EARTH: Accelerating Spatiotemporal Network K-function-based Analytics

Tsz Nam Chan
Hongwei Ye
College of Computer Science and Software Engineering, Shenzhen University
Shenzhen, China
edisonchan@szu.edu.cn
yehongwei2023@email.szu.edu.cn

Leong Hou U
Department of Computer and Information Science, University of Macau
Macao, China
ryanlhu@um.edu.mo

Yun Peng*
School of Artificial Intelligence, Guangzhou University
Guangzhou, China
yunpeng@gzhu.edu.cn

Dingming Wu*
College of Computer Science and Software Engineering, Shenzhen University
Shenzhen, China
dingming@szu.edu.cn

Jianliang Xu
Department of Computer Science, Hong Kong Baptist University
Hong Kong, China
xujl@comp.hkbu.edu.hk

Christian S. Jensen
Department of Computer Science, Aalborg University
Aalborg, Denmark
csj@cs.aau.dk

Abstract

The spatiotemporal network K-function is used widely in diverse domains, e.g., transportation science, criminology, and social science, for analyzing spatiotemporal point patterns in location datasets. Domain experts calculate multiple spatiotemporal network K-functions, considering different spatial and temporal thresholds, to generate a spatiotemporal network K-function plot. However, generating this plot is computationally expensive and does not scale to large-scale location datasets. To address this shortcoming, we propose two novel methods, called edge augmentation with range-tree (EAR) and multi-threshold sharing (MTS), that reduce the time complexity for computing a spatiotemporal network K-function and generating the corresponding plot. By wisely combining these two methods, our new approach, EARTH, achieves the lowest time complexity for generating a spatiotemporal network K-function plot. Experimental results based on four large-scale location datasets show that the new methods enable speedups ranging from 2.26x to 19.04x over the state-of-the-art solutions. The implementation of this paper can be found in https://github.com/edisonchan2013928/EARTH.

CCS Concepts

• **Information systems** → **Geographic information systems**.

Keywords

Spatiotemporal Network K-function, Complexity-Reduced Methods, Orthogonal Range Counting Problem

*Yun Peng and Dingming Wu are the first and second corresponding authors, respectively.

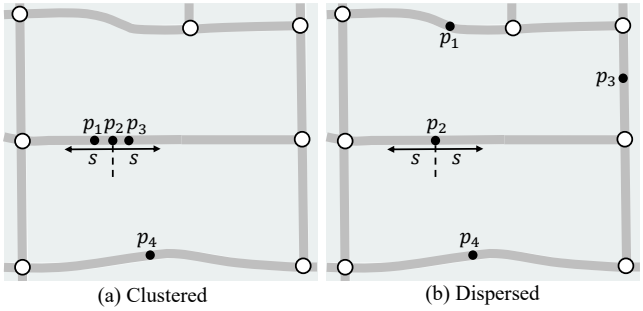
KDD '26, August 09–13, 2026, Jeju Island, Republic of Korea

ACM ISBN 979-8-4007-2259-2/2026/08
https://doi.org/10.1145/3770855.3817706

ACM Reference Format:
Tsz Nam Chan, Hongwei Ye, Leong Hou U, Yun Peng, Dingming Wu, Jianliang Xu, and Christian S. Jensen. 2026. EARTH: Accelerating Spatiotemporal Network K-function-based Analytics. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '26), August 09–13, 2026, Jeju Island, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3770855.3817706

1 Introduction

Point pattern analysis is fundamental functionality in spatial data science/spatial data mining [14, 49]. The network K-function [14, 23, 54] (illustrated in Figure 1), which is one of the tools in point pattern analysis, is used to determine which property, namely clustered (see Figure 1a) or dispersed (see Figure 1b), a location dataset exhibits with respect to different spatial thresholds in a road network. This tool is used extensively in diverse domains, including transportation science [12, 17, 35, 44, 51, 63, 65, 70], criminology [13, 43, 48, 55, 57], and social science [16, 34, 37, 52, 53, 58, 60, 61]. Due to its widespread use, it is supported by prominent software suites such as QGIS/ArcGIS [1, 5] (based on the SANET plugin [7]), spNetwork [9] (an R-package), and NetPointLib [42] (a python package).



(a) Clustered (b) Dispersed

Figure 1: The network K-function counts the number of data points (excluding itself) within a threshold s of each data point (e.g., $2 + 2 + 2 + 0 = 6$ in a and 0 in b) and determines whether the dataset is clustered (with a large count value) or dispersed (with a small count value) with respect to s.

However, the network K-function has a significant drawback in which it does not consider the occurrence time of each location data point. As such, its use can potentially yield misleading results. Figure 2 shows an example of two location datasets. As spatial distributions are the same, the two datasets have the same network K-function value (i.e., 6). Despite this, those blue data points have similar event time, which indicates that the cluster is formed in this region. In contrast, the event time of red data points is far from each other, which can be the occasional (random) cases.

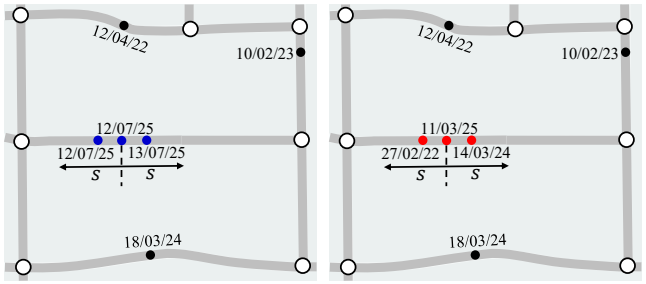


Figure 2: Two location datasets with the same spatial distribution and different temporal distributions.

Recognizing this limitation, domain experts [13, 51] have emphasized the importance of incorporating the time information into the K-function-based analysis. As an example, Moradi et al. [51] state that "... taking the time component into account when analyzing point patterns on networks becomes an important and crucial aspect in the statistical analysis.".

One approach to overcome this limitation is to use the spatiotemporal network K-function [29, 31–33, 50, 51, 64]. This tool counts the number of data points that are simultaneously within a spatial threshold s and temporal threshold t of each data point. To analyze a dataset P, domain experts first specify M spatial thresholds, s_1, s_2,..., s_M, and T temporal thresholds, t_1, t_2,..., t_T, and generate L random datasets with the same size as P (using Monte Carlo simulation [64]). Then, they compute spatiotemporal network K-function values for the $L+1$ datasets with respect to each (s_α, t_β)-pair, where $1 \leq \alpha \leq M$ and $1 \leq \beta \leq T$, in order to generate a spatiotemporal network K-function plot. Figure 3 shows such a plot for the large-scale New York traffic accident dataset [3] with (1) $M = 7$ spatial thresholds (200m to 1400m), (2) $T = 8$ temporal thresholds (3 days to 17 days), and (3) $L = 7$ randomly generated datasets. The green surface shows the spatiotemporal network K-function values for the original dataset, while the red and yellow surfaces denote the minimum and maximum spatiotemporal network K-function values, respectively, for the L randomly generated datasets. Once the green surface is above the yellow surface, this dataset tends to be clustered with respect to those (s_α, t_β)-pairs (e.g., the dataset tends to have meaningful clusters/hotspots with larger spatial thresholds and smaller temporal thresholds.). On the other hand, the dataset tends to be dispersed once the green surface is below the red surface.

However, spatiotemporal network K-function is a time-consuming tool. Given a location dataset with 100,000 data points in a road network, a naïve approach for using this tool can take at least ten billion operations ($100,000 \times 100,000$). Hence, this tool does not scale to support even moderate-scale datasets, let alone to generate a spatiotemporal network K-function plot. This raises the

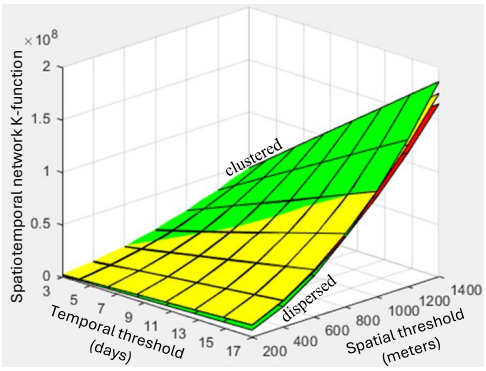


Figure 3: A spatiotemporal network K-function plot.

key question of *whether it is possible to reduce the time complexity and improve the practical efficiency of computing a spatiotemporal network K-function and generating a plot.* To address this issue, we have the following contributions.

