bandwidth exploration framework for kernel density visualization," *Proc. VLDB Endow.*, vol. 15, no. 3, pp. 513–526, 2022.

[21] ——, "SWS: A complexity-optimized solution for spatial-temporal kernel density visualization," *Proc. VLDB Endow.*, vol. 15, no. 4, pp. 814–827, 2022. [Online]. Available: https://www.vldb.org/pvldb/vol15/p814-chan.pdf.

[22] T. N. Chan, P. L. Ip, L. H. U, W. H. Tong, S. Mittal, Y. Li, and R. Cheng, "KDV-Explorer: A near real-time kernel density visualization system for spatial analysis," *Proc. VLDB Endow.*, vol. 14, no. 12, pp. 2655–2658, 2021.

[23] T. N. Chan, P. L. Ip, K. Zhao, L. H. U, B. Choi, and J. Xu, "LIBKDV: A versatile kernel density visualization library for geospatial analytics," *Proc. VLDB Endow.*, vol. 15, no. 12, pp. 3606–3609, 2022. [Online]. Available: https://www.vldb.org/pvldb/vol15/p3606-chan.pdf.

[24] T. N. Chan, P. L. Ip, B. Zhu, L. H. U, D. Wu, J. Xu, and C. S. Jensen, "Supplementary document for "large-scale spatiotemporal kernel density visualization"," https://github.com/edisonchan2013928/PREFIX-SD/blob/main/PREFIX_supplementary.pdf.

[25] T. N. Chan, Z. Li, L. H. U, J. Xu, and R. Cheng, "Fast augmentation algorithms for network kernel density visualization," *Proc. VLDB Endow.*, vol. 14, no. 9, pp. 1503–1516, 2021. [Online]. Available: http://www.vldb.org/pvldb/vol14/p1503-chan.pdf.

[26] T. N. Chan, L. H. U, R. Cheng, M. L. Yiu, and S. Mittal, "Efficient algorithms for kernel aggregation queries," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 6, pp. 2726–2739, 2022. [Online]. Available: https://doi.org/10.1109/TKDE.2020.3018376

[27] T. N. Chan, L. H. U, B. Choi, and J. Xu, "SLAM: Efficient sweep line algorithms for kernel density visualization," in *SIGMOD*, 2022, pp. 2120–2134. [Online]. Available: https://doi.org/10.1145/3514221.3517823.

[28] T. N. Chan, L. H. U, B. Choi, J. Xu, and R. Cheng, "Kernel density visualization for big geospatial data: Algorithms and applications," in *MDM*, 2023, pp. 231–234. [Online]. Available: https://doi.org/10.1109/MDM58254.2023.00046

[29] ——, "Large-scale geospatial analytics: Problems, challenges, and opportunities," in *SIGMOD Companion*, 2023, pp. 21–29. [Online]. Available: https://doi.org/10.1145/3555041.3589401

[30] T. N. Chan, L. H. U, Y. Peng, B. Choi, and J. Xu, "Fast network k-function-based spatial analysis," *Proc. VLDB Endow.*, vol. 15, no. 11, pp. 2853–2866, 2022. [Online]. Available: https://www.vldb.org/pvldb/vol15/p2853-chan.pdf

[31] T. N. Chan, M. L. Yiu, and L. H. U, "KARL: Fast kernel aggregation queries," in *ICDE*, 2019, pp. 542–553. [Online]. Available: https://doi.org/10.1109/ICDE.2019.00055

[32] T. N. Chan, R. Zang, P. L. Ip, L. H. U, and J. Xu, "PyNKDV: An efficient network kernel density visualization library for geospatial analytic systems," in *SIGMOD Companion*, 2023, pp. 99–102. [Online]. Available: https://doi.org/10.1145/3555041.3589711

[33] T. N. Chan, R. Zang, B. Zhu, L. H. U, D. Wu, and J. Xu, "LION: Fast and high-resolution network kernel density visualization," *Proc. VLDB Endow.*, vol. 17, no. 6, pp. 1255–1268, 2024. [Online]. Available: https://www.vldb.org/pvldb/vol17/p1255-chan.pdf.

[34] T. N. Chan, B. Zhu, D. Wu, Y. Peng, and L. H. U, "LARGE: A length-aggregation-based grid structure for line density visualization," *Proc. VLDB Endow.*, vol. 17, no. 13, pp. 4585–4598, 2024.

[35] M. Crimmins, S. Park, V. Smith, and P. Kremer, "A spatial assessment of high-resolution drainage characteristics and roadway safety during wet conditions," *Applied Geography*, vol. 133, p. 102477, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S014362282100093X.

[36] T. M. Davies, J. C. Marshall, and M. L. Hazelton, "Tutorial on kernel estimation of continuous spatial and spatiotemporal relative risk," *Statistics in Medicine*, vol. 37, no. 7, pp. 1191–1221, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.7577.

[37] E. Delmelle, C. Dony, I. Casas, M. Jia, and W. Tang, "Visualizing the impact of space-time uncertainties on dengue fever patterns," *International Journal of Geographical Information Science*, vol. 28, no. 5, pp. 1107–1127, 2014. [Online]. Available: https://doi.org/10.1080/13658816.2013.871285.

[38] E. Delmelle, M. Jia, C. Dony, I. Casas, and W. Tang, "Space-time visualization of dengue fever outbreaks," in *Spatial analysis in health geography*. Routledge, 2016, pp. 85–100.

[39] E. Gan and P. Bailis, "Scalable kernel density classification via threshold-based pruning," in *SIGMOD*, 2017, pp. 945–959.

[40] S. Gao, "Spatio-temporal analytics for exploring human mobility patterns and urban dynamics in the mobile age," *Spatial Cognition & Computation*, vol. 15, no. 2, pp. 86–114, 2015.

[41] J. Gelb and P. Apparicio, "Temporal network kernel density estimation," *Geographical Analysis*, vol. 56, no. 1, pp. 62–78, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/gean.12368.

[42] T. M. Ghanem, M. A. Hammad, M. F. Mokbel, W. G. Aref, and A. K. Elmagarmid, "Incremental evaluation of sliding-window queries over data streams," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 1, pp. 57–72, 2007. [Online]. Available: https://doi.org/10.1109/TKDE.2007.250585.

[43] S. Ghezelbash, R. Ghezelbash, and M. Kalantari, "Developing a spatio-temporal interactions model for car crashes using a novel data-driven AHP-TOPSIS," *Applied Geography*, vol. 162, p. 103151, 2024.

[44] X. Gou, L. He, Y. Zhang, K. Wang, X. Liu, T. Yang, Y. Wang, and B. Cui, "Sliding sketches: A framework using time zones for data stream processing in sliding windows," in *SIGKDD*, 2020, pp. 1015–1025. [Online]. Available: https://doi.org/10.1145/3394486.3403144.

[45] X. Gou, Y. Zhang, Z. Hu, L. He, K. Wang, X. Liu, T. Yang, Y. Wang, and B. Cui, "A sketch framework for approximate data stream processing in sliding windows," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4411–4424, 2023. [Online]. Available: https://doi.org/10.1109/TKDE.2022.3151140.

[46] A. G. Gray and A. W. Moore, "Nonparametric density estimation: Toward computational tractability," in *SDM*, 2003, pp. 203–211.

[47] J. Heer, "Fast & accurate gaussian kernel density estimation," in *VIS*, 2021, pp. 11–15. [Online]. Available: https://doi.org/10.1109/VIS49827.2021.9623323.

[48] G. R. Hjaltason and H. Samet, "Index-driven similarity search in metric spaces," *ACM Trans. Database Syst.*, vol. 28, no. 4, pp. 517–580, 2003. [Online]. Available: https://doi.org/10.1145/958942.958948.

[49] A. Hohl, E. Delmelle, W. Tang, and I. Casas, "Accelerating the discovery of space-time patterns of infectious diseases using parallel computing," *Spatial and Spatio-temporal Epidemiology*, vol. 19, pp. 10 – 20, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S187758451530040X.

[50] Y. Hu, F. Wang, C. Guin, and H. Zhu, "A spatio-temporal kernel density estimation framework for predictive crime hotspot mapping and evaluation," *Applied Geography*, vol. 99, pp. 89 – 97, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0143622818300560.

[51] J. Huang and M.-P. Kwan, "Uncertainties in the assessment of COVID-19 risk: A study of people's exposure to high-risk environments using individual-level activity data," *Annals of the American Association of Geographers*, vol. 112, no. 4, pp. 968–987, 2022.