- We have established the connection between the spatiotemporal network K-function and the orthogonal range counting problem (see Figures 5 and 6) [10, 28]. Based on this connection, we develop the Edge Augmentation with Range-tree (EAR) method, which constructs the range-tree index structure for each edge in a road network so that it can **reduce the time complexity for computing a spatiotemporal network K-function and generating its plot (see Table 1).**
- Note that generating a spatiotemporal network K-function plot involves the computation of spatiotemporal network K-functions for each dataset with multiple (i.e., M) spatial thresholds and multiple (i.e., T) temporal thresholds (i.e., solving multiple orthogonal range counting problems). By observing that solving multiple orthogonal range counting problems is equivalent to counting data points within the $2T \times M$ grids (see Figure 8a) or $2T \times 1$ grids (see Figure 8b), we develop the Multi-Threshold Sharing (MTS) method, which constructs the prefix-count arrays for each edge in a road network, so that it can **reduce the time complexity for generating a spatiotemporal network K-function plot (see Table 1).**
- We have further developed a unified approach, called EARTH, that selects either the range-tree (from EAR) or the prefix-count array (from MTS) on each edge for each dataset ($L + 1$ datasets in total) in a road network. Our theoretical results show that **EARTH can achieve the lowest time complexity for generating a spatiotemporal network K-function plot (see Table 1).**
- We have conducted extensive experiments (with four large-scale location datasets) to verify that our methods can **achieve speedups ranging from 2.26x to 19.04x over the state-of-the-art solutions.**

The rest of the paper is organized as follows. We first review related work in Section 2. Then, we discuss the preliminaries in Section 3. Next, we present our methods in Section 4. After that, we provide the experimental evaluation in Section 5. Finally, we conclude our paper in Section 6. Appendix is in Section 7.

Table 1: Theoretical results of different exact methods for computing a spatiotemporal network K-function and generating a spatiotemporal network K-function plot, where n, $|E|$, L, M, T, $\mathcal{T}_{\text{SP}}$, and $\mathcal{S}_{\text{SP}}$ denote the number of data points, the number of edges, the number of random datasets, the number of spatial thresholds, the number of temporal thresholds, the time complexity of the shortest path algorithm, and the space complexity of the shortest path algorithm, respectively.

Problem	Method	Time complexity	Space complexity
Spatiotemporal network K-function	RQS [54]	$O(n(\mathcal{T}_{\text{SP}}+n))$	$O(\mathcal{S}_{\text{SP}}+n)$
	SPS [55]	$O(\lvert E\rvert\mathcal{T}_{\text{SP}}+n^2)$	
	EAR (Section 4.1)	$O\big(\lvert E\rvert\mathcal{T}_{\text{SP}}+n\lvert E\rvert\log\big(\frac{n}{\lvert E\rvert}\big)+n\log n\big)$	$O(\mathcal{S}_{\text{SP}}+n)$ (Section 4.4)
Spatiotemporal network K-function plot	RQS [54]	$O(LMTn(\mathcal{T}_{\text{SP}}+n))$	$O(\mathcal{S}_{\text{SP}}+Ln+LMT)$
	SPS [55]	$O(LMT(\lvert E\rvert\mathcal{T}_{\text{SP}}+n^2))$	
	EAR (Section 4.1)	$O\big(\lvert E\rvert\mathcal{T}_{\text{SP}}+LMTn\lvert E\rvert\log\big(\frac{n}{\lvert E\rvert}\big)+Ln\log n\big)$	$O(\mathcal{S}_{\text{SP}}+Ln+LMT)$ (Section 4.4)
	MTS (Section 4.2)	$O(\lvert E\rvert\mathcal{T}_{\text{SP}}+Ln^2+LMTn\lvert E\rvert)$	
	EARTH (Section 4.3)	$O\big(\lvert E\rvert\mathcal{T}_{\text{SP}}+\min\big(LMTn\lvert E\rvert\log\big(\frac{n}{\lvert E\rvert}\big)+Ln\log n, Ln^2+LMTn\lvert E\rvert\big)\big)$	

2 Related Work

We review three lines of research that are related to this work.

Efficient algorithms for network K-function: Several research studies, including the range-query-based solution (RQS) [54, 62] and the shortest path sharing solution (SPS) [55], have been proposed to improve the efficiency for computing a network K-function. These solutions can also be extended to solve our problems (will be discussed in Section 3.2). However, since RQS and SPS exhibit high time complexities, these solutions are still computationally expensive. Recently, Chan et al. [23] propose two representative methods, called count augmentation (CA) and neighbor sharing (NS), for reducing the time complexity of computing a network K-function. Since CA and NS do not consider the temporal component of data points, they cannot be used for computing a spatiotemporal network K-function. Consider Figure 4 as an example. Even though the red and blue data points are consecutive on the edge $\widehat{e}=(a,b)$, only the set of middle and rightmost black data points and the set of leftmost and rightmost black data points on the edge $e=(u,v)$ can contribute to the spatiotemporal network K-function (with $t=5$ days), indicating that these sets do not exhibit (1) the continuity property (e.g., the middle black data point is missing for contributing to the spatiotemporal network K-function with respect to the blue data point) and (2) the sharing property (e.g., the leftmost black data point is not simultaneously in both sets), which are the prerequisites for using CA and NS, respectively. Furthermore, since every data point has its own timestamp, we cannot simply combine the time-range filtering approach with these methods (as we need to apply the time-range filtering for the dataset with respect to each data point, which results in $O(n^2)$ time in the worst-case). More detailed discussions can be found in Section S1 of [24].

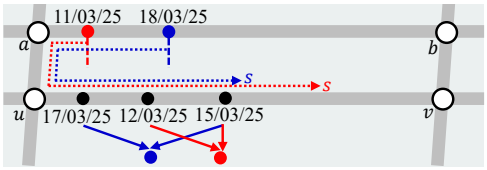


Figure 4: An example (with the spatial threshold s and the temporal threshold $t=5$ days) for illustrating why CA and NS cannot support the spatiotemporal network K-function.

Efficient shortest path algorithms: Recall that spatiotemporal network K-function involves finding the shortest path distances. Although many efficient algorithms for computing the shortest path distances, e.g., shortest path caching [45, 59], hub labeling [11, 41, 46, 47, 56], and hierarchical indexing [38, 69], exist, most of them (e.g., hub labeling and shortest path caching) can only improve the efficiency of distance queries (i.e., single-source-and-single-destination queries) and thus cannot be used to improve the efficiency for solving our problems. In addition, with the large-scale location dataset (i.e., large n), the main bottleneck in computing a spatiotemporal network K-function is to process data points on edges. Therefore, efficient shortest path algorithms cannot substantially improve the efficiency for solving our problems.

Efficient algorithms for kernel density visualization and its variants: Many research studies have been proposed for handling another point pattern analysis tool, called kernel density visualization (KDV) [18, 22, 27, 36, 39, 66, 67], which aims to generate a heatmap visualization based on the kernel density function value of each pixel. Note that the kernel density function involves computing an aggregate function, which is similar to our spatiotemporal network K-function. However, these existing KDV solutions suffer from these two issues. First, all these solutions rely on properties of Euclidean space (e.g., minimum bounding rectangles [18, 36, 39]) for improving efficiency. Second, some solutions [22, 27] further utilize the property of pixels, e.g., consider pixels with the same row (i.e., the same y-coordinate), to reduce the time complexity. Since the spatiotemporal network K-function considers the road network (but not the Euclidean space) and does not involve the concept of pixels, all these solutions [18, 22, 27, 36, 39, 66, 67] cannot be extended to solve our problems. In the literature, some researchers have developed efficient algorithms for supporting spatiotemporal kernel density visualization (STKDV) [19, 20, 68]. However, like [22], all these methods either do not consider the road network or utilize the properties of voxels [19] and pixel-timestamp pairs [20] to develop efficient algorithms, which, therefore, cannot be extended to support this work. Recently, Chan et al. [21, 25, 26] have developed efficient methods for supporting KDV in road networks, called network kernel density visualization (NKDV). However, NKDV does not consider the temporal component in the density function (like the CA and NS methods [23] of the network K-function). Therefore, these methods cannot also be extended to compute a spatiotemporal network K-function.