[52] ——, "Associations between COVID-19 risk, multiple environmental exposures, and housing conditions: A study using individual-level GPS-based real-time sensing data," *Applied Geography*, vol. 153, p. 102904, 2023.

[53] Z. Kan, M.-P. Kwan, J. Huang, J. Cai, and D. Liu, "A spatial network-based assessment of individual exposure to COVID-19," *Annals of the American Association of Geographers*, pp. 1–11, 2023.

[54] Y. Lan and E. Delmelle, "Space-time cluster detection techniques for infectious diseases: A systematic review," *Spatial and Spatio-temporal Epidemiology*, vol. 44, p. 100563, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877584522000867.

[55] J. Lee, *Spatiotemporal Analytics*. CRC Press, 2023.

[56] J. Lee, J. Gong, and S. Li, "Exploring spatiotemporal clusters based on extended kernel estimation methods," *Int. J. Geogr. Inf. Sci.*, vol. 31, no. 6, pp. 1154–1177, 2017. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/13658816.2017.1287371.

[57] J. Li, D. Maier, K. Tufte, V. Papadimos, and P. A. Tucker, "Semantics and evaluation techniques for window aggregates in data streams," in *SIGMOD*, 2005, pp. 311–322. [Online]. Available: https://doi.org/10.1145/1066157.1066193.

[58] Y. Li, M. Abdel-Aty, J. Yuan, Z. Cheng, and J. Lu, "Analyzing traffic violation behavior at urban intersections: A spatio-temporal kernel density estimation approach using automated enforcement system data," *Accident Analysis & Prevention*, vol. 141, p. 105509, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000145751931485X.

[59] X. Lin, Y. Yuan, W. Wang, and H. Lu, "Stabbing the sky: Efficient skyline computation over sliding windows," in *ICDE*, 2005, pp. 502–513. [Online]. Available: https://doi.org/10.1109/ICDE.2005.137.

[60] Y. Luo, X. Qin, N. Tang, and G. Li, "Deepeye: Towards automatic data visualization," in *ICDE*, 2018, pp. 101–112. [Online]. Available: https://doi.org/10.1109/ICDE.2018.00019

[61] R. Maciejewski, R. Hafen, S. Rudolph, S. G. Larew, M. A. Mitchell, W. S. Cleveland, and D. S. Ebert, "Forecasting hotspots - A predictive analytics approach," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 4, pp. 440–453, 2011. [Online]. Available: https://doi.org/10.1109/TVCG.2010.82.

[62] R. Maciejewski, S. Rudolph, R. Hafen, A. M. Abusalah, M. Yakout, M. Ouzzani, W. S. Cleveland, S. J. Grannis, and D. S. Ebert, "A visual analytics approach to understanding spatiotemporal hotspots," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 205–220, 2010. [Online]. Available: https://doi.org/10.1109/TVCG.2009.100.

[63] R. Maciejewski, S. Rudolph, R. Hafen, A. M. Abusalah, M. Yakout, M. Ouzzani, W. S. Cleveland, S. J. Grannis, M. Wade, and D. S. Ebert, "Understanding syndromic hotspots - a visual analytics approach," in *VAST*, 2008, pp. 35–42. [Online]. Available: https://doi.org/10.1109/VAST.2008.4677354.

[64] A. W. Moore, "The anchors hierarchy: Using the triangle inequality to survive high dimensional data," in *UAI*, 2000, pp. 397–405.

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. VanderPlas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[66] J. M. Phillips, "ϵ-samples for kernels," in *SODA*, 2013, pp. 1622–1632. [Online]. Available: https://doi.org/10.1137/1.9781611973105.116.

[67] J. M. Phillips and W. M. Tai, "Improved coresets for kernel density estimates," in *SODA*, 2018, pp. 2718–2727. [Online]. Available: https://doi.org/10.1137/1.9781611975031.173.

[68] ——, "Near-optimal coresets of kernel density estimates," in *SOCG*, 2018, pp. 66:1–66:13. [Online]. Available: https://doi.org/10.4230/LIPIcs.SoCG.2018.66.

[69] X. Qin, Y. Luo, N. Tang, and G. Li, "Making data visualization more efficient and effective: a survey," *VLDB J.*, vol. 29, no. 1, pp. 93–117, 2020. [Online]. Available: https://doi.org/10.1007/s00778-019-00588-3

[70] S. Rakshit, A. Baddeley, and G. Nair, "Efficient code for second order analysis of events on a linear network," *Journal of Statistical Software*, vol. 90, no. 1, pp. 1–37, 2019. [Online]. Available: https://www.jstatsoft.org/v090/i01.

[71] V. C. Raykar, R. Duraiswami, and L. H. Zhao, "Fast computation of kernel estimators," *Journal of Computational and Graphical Statistics*, vol. 19, no. 1, pp. 205–220, 2010.

[72] B. Romano and Z. Jiang, "Visualizing traffic accident hotspots based on spatial-temporal network kernel density estimation," in *SIGSPATIAL*, 2017, pp. 98:1–98:4. [Online]. Available: https://doi.org/10.1145/3139958.3139981.

[73] A. G. F. Rosa, C. M. de Miranda Mota, and C. J. J. de Figueiredo, "A spatial multi-criteria decision analysis framework to reveal vulnerabilities of areas to incidences of street robberies," *Applied Geography*, vol. 151, p. 102840, 2023.

[74] G. Rosser, T. O. Davies, K. Bowers, S. D. Johnson, and T. Cheng, "Predictive crime mapping: Arbitrary grids or street networks?" *Journal of Quantitative Criminology*, vol. 33, pp. 569 – 594, 2017.

[75] H. Samet, *Foundations of Multidimensional and Metric Data Structures*, ser. Morgan Kaufmann, 2006.

[76] ——, "Hierarchical representations of collections of small rectangles," *ACM Comput. Surv.*, vol. 20, no. 4, pp. 271–309, 1988. [Online]. Available: https://doi.org/10.1145/50020.50021.

[77] E. Saule, D. Panchananam, A. Hohl, W. Tang, and E. Delmelle, "Parallel space-time kernel density estimation," in *ICPP*, 2017, pp. 483–492. [Online]. Available: https://doi.org/10.1109/ICPP.2017.57.

[78] D. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley, 1992.

[79] K. Tangwongsan, M. Hirzel, and S. Schneider, "Optimal and general out-of-order sliding-window aggregation," *PVLDB*, vol. 12, no. 10, pp. 1167–1180, 2019. [Online]. Available: http://www.vldb.org/pvldb/vol12/p1167-tangwongsan.pdf.

[80] ——, "In-order sliding-window aggregation in worst-case constant time," *VLDB J.*, vol. 30, no. 6, pp. 933–957, 2021. [Online]. Available: https://doi.org/10.1007/s00778-021-00668-3.

[81] K. Tangwongsan, M. Hirzel, S. Schneider, and K. Wu, "General incremental sliding-window aggregation," *PVLDB*, vol. 8, no. 7, pp. 702–713, 2015. [Online]. Available: http://www.vldb.org/pvldb/vol8/p702-tangwongsan.pdf.

[82] Y. Tao and D. Papadias, "Maintaining sliding window skylines on data streams," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 2, pp. 377–391, 2006. [Online]. Available: https://doi.org/10.1109/TKDE.2006.48.

[83] A. Tapsoba, "The cost of fear: Impact of violence risk on child health during conflict," *Journal of Development Economics*, vol. 160, p. 102975, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304387822001171.

[84] Á. Villalba, J. L. Berral, and D. Carrera, "Constant-time sliding window framework with reduced memory footprint and efficient bulk evictions," *IEEE Trans. Parallel Distributed Syst.*, vol. 30, no. 3, pp. 486–500, 2019. [Online]. Available: https://doi.org/10.1109/TPDS.2018.2868960.

[85] P. Virtanen, R. Gommers, and T. E. e. a. Oliphant, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[86] X. Wang, Y. Zhang, W. Zhang, X. Lin, and Z. Huang, "SKYPE: top-k spatial-keyword publish/subscribe over sliding window," *PVLDB*, vol. 9, no. 7, pp. 588–599, 2016. [Online]. Available: http://www.vldb.org/pvldb/vol9/p588-wang.pdf.