3 Preliminaries

We first define our problems in Section 3.1. Then, we overview the state-of-the-art solutions for solving the problems in Section 3.2.

Lastly, we discuss the decomposition property of the spatiotemporal network K-function (similar to [23]) in Section 3.3.

3.1 Problem Definitions

We first define the spatiotemporal network K-function in Problem 1.

Problem 1. *Given a road network $G = (V, E)$, a spatiotemporal dataset $P = \{(p_1, \tau_{p_1}), (p_2, \tau_{p_2}), ..., (p_n, \tau_{p_n})\}$ of cardinality n, where each data point lies on one and only one edge, a spatial threshold s, and a temporal threshold t, the spatiotemporal network K-function for P, $K_P(s, t)$, is defined as follows.*

$$K_P(s,t) = \sum_{(p_i,\tau_{p_i})\in P} \sum_{\substack{(p_j,\tau_{p_j})\in P \\ j\neq i}} \mathcal{I}\left(d_G(p_i,p_j) \le s, d(\tau_{p_i},\tau_{p_j}) \le t\right) \quad (1)$$

where d_G, d, and $\mathcal{I}$ denote the shortest path distance, Euclidean distance, and the indicator function, respectively.

Based on Problem 1, we define the spatiotemporal network K-function plot (see Figure 3) in Problem 2.

Problem 2. *Given a road network $G = (V, E)$, a spatiotemporal dataset P, L randomly generated datasets (of cardinality n), i.e., R_1, R_2,..., R_L, M spatial thresholds, i.e., s_1, s_2,..., s_M ($s_1 \le s_2 \le ... \le s_M$), and T temporal thresholds, i.e., t_1, t_2,..., t_T ($t_1 \le t_2 \le ... \le t_T$), we compute $K_P(s_\alpha, t_\beta)$, $\mathcal{L}(s_\alpha, t_\beta)$ (Equation 2), and $\mathcal{U}(s_\alpha, t_\beta)$ (Equation 3) for each (s_α, t_β)-pair, where $1 \le \alpha \le M$ and $1 \le \beta \le T$, in order to generate the spatiotemporal network K-function plot.*

$$\mathcal{L}(s_\alpha,t_\beta) = \min(K_{R_1}(s_\alpha,t_\beta), K_{R_2}(s_\alpha,t_\beta), ..., K_{R_L}(s_\alpha,t_\beta)) \quad (2)$$

$$\mathcal{U}(s_\alpha,t_\beta) = \max(K_{R_1}(s_\alpha,t_\beta), K_{R_2}(s_\alpha,t_\beta), ..., K_{R_L}(s_\alpha,t_\beta)) \quad (3)$$

3.2 Overview of State-of-the-art Solutions

There are two state-of-the-art solutions for efficiently solving Problems 1 and 2.

Range-Query-based Solution (RQS): Consider each data point (p_i, τ_{p_i}) in a dataset P. RQS (based on [54, 62]) needs to (1) invoke the shortest path algorithm for each data point (p_i, τ_{p_i}) to search for all data points (p_j, τ_{p_j}) that are within the spatial threshold s on a road network G and (2) check whether the corresponding $d(\tau_{p_i}, \tau_{p_j}) \le t$ is fulfilled. Therefore, the worst-case time complexity of this method for solving Problem 1 is $O(n(\mathcal{T}_{\text{SP}} + n))$, where $\mathcal{T}_{\text{SP}}$ denotes the time complexity of the shortest path algorithm (e.g., $\mathcal{T}_{\text{SP}} = O(|V| \log |V| + |E|)$ for Dijkstra's algorithm [30]). Recall that we need to compute $L + 1$ spatiotemporal network K-functions for each pair of spatial and temporal thresholds in order to solve Problem 2, which, therefore, takes $O(LMTn(\mathcal{T}_{\text{SP}} + n))$ time.

Shortest Path Sharing Solution (SPS): Rakshit et al. [55] observe that the shortest paths from the nodes, say a and b, on an edge $e = (a, b)$ can be shared with all data points on this edge e. Therefore, instead of calling the shortest path algorithm for each data point (p_i, τ_{p_i}) on e (as in RQS), SPS only invokes the shortest path algorithm on a and b for each $e = (a, b)$ and shares the shortest path distance values with all (p_i, τ_{p_i}) on e. Since the total number of shortest path computations is reduced from $O(n)$ to $O(|E|)$, the time complexities of using SPS to solve Problems 1 and 2 are reduced to $O(|E|\mathcal{T}_{\text{SP}} + n^2)$ and $O(LMT(|E|\mathcal{T}_{\text{SP}} + n^2))$, respectively.

3.3 Decomposition of Spatiotemporal Network K-function

We first define the set of location data points on an edge e (see Definition 1).

Definition 1. *Given a location dataset P and a road network $G = (V, E)$, $P(e)$ denotes the set of location data points on an edge e.*

With this definition, we can decompose the spatiotemporal network K-function $K_P(s, t)$ (Equation 1) by the following expression.

$$K_P(s,t) = \sum_{(p_i,\tau_{p_i})\in P} \sum_{e\in E} C_{P(e)}^{(p_i,\tau_{p_i})}(s,t) \quad (4)$$

where $C_{P(e)}^{(p_i,\tau_{p_i})}(s, t)$ is the point-edge count function (see Equation 5), i.e., the total number of data points in $P(e)$ that are within the spatial threshold s and the temporal threshold t from (p_i, τ_{p_i}).

$$C_{P(e)}^{(p_i,\tau_{p_i})}(s,t) = \sum_{\substack{(p_j,\tau_{p_j})\in P(e) \\ j\neq i}} \mathcal{I}\left(d_G(p_i,p_j) \le s, d(\tau_{p_i},\tau_{p_j}) \le t\right) \quad (5)$$

Therefore, once we can efficiently compute $C_{P(e)}^{(p_i,\tau_{p_i})}(s, t)$, we can improve the efficiency of computing a spatiotemporal network K-function $K_P(s, t)$ and generating its plot.

4 Our Methods

We first discuss two complexity-reduced methods, called edge augmentation with range-tree (EAR) and multi-threshold sharing (MTS), in Sections 4.1 and 4.2, respectively. Then, we propose a unified approach, called EARTH, that integrates EAR and MTS to further reduce the time complexity of generating a spatiotemporal network K-function plot in Section 4.3. Lastly, we discuss the space complexities of all methods in Section 4.4.

4.1 Edge Augmentation with Range-tree (EAR)

In this section, we first cover two core ideas of EAR. Then, we illustrate this method based on them.

Core idea 1: Recall that each data point in the dataset P is on one and only one edge of a road network and contains the timestamp. Therefore, observe from Figure 5 that we can map every black data point (p_j, τ_{p_j}) on an edge $e = (u, v)$ into a two-dimensional plane, where the x-coordinate and the y-coordinate denote the timestamp τ_{p_j} and the spatial distance from p_j to one of the nodes, i.e., u or v, of the edge $e = (u, v)$, respectively.

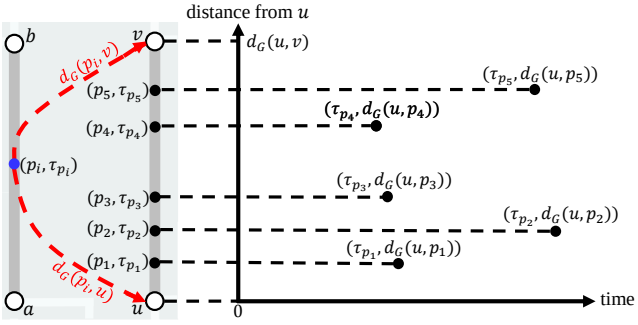
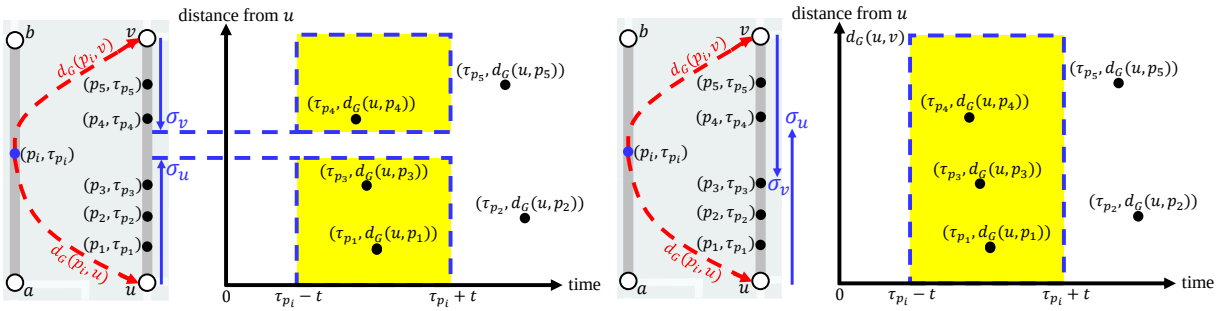


Figure 5: Mapping all black data points on an edge $e = (u, v)$ to the (time, distance)-plane, where the distance is measured from (either) the node u (or node v) to the black data point.

Core idea 2: Given the (time, distance)-plane for each edge e (see Figure 5), we show that computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s, t)$ (see Equation 5) for any data point (p_i, τ_{p_i}) in the dataset P is equivalent to solving the orthogonal range counting problem [10, 28].

Consider the blue data point (p_i, τ_{p_i}) on the edge $\widehat{e} = (a, b)$ in Figure 5. The red dashed lines denote the shortest paths from this data point to the nodes u and v on the edge $e = (u, v)$, where



(a) $\sigma_u + \sigma_v < d_G(u,v)$ (b) $\sigma_u + \sigma_v \geq d_G(u,v)$

Figure 6: Each yellow region denotes the orthogonal range counting query with the two-dimensional range.

$$d_G(p_i,u) = \min \begin{cases} d_G(p_i,a) + d_G(a,u) \\ d_G(p_i,b) + d_G(b,u) \end{cases} \quad (6)$$

$$d_G(p_i,v) = \min \begin{cases} d_G(p_i,a) + d_G(a,v) \\ d_G(p_i,b) + d_G(b,v) \end{cases} \quad (7)$$

There are four possible cases for $d_G(p_i,u)$ and $d_G(p_i,v)$, which are (1) $d_G(p_i,u) > s$ and $d_G(p_i,v) > s$, (2) $d_G(p_i,u) \leq s$ and $d_G(p_i,v) > s$, (3) $d_G(p_i,u) > s$ and $d_G(p_i,v) \leq s$, and (4) $d_G(p_i,u) \leq s$ and $d_G(p_i,v) \leq s$. Here, we discuss Case 4, which is the most complicated case, in detail. Other cases are covered in Section S2 of [24].

Consider $d_G(p_i,u) \leq s$ and $d_G(p_i,v) \leq s$. We note that those data points (p_j,τ_{p_j}) on the edge $e=(u,v)$ with (1) $d_G(p_i,u)+d_G(u,p_j) \leq s$ (i.e., $d_G(u,p_j) \leq \sigma_u$, where $\sigma_u = s - d_G(p_i,u)$) or (2) $d_G(p_i,v) + d_G(v,p_j) \leq s$ (i.e., $d_G(v,p_j) \leq \sigma_v$, where $\sigma_v = s - d_G(p_i,v)$) can fulfill the spatial constraint $d_G(p_i,p_j) \leq s$ in Equation 5. In addition, those data points (p_j,τ_{p_j}) on the edge $e=(u,v)$ with $\tau_{p_i} - t \leq \tau_{p_j} \leq \tau_{p_i} + t$ can fulfill the temporal constraint $d(\tau_{p_i},\tau_{p_j}) \leq t$ in Equation 5.

Therefore, if we have $\sigma_u + \sigma_v < d_G(u,v)$ (see Figure 6a), computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s,t)$ is equivalent to solving two orthogonal range counting queries with the ranges $[\tau_{p_i}-t, \tau_{p_i}+t][0,\sigma_u]$ and $[\tau_{p_i}-t, \tau_{p_i}+t][d_G(u,v)-\sigma_v, d_G(u,v)]$. Otherwise (i.e., $\sigma_u + \sigma_v \geq d_G(u,v)$ in Figure 6b), computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s,t)$ is equivalent to solving the orthogonal range counting query with the range $[\tau_{p_i}-t, \tau_{p_i}+t][0, d_G(u,v)]$.

Illustration of EAR: With the core ideas 1 and 2, once we have obtained the shortest path distances from (p_i,τ_{p_i}) to other nodes, e.g., u and v in Figure 5 (i.e., $d_G(p_i,u)$ (Equation 6) and $d_G(p_i,v)$ (Equation 7)), computing the point-edge count function $C_{P(e)}^{(p_i,\tau_{p_i})}(s,t)$ is equivalent to solving orthogonal range counting queries in the (time, distance)-plane (see Figure 6). Therefore, EAR augments the range-tree index structure [10, 28] for each edge e in the road network, which achieves the lowest time complexity, $O(\log|P(e)|)$, for solving the orthogonal range counting problem in each edge e (see Table 1 in [28]). Hence, computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s,t)$ takes $O(\log|P(e)|)$ time (see Lemma 1).

Lemma 1. *Given a location dataset P, a road network $G=(V,E)$, where each edge is augmented by a range-tree, and the shortest path distances from (p_i,τ_{p_i}) in P to other nodes in V, computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s,t)$ for the data point (p_i,τ_{p_i}) takes $O(\log|P(e)|)$ time.*

The detailed description of EAR is in Section 7.1. Based on Lemma 1, we further show that the time complexity of EAR is $O\big(|E|\mathcal{T}_{\text{SP}} + n|E|\log\big(\frac{n}{|E|}\big) + n\log n\big)$ in Theorem 1. The proof of this theorem is in Section 7.2.

Theorem 1. *The time complexity of EAR to compute a spatiotemporal network K-function is $O\big(|E|\mathcal{T}_{\text{SP}} + n|E|\log\big(\frac{n}{|E|}\big) + n\log n\big)$.*

By adopting simple optimizations, we can reduce the time complexity of EAR for generating a spatiotemporal network K-function plot to $O\big(|E|\mathcal{T}_{\text{SP}} + LMTn|E|\log\big(\frac{n}{|E|}\big) + Ln\log n\big)$ in Theorem 2. More discussions (including the validity of Theorem 2) are in Section 7.3.

Theorem 2. *The time complexity of EAR to generate a spatiotemporal network K-function plot is $O\big(|E|\mathcal{T}_{\text{SP}} + LMTn|E|\log\big(\frac{n}{|E|}\big) + Ln\log n\big)$.*

Compared with the state-of-the-art SPS method, which takes $O(|E|\mathcal{T}_{\text{SP}} + n^2)$ time and $O(LMT(|E|\mathcal{T}_{\text{SP}} + n^2))$ time for computing a spatiotemporal network K-function and generating its plot, respectively, EAR can further reduce the time complexities of solving these two problems (see Theorems 1 and 2).[1]

4.2 Multi-Threshold Sharing (MTS)

In this section, we first discuss two core ideas of MTS. With these two core ideas, we then illustrate this method.

Core idea 3: Consider a set of black data points of cardinality $\widehat{n}$ in a two-dimensional plane (see Figure 7a). Suppose that we have established $\widehat{T} \times \widehat{M}$ grids and scanned all data points. We can assign the count value $CG(g_x,g_y)$ to each (g_x,g_y)-grid (e.g., $CG(1,4)=3$ in Figure 7a). Based on these count values, we build a prefix-count array $\mathcal{P}$ for each (g_x,g_y)-pair (see Figure 7b) [40], where

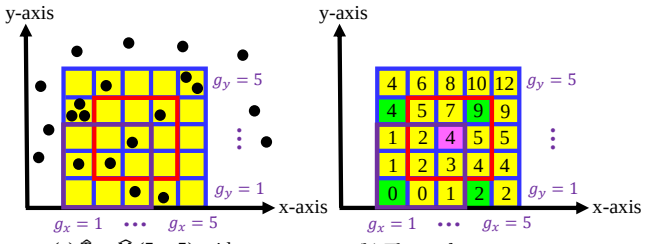
$$\mathcal{P}[g_x,g_y] = \sum_{g_i=1}^{g_x}\sum_{g_j=1}^{g_y} CG(g_i,g_j) \quad (8)$$

For example, $\mathcal{P}[3,3]=4$ (the pink grid) in Figure 7b denotes that there are four data points inside the purple region in Figure 7a.

With this prefix-count array, we can compute the count value $CG(\Omega)$ of any rectangular region $\Omega = [g_{x_l}, g_{x_u}][g_{y_l}, g_{y_u}]$ that fully covers the grids with $g_{x_l} \leq g_x \leq g_{x_u}$ and $g_{y_l} \leq g_y \leq g_{y_u}$ by accessing at most four entries [40]. Using Figure 7 as an example, we can obtain the total count value in the red region of Figure 7a (which is 3) by accessing four green entries in Figure 7b (which is $9-4-2+0$).

Based on [40], Lemma 2 concludes that it takes $O(\widehat{T}\widehat{M}+\widehat{n})$ time to construct the prefix-count array and $O(1)$ time to compute the count value $CG(\Omega)$ for all grids that are fully covered by any rectangular region $\Omega = [g_{x_l}, g_{x_u}][g_{y_l}, g_{y_u}]$.

[1] The main reason is that $O\big(\log\big(\frac{n}{|E|}\big)\big) < O\big(\frac{n}{|E|}\big) \implies O\big(n|E|\log\big(\frac{n}{|E|}\big)\big) < O(n^2)$.



(a) $\widehat{T} \times \widehat{M}$ (5 × 5) grids (b) The prefix-count array

Figure 7: Partition the yellow region into $\widehat{T} \times \widehat{M}$ grids and construct the prefix-count array, where $\widehat{T} = \widehat{M} = 5$ in this example.

Lemma 2. *[40] Given a two-dimensional $\widehat{T} \times \widehat{M}$ grids and a set of data points of cardinality $\widehat{n}$, constructing the prefix-count array for these grids and computing the count value $CG(\Omega)$ for all grids that are fully covered by any rectangular region $\Omega = [g_{x_l}, g_{x_u}][g_{y_l}, g_{y_u}]$ take $O(\widehat{T}\widehat{M} + \widehat{n})$ time and $O(1)$ time, respectively.*

Core idea 4: With the concept of the prefix-count array, we discuss how to adopt this technique to efficiently compute the point-edge count function $C_{P(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ (see Equation 5) with multiple spatial thresholds, $1 \le \alpha \le M$, and multiple temporal thresholds, $1 \le \beta \le T$. Like the EAR method (see Section 4.1), we also have $\sigma_{u_\alpha} = s_\alpha - d_G(p_i, u)$ and $\sigma_{v_\alpha} = s_\alpha - d_G(p_i, v)$ (which are similar to Figure 6). In addition, we also consider four possible cases for comparing the shortest path distances from (p_i, τ_{p_i}) to nodes u and v on the edge $e = (u, v)$ with the largest spatial threshold s_M, which are (1) $d_G(p_i, u) > s_M$ and $d_G(p_i, v) > s_M$, (2) $d_G(p_i, u) \le s_M$ and $d_G(p_i, v) > s_M$, (3) $d_G(p_i, u) > s_M$ and $d_G(p_i, v) \le s_M$, and (4) $d_G(p_i, u) \le s_M$ and $d_G(p_i, v) \le s_M$. Here, we discuss the most complicated case, i.e., Case 4, in detail.

Consider $\sigma_{u_M} + \sigma_{v_M} < d_G(u, v)$ (see Figure 8a), which indicates that these two yellow regions (with the largest spatial threshold s_M) never intersect. For each yellow region, we can first establish the $2T \times M$ grids and then construct the prefix-count array (based on the core idea 3), which takes $O(TM + n)$ time (Lemma 2). With these two prefix-count arrays, we can evaluate the range counting queries with different (s_α, t_β)-pairs ($2MT$ in total), which takes $O(TM)$ time (Lemma 2). Hence, the time complexity is $O(TM + n)$ for this subcase.

Consider $\sigma_{u_M} + \sigma_{v_M} \ge d_G(u, v)$ (see Figure 8b), which indicates that these two yellow regions (with the largest spatial threshold s_M) can intersect. Since $\sigma_{u_1} \le \sigma_{u_2} \le \ldots \le \sigma_{u_M}$ and $\sigma_{v_1} \le \sigma_{v_2} \le \ldots \le \sigma_{v_M}$, there exists the smallest integer ℓ such that $\sigma_{u_\alpha} + \sigma_{v_\alpha} \ge d_G(u, v)$ (see Figure 8), where $1 \le \ell \le \alpha \le M$. Therefore, we can first construct a one-dimensional prefix-count array, which is the special case in the core idea 3, for these strips ($2T \times 1$ grids) in $O(T + |P(e)|)$ time (see Lemma 2). Then we adopt this structure to find the count value in each region $[\tau_{p_i} - t_\beta, \tau_{p_i} + t_\beta][0, d_G(u, v)]$ for computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ in $O(1)$ time (see Lemma 2). Hence, the time complexity of computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ with $\ell \le \alpha \le M$ and $1 \le \beta \le T$ is $O(T(M-\ell)+|P(e)|)$ time. On the other hand, if $1 \le \alpha < \ell$ and $1 \le \beta \le T$, computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ for all these α and β is equivalent to the subcase in Figure 8a (as $\sigma_{u_{\ell-1}} + \sigma_{v_{\ell-1}} < d_G(u, v)$ in this condition), which takes $O(T\ell + |P(e)|)$ time. Therefore, the time complexity of this subcase is $O(TM + |P(e)|)$.

As a remark, other cases (i.e., Cases 1 to 3) are discussed in Section S3 of [24], which also take $O(TM + |P(e)|)$ time.

Illustration of MTS: With the core ideas 3 and 4, once we have obtained $d_G(p_i, u)$ (Equation 6) and $d_G(p_i, v)$ (Equation 7), i.e., the red dashed lines in Figure 8, we can use $O(TM + |P(e)|)$ time to evaluate $C_{P(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ with $T \times M$ threshold-pairs on the edge $e = (u, v)$ (see Lemma 3).

Lemma 3. *Given a location dataset P, a road network $G = (V, E)$, M spatial thresholds, i.e., $s_1, s_2, \ldots, s_M$ ($s_1 \le s_2 \le \ldots \le s_M$), T temporal thresholds, i.e., $t_1, t_2, \ldots, t_T$ ($t_1 \le t_2 \le \ldots \le t_T$), and the shortest path distances from (p_i, τ_{p_i}) in P to other nodes in V, the time complexity of computing $C_{P(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ for all (s_α, t_β)-pairs is $O(TM + |P(e)|)$.*

We provide a detailed description of MTS in Section 7.4. Based on Lemma 3, we conclude in Theorem 3 that the time complexity of MTS is $O(|E|\mathcal{T}_{\text{SP}} + Ln^2 + LMTn|E|)$ (see Section 7.5 for its proof).

Theorem 3. *The time complexity of MTS for generating a spatiotemporal network K-function plot is $O(|E|\mathcal{T}_{\text{SP}} + Ln^2 + LMTn|E|)$.*

Compared with EAR, which takes $O\big(|E|\mathcal{T}_{\text{SP}} + LMTn|E| \log\big(\frac{n}{|E|}\big) + Ln \log n\big)$ time for generating a spatiotemporal network K-function plot (see Theorem 2), MTS is superior than EAR when $MT \ge \frac{n}{|E| \log(\frac{n}{|E|})}$ (based on $Ln^2 \le LMTn|E| \log(\frac{n}{|E|})$). In practice, this condition can be satisfied. Using the Los Angeles dataset (crime events) [2] as an example, this dataset contains $n = 2{,}108{,}154$ data points and the road network contains $|E| = 109{,}708$ edges. Observe that $\frac{n}{|E| \log(\frac{n}{|E|})}$ is 4.5063 (the base of the logarithm is 2), which is normally smaller than MT (e.g., $M = 27$ and $T = 11$ are used in [29]).

4.3 EARTH: A Unified Approach

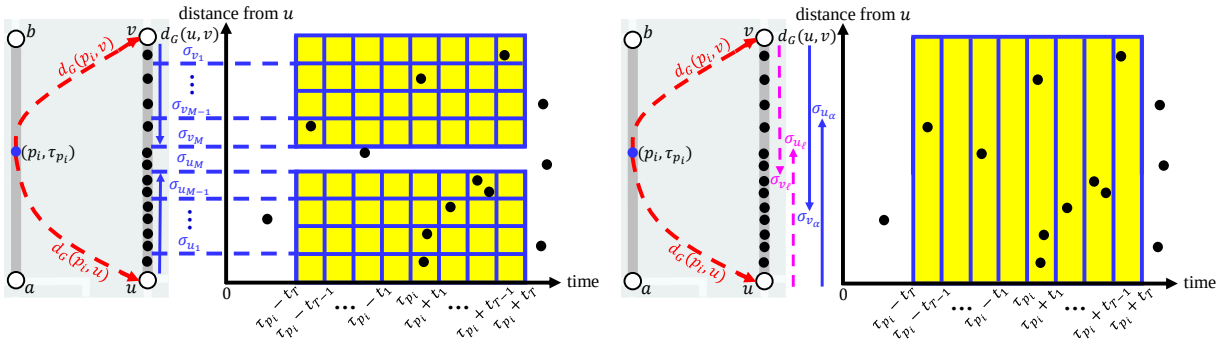
Although MTS is more scalable to MT compared with EAR, the time complexity of MTS (Theorem 3) depends on $O(Ln^2)$, which does not scale to large n. On the other hand, EAR takes $O\big(|E|\mathcal{T}_{\text{SP}} + LMTn|E| \log\big(\frac{n}{|E|}\big) + Ln \log n\big)$ time (Theorem 2), which is more scalable to large n. Hence, we propose EARTH, which integrates EAR, i.e., Edge Augmentation with Range-tree, and MTS, i.e., multi-THreshold sharing, to further boost the efficiency for generating a spatiotemporal network K-function plot.

Recall that we can (1) either choose to build a range-tree (EAR) or a prefix-count array (MTS) on each edge e and (2) both EAR and MTS need to compute the point-edge count function $C_{R_l(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ with MT (s_α, t_β)-pairs for all data points (p_i, τ_{p_i}) in each dataset R_l ($0 \le l \le L$). As such, we model the cost values of EAR and MTS on an edge e for all data points (p_i, τ_{p_i}) in the dataset R_l (of cardinality n) to be $\mathbb{C}_{\text{EAR}}(R_l, e)$ and $\mathbb{C}_{\text{MTS}}(R_l, e)$, respectively, where

$$\mathbb{C}_{\text{EAR}}(R_l, e) = \underbrace{|R_l(e)| \log |R_l(e)|}_{\substack{\text{construction cost of} \\ \text{the range-tree in } R_l(e)}} + \underbrace{nTM \log |R_l(e)|}_{\text{Lemma 1}} \tag{9}$$

$$\mathbb{C}_{\text{MTS}}(R_l, e) = \underbrace{n(TM + |R_l(e)|)}_{\text{Lemma 3}} \tag{10}$$

Using these cost functions, EARTH can select the method with the lowest cost on e to compute $C_{R_l(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ with MT (s_α, t_β)-pairs for all (p_i, τ_{p_i}) in the dataset R_l. Although this approach is simple, EARTH can further reduce the time complexity of generating a spatiotemporal network K-function plot (see Theorem 4) compared with EAR and MTS. The proof is in Section 7.6.



(a) $\sigma_{u_M} + \sigma_{v_M} < d_G(u,v)$ (b) $\sigma_{u_M} + \sigma_{v_M} \geq d_G(u,v)$

Figure 8: Illustration of Case 4 for the MTS method.

Theorem 4. *The time complexity of EARTH for generating a spatiotemporal network K-function plot is* $O\big(|E|\mathcal{T}_{SP} + \min\big(LMTn|E|\log\big(\frac{n}{|E|}\big) + Ln\log n, Ln^2 + LMTn|E|\big)\big)$.

4.4 Space Complexities of All Methods

In this section, we investigate the space complexities of all methods, i.e., EAR, MTS, and EARTH. Since all methods need to access $L+1$ location datasets, P (= R_0), R_1, R_2,..., and R_L, invoke the shortest path algorithm, and return $(L+1)MT$ spatiotemporal network K-function values in order to generate a spatiotemporal network K-function plot, all methods consume at least $O(\mathcal{S}_{\text{SP}} + Ln + LMT)$ space, where $\mathcal{S}_{\text{SP}}$ denotes the space complexity of the shortest path algorithm. In the following, we discuss the additional space complexities of all methods.

EAR: Recall that EAR needs to construct a range-tree for each dataset R_l $(0 \leq l \leq L)$ on each edge e (see Section 4.1). By adopting the state-of-the-art approach [10, 28] to construct this structure for each set $R_l(e)$, which takes $O(|R_l(e)|)$ space, we conclude that the additional space complexity is $\sum_{l=0}^{L}\sum_{e\in E} O(|R_l(e)|) = O(nL)$.

MTS: Since MTS builds a prefix-count array in order to evaluate the point-edge count function $C_{R_l(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ for all (s_α, t_β)-pairs (see Section 4.2) with the dataset R_l, the additional space complexity of MTS is $O(MT)$.[2]

EARTH: EARTH only needs to select either EAR or MTS to compute $C_{R_l(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ for the set $R_l(e)$ with respect to all (s_α, t_β)-pairs (see Section 4.3). As such, the additional space complexity of this method is at most $O(nL + MT)$.

Based on the above discussion, we conclude that all our methods do not increase the space complexity of generating a spatiotemporal network K-function plot compared with the RQS and SPS methods, i.e., the space complexities remain in $O(\mathcal{S}_{\text{SP}} + Ln + LMT)$ (as stated in Theorem 5).

Theorem 5. *The space complexities of the EAR, MTS, and EARTH methods for generating a spatiotemporal network K-function plot are* $O(\mathcal{S}_{SP} + Ln + LMT)$.

As a remark, since computing a spatiotemporal network K-function is the special case of generating its plot, we can set $L = 0$ ($L+1$ datasets ($P = R_0$)), $M = 1$, and $T = 1$ to obtain the space complexity of EAR, i.e., $O(\mathcal{S}_{\text{SP}} + n)$.

[2]After we evaluate $C_{R_l(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ for all (s_α, t_β)-pairs with the set $R_l(e)$, we can clear the prefix-count array. Therefore, the space complexity remains in $O(MT)$.

5 Experimental Evaluation

We first discuss the experimental settings in Section 5.1. Then, we show the experimental results for the time efficiency of computing a spatiotemporal network K-function (see Problem 1) and generating its plot (see Problem 2) in Sections 5.2 and 5.3, respectively. Due to space limitations, space efficiency experiments are provided in Section 7.7.

5.1 Experimental Settings

We choose four large-scale location datasets to conduct the experiments (see Table 2). To obtain the road network of each city (or dataset) from the OpenStreetMap [4], we adopt the commonly used OSMnx python package [15].

Table 2: Datasets.

Name	n	$\lvert V\rvert$	$\lvert E\rvert$	Category	Ref.
Seattle	543,969	11,383	37,472	Crime events	[8]
London	822,382	19,984	75,609	Traffic accidents	[6]
New York	1,542,738	41,307	115,021	Traffic accidents	[3]
Los Angeles	2,108,154	35,933	109,708	Crime events	[2]

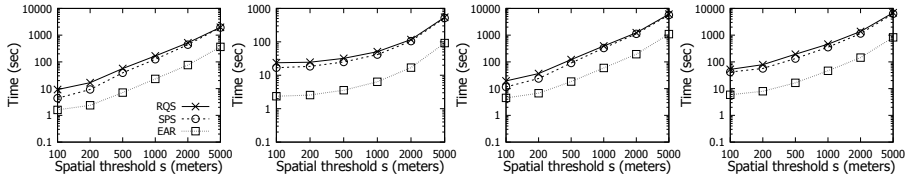
We compare our methods, EAR, MTS, and EARTH, with the state-of-the-art methods, RQS and SPS (see Table 1). We implemented all these methods in C++ and conducted the experiments on an Intel i7 3.19GHz PC with 32GB memory.

5.2 Time-Efficiency for Spatiotemporal Network K-function

We conduct the following experiments to test the time efficiency of RQS, SPS, and EAR. By default, we fix the spatial and temporal thresholds to be 1000m and 7 days, respectively. Since we focus on computing a spatiotemporal network K-function, we omit MTS and EARTH in this section.

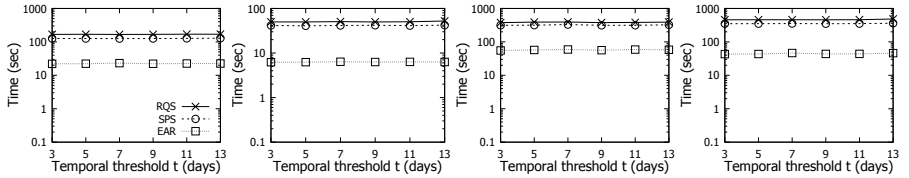
Varying the spatial threshold s. We investigate how s affects the time efficiency of all methods. To conduct this experiment, we vary s from 100m to 5000m. In Figure 9, once we increase s, RQS, SPS, and EAR need to process more data points on road networks. As such, the response time of all methods increases. Since EAR has the lower time complexity compared with RQS and SPS, this method achieves speedups of 2.57x to 8.09x.

Varying the temporal threshold t. We study how t affects the time efficiency of all methods. Since t does not affect the number of data points that are processed by all methods, RQS, SPS, and EAR are not sensitive to t (see Figure 10). EAR achieves speedups of 5.13x to 8.14x over RQS and SPS, due to its lower time complexity.



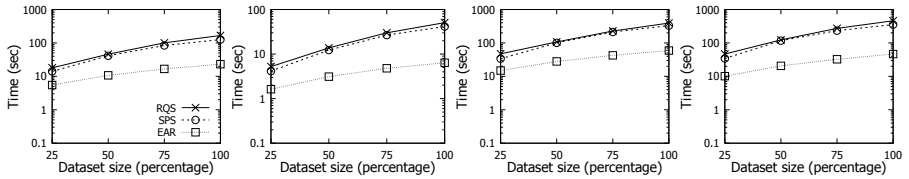
(a) Seattle (b) London (c) New York (d) Los Angeles

Figure 9: Response time for computing a spatiotemporal network K-function with the default temporal threshold $t = 7$ days, varying the spatial threshold s.



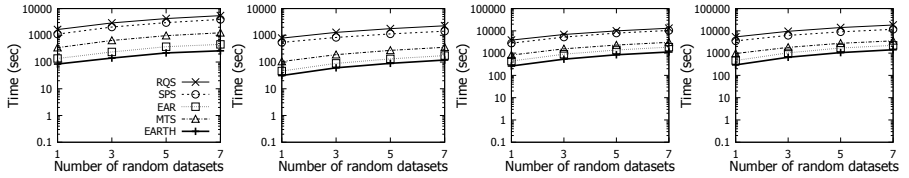
(a) Seattle (b) London (c) New York (d) Los Angeles

Figure 10: Response time for computing a spatiotemporal network K-function with the default spatial threshold $s = 1000$m, varying the temporal threshold t.



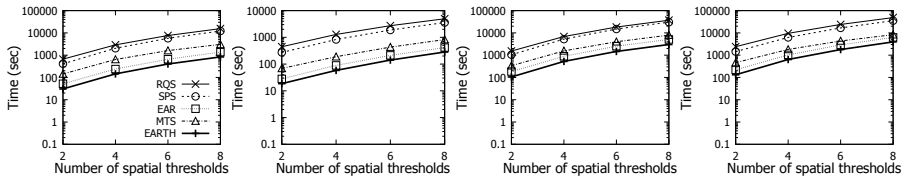
(a) Seattle (b) London (c) New York (d) Los Angeles

Figure 11: Response time for computing a spatiotemporal network K-function with the default spatial threshold $s = 1000$m and the default temporal threshold $t = 7$ days, varying the dataset size.



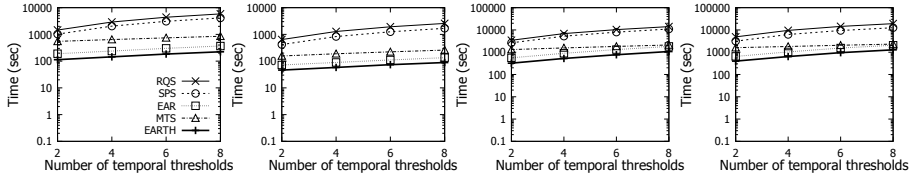
(a) Seattle (b) London (c) New York (d) Los Angeles

Figure 12: Response time for generating a spatiotemporal network K-function plot with $M = 4$ spatial thresholds and $T = 4$ temporal thresholds, varying the number of random datasets L.



(a) Seattle (b) London (c) New York (d) Los Angeles

Figure 13: Response time for generating a spatiotemporal network K-function plot with $L = 3$ random datasets and $T = 4$ temporal thresholds, varying the number of spatial thresholds M.



(a) Seattle (b) London (c) New York (d) Los Angeles

Figure 14: Response time for generating a spatiotemporal network K-function plot with $L = 3$ random datasets and $M = 4$ spatial thresholds, varying the number of temporal thresholds T.

Varying the dataset size. We proceed to investigate how the dataset size affects the time efficiency of all methods. In this experiment, we first sample each dataset with four ratios, i.e., 25%, 50%, 75%, and 100% (no sampling). Then, we measure the response time of all methods in each reduced dataset. In Figure 11, the response time of all methods increases when we increase the dataset size. The main reason is that all methods need to process more data points with larger dataset sizes. Since EAR has the lower time complexity for computing a spatiotemporal network K-function, this method achieves speedups of 2.26x to 7.63x over RQS and SPS.

5.3 Time-Efficiency for Spatiotemporal Network K-function Plot

Generating a spatiotemporal network K-function plot involves the computation of multiple spatiotemporal network K-functions with respect to L random datasets, M spatial thresholds, and T temporal thresholds. Here, we investigate how these parameters affect the efficiency of all methods. As a remark, we set the smallest spatial threshold (i.e., s_1) to be 200m and increase this value by 200m to obtain other spatial thresholds (e.g., $M = 3$ means that we use $s_1 = 200$m, $s_2 = 400$m, and $s_3 = 600$m). Similarly, we also set the smallest temporal threshold (i.e., t_1) to be 3 days and the incremental value to be 2 days for obtaining multiple temporal thresholds (e.g., $T = 3$ means that we use $t_1 = 3$ days, $t_2 = 5$ days, and $t_3 = 7$ days).

Varying the number of random datasets L. We examine how L affects the response time of all methods. To conduct this experiment, we fix M and T to be 4 and vary L from 1 to 7. Figure 12 shows the results of all methods. Since our methods have the lower time complexities than the existing methods, EAR and MTS achieve speedups of 3.07x to 11.72x over RQS and SPS no matter which L we adopt. By combining the advantages of EAR and MTS, EARTH further achieves speedups of 8.26x to 17.31x over RQS and SPS.

Varying the number of spatial thresholds M. To assess how M affects the time efficiency of all methods, we choose M to be 2, 4, 6, and 8 while we adopt the default values of $T = 4$ and $L = 3$. Due to the lower time complexities of our methods, EAR, MTS, and EARTH achieve speedups of 2.86x to 14.67x over RQS and SPS (see Figure 13). Furthermore, when we increase M, the time gap between EAR and MTS is smaller. The main reason is that MTS is more scalable to M compared with EAR.

Varying the number of temporal thresholds T. We investigate how T affects the time efficiency of all methods. To conduct this experiment, we vary T from 2 to 8 and choose L to be 3 and M to be 4. Observe from Figure 14 that all our complexity-reduced methods achieve notable speedups (up to 19.04x) compared with RQS and SPS. As in the previous experiment (i.e., Figure 13), the time gap between EAR and MTS reduces if we adopt the large T.

6 Conclusion

We study the spatiotemporal network K-function, which is used extensively by domain experts for analyzing spatiotemporal point patterns in location datasets. However, computing a spatiotemporal network K-function is already time-consuming, let alone to generate its plot. To tackle this issue, we contribute three methods, namely EAR, MTS, and EARTH, that reduce the time complexity of supporting this tool, without increasing the space complexity. Experiments on four large-scale location datasets show that our proposed methods achieve speedups of 2.26x to 19.04x over the state-of-the-art solutions (i.e., RQS and SPS).

In the future, we will (1) develop plugins (based on our methods) for QGIS and ArcGIS to support this tool and (2) extend our methods to study other tools, e.g., network clustering [54].

Acknowledgments

This work was supported by the Natural Science Foundation of China under grants 62572326, 23IAA00610, 62372308, and 62472116, Scientific Foundation for Youth Scholars of Shenzhen University, the Science and Technology Development Fund Macau SAR (0003/2023/RIC, 0052/2023/RIA1, 0011/2025/RIC, 001/2024/SKL for SKL-IOTSC), Scientific Research Innovation Capability Support Project for Young Faculty SRICSPYF-BS2025017, HK RGC (Project No. 12200524), and Guangdong Basic and Applied Basic Research Foundation (Project No. 2023B1515130002). This work was performed in part at SICC which is supported by SKL-IOTSC, University of Macau.

References

[1] 2026. ArcGIS. https://www.esri.com/en-us/arcgis/products/arcgis-pro/overview.
[2] 2026. Los Angeles Open Data. https://data.lacity.org/A-Safe-City/Crime-Data-from-2010-to-2019/63jg-8b9z.
[3] 2026. NYC Open Data. https://data.cityofnewyork.us/Public-Safety/Motor-Vehicle-Collisions-Crashes/h9gi-nx95.
[4] 2026. OpenStreetMap. https://www.openstreetmap.org/.
[5] 2026. QGIS. https://qgis.org/.
[6] 2026. Road Safety Data. https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data.
[7] 2026. SANET. http://sanet.csis.u-tokyo.ac.jp/.
[8] 2026. Seattle Open Data. https://data.seattle.gov/Public-Safety/SPD-Crime-Data-2008-Present/tazs-3rd5.
[9] 2026. spNetwork: Spatial Analysis on Network - R Project. https://cran.r-project.org/web/packages/spNetwork/index.html.
[10] Pankaj K Agarwal and Jeff Erickson. 1999. Geometric range searching and its relatives. *Contemp. Math.* 223 (1999), 1–56.
[11] Takuya Akiba, Yoichi Iwata, and Yuichi Yoshida. 2013. Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In *SIGMOD*. 349–360. https://doi.org/10.1145/2463676.2465315
[12] Amira K. Al-Aamri, Graeme Hornby, Li-Chun Zhang, Abdullah A. Al-Maniri, and Sabu S. Padmadas. 2021. Mapping road traffic crash hotspots using GIS-based methods: A case study of Muscat Governorate in the Sultanate of Oman. *Spatial Statistics* 42 (2021), 100458.
[13] Adrian Baddeley, Gopalan Nair, Suman Rakshit, Greg McSwiggan, and Tilman M. Davies. 2021. Analysing point patterns on networks — A review. *Spatial Statistics* 42 (2021), 100435.
[14] Adrian Baddeley, Ege Rubak, and Rolf Turner. 2015. *Spatial Point Patterns: Methodology and Applications with R.* Chapman and Hall/CRC Press, London. https://www.routledge.com/Spatial-Point-Patterns-Methodology-and-Applications-with-R/Baddeley-Rubak-Turner/9781482210200/
[15] Geoff Boeing. 2017. OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems* 65 (2017), 126 – 139.
[16] Noli Brazil. 2025. School Closures and the Spatial Ecology of Education Access in 10 U.S. Cities. *Geographical Analysis* 57, 2 (2025), 205–232. https://doi.org/10.1111/gean.12414
[17] Jiannan Cai, Min Deng, Qiliang Liu, Zhanjun He, Jianbo Tang, and Xuexi Yang. 2019. Nonparametric Significance Test for Discovery of Network-Constrained Spatial Co-Location Patterns. *Geographical Analysis* 51, 1 (2019), 3–22. https://doi.org/10.1111/gean.12155 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/gean.12155
[18] Tsz Nam Chan, Reynold Cheng, and Man Lung Yiu. 2020. QUAD: Quadratic-Bound-based Kernel Density Visualization. In *SIGMOD*. 35–50. https://doi.org/10.1145/3318464.3380561
[19] Tsz Nam Chan, Pak Lon Ip, Leong Hou U, Byron Choi, and Jianliang Xu. 2022. SWS: A Complexity-Optimized Solution for Spatial-Temporal Kernel Density Visualization. *Proc. VLDB Endow.* 15, 4 (2022), 814–827.
[20] Tsz Nam Chan, Pak Lon Ip, Bojian Zhu, Leong Hou U, Dingming Wu, Jianliang Xu, and Christian S. Jensen. 2025. Large-scale Spatiotemporal Kernel Density Visualization. In *ICDE*. 99–113. https://doi.org/10.1109/ICDE65448.2025.00015

[21] Tsz Nam Chan, Zhe Li, Leong Hou U, Jianliang Xu, and Reynold Cheng. 2021. Fast Augmentation Algorithms for Network Kernel Density Visualization. *Proc. VLDB Endow.* 14, 9 (2021), 1503–1516.

[22] Tsz Nam Chan, Leong Hou U, Byron Choi, and Jianliang Xu. 2022. SLAM: Efficient Sweep Line Algorithms for Kernel Density Visualization. In *SIGMOD*. ACM, 2120–2134. https://doi.org/10.1145/3514221.3517823

[23] Tsz Nam Chan, Leong Hou U, Yun Peng, Byron Choi, and Jianliang Xu. 2022. Fast Network K-function-based Spatial Analysis. *Proc. VLDB Endow.* 15, 11 (2022), 2853–2866.

[24] Tsz Nam Chan, Leong Hou U, Yun Peng, Dingming Wu, Jianliang Xu, and Christian S. Jensen. 2026. Supplementary Document for "EARTH: Accelerating Spatiotemporal Network K-function-based Analytics". https://github.com/edisonchan2013928/EARTH-supplementary-document/.

[25] Tsz Nam Chan, Hongwei Ye, Bojian Zhu, Leong Hou U, Dingming Wu, Ruisheng Wang, and Joshua Zhexue Huang. 2026. PLAN: Fast and Approximate Gaussian Kernel Density Visualization in Road Networks. In *ICDE (To appear)*.

[26] Tsz Nam Chan, Rui Zang, Bojian Zhu, Leong Hou U, Dingming Wu, and Jianliang Xu. 2024. LION: Fast and High-Resolution Network Kernel Density Visualization. *Proc. VLDB Endow.* 17, 6 (2024), 1255–1268. https://www.vldb.org/pvldb/vol17/p1255-chan.pdf

[27] Tsz Nam Chan, Bojian Zhu, Leong Hou U, Dingming Wu, Wei Tu, and Jianliang Xu. 2026. A Fast, Versatile, and User-friendly Plugin for Kernel Density Analysis. In *ICDE (To appear)*.

[28] Bernard Chazelle. 1988. A Functional Approach to Data Structures and Its Use in Multidimensional Searching. *SIAM J. Comput.* 17, 3 (1988), 427–462. https://doi.org/10.1137/0217026

[29] Lindsey Conrow, Jared Aldstadt, and Natasha S. Mendoza. 2015. A spatio-temporal analysis of on-premises alcohol outlets and violent crime events in Buffalo, NY. *Applied Geography* 58 (2015), 198–205. https://doi.org/10.1016/j.apgeog.2015.02.006

[30] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, 3rd Edition*. MIT Press. http://mitpress.mit.edu/books/introduction-algorithms

[31] Nicoletta D'Angelo and Giada Adelfio. 2025. stopp: An R Package for Spatio-Temporal Point Pattern Analysis. *Journal of Statistical Software* 113, 10 (2025), 1–35. https://doi.org/10.18637/jss.v113.i10

[32] Nicoletta D'Angelo, Giada Adelfio, and Jorge Mateu. 2021. Assessing local differences between the spatio-temporal second-order structure of two point patterns occurring on the same linear network. *Spatial Statistics* 45 (2021), 100534. https://doi.org/10.1016/j.spasta.2021.100534

[33] Nicoletta D'Angelo, Giada Adelfio, and Jorge Mateu. 2023. Local inhomogeneous second-order characteristics for spatio-temporal point processes occurring on linear networks. *Statistical Papers* 64, 3 (2023), 779–805.

[34] Matthias Eckardt and Mehdi Moradi. 2024. Marked spatial point processes: current state and extensions to point processes on linear networks. *Journal of Agricultural, Biological and Environmental Statistics* 29, 2 (2024), 346–378.

[35] Zidong Fang, Ci Song, Hua Shu, Jie Chen, Tianyu Liu, Xi Wang, Xiao Chen, Xiaorui Yan, and Tao Pei. 2023. Length L-function for network-constrained point data. *Transactions in GIS* 27, 2 (2023), 476–493. https://doi.org/10.1111/tgis.13035 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.13035

[36] Edward Gan and Peter Bailis. 2017. Scalable Kernel Density Classification via Threshold-Based Pruning. In