[87] Q. Wei, J. She, S. Zhang, and J. Ma, "Using individual GPS trajectories to explore foodscape exposure: A case study in Beijing metropolitan

area," *International journal of environmental research and public health*, vol. 15, 02 2018.

[88] W. Wu, P. A. Bernstein, A. Raizman, and C. Pavlopoulou, "Factor windows: Cost-based query rewriting for optimizing correlated window aggregates," in *ICDE*, 2022, pp. 2722–2734. [Online]. Available: https://doi.org/10.1109/ICDE53745.2022.00249.

[89] K. Xie, K. Ozbay, A. Kurkcu, and H. Yang, "Analysis of traffic crashes involving pedestrians using big data: Investigation of contributing factors and identification of hotspots," *Risk Analysis*, vol. 37, no. 8, pp. 1459–1476, 2017. [Online]. Available: https://EconPapers.repec.org/RePEc:wly:riskan:v:37:y:2017:i:8:p:1459-1476.

[90] L. Xu, M.-P. Kwan, S. McLafferty, and S. Wang, "Predicting demand for 311 non-emergency municipal services: An adaptive space-time kernel approach," *Applied Geography*, vol. 89, pp. 133–141, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0143622817304538.

[91] P. Zezula, G. Amato, V. Dohnal, and M. Batko, *Similarity Search: The Metric Space Approach*, ser. Advances in Database Systems. Springer US, 2006. [Online]. Available: https://books.google.com.hk/books?id=KTkWXsiPXR4C.

[92] C. Zhang, R. Akbarinia, and F. Toumani, "Efficient incremental computation of aggregations over sliding windows," in *SIGKDD*, 2021, pp. 2136–2144. [Online]. Available: https://doi.org/10.1145/3447548.3467360.

[93] H. Zhang, Y. Gao, D. Yao, and J. Zhang, "Interaction of crime risk across crime types in hotspot areas," *ISPRS International Journal of Geo-Information*, vol. 12, no. 4, 2023.

[94] W. Zhang, X. Lin, Y. Zhang, W. Wang, and J. X. Yu, "Probabilistic skyline operator over sliding windows," in *ICDE*, 2009, pp. 1060–1071. [Online]. Available: https://doi.org/10.1109/ICDE.2009.83.

[95] Z. Zhang, D. Chen, W. Liu, J. Racine, S.-H. Ong, Y. Chen, G. Zhao, and Q. Jiang, "Nonparametric evaluation of dynamic disease risk: A spatio-temporal kernel approach," *PloS one*, vol. 6, p. e17381, 03 2011.

[96] X. Zhao and J. Tang, "Crime in urban areas: A data mining perspective," *SIGKDD Explorations*, vol. 20, no. 1, pp. 1–12, 2018. [Online]. Available: https://doi.org/10.1145/3229329.3229331.

[97] Y. Zheng, J. Jestes, J. M. Phillips, and F. Li, "Quality and efficiency for kernel density estimates in large data," in *SIGMOD*, 2013, pp. 433–444.

[98] Y. Zheng, Y. Ou, A. Lex, and J. M. Phillips, "Visualization of big spatial data using coresets for kernel density estimates," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 524–534, 2021. [Online]. Available: https://doi.org/10.1109/TBDATA.2019.2913655.

[99] Y. Zheng and J. M. Phillips, "L∞ error and bandwidth selection for kernel density estimates of large data," in *SIGKDD*, 2015, pp. 1533–1542. [Online]. Available: http://doi.acm.org/10.1145/2783258.2783357.

[100] H. Zhu and F. Wang, "An agent-based model for simulating urban crime with improved daily routines," *Computers, Environment and Urban Systems*, vol. 89, p. 101680, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0198971521000879.

[101] R. Zhu, Y. Jia, X. Yang, B. Zheng, B. Wang, and C. Zong, "Multiple continuous top-k queries over data stream," in *ICDE*, 2024, pp. 1575–1588. [Online]. Available: https://doi.org/10.1109/ICDE60146.2024.00129.

[102] R. Zhu, B. Wang, X. Yang, B. Zheng, and G. Wang, "SAP: improving continuous top-k queries over streaming data," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 6, pp. 1310–1328, 2017. [Online]. Available: https://doi.org/10.1109/TKDE.2017.2662236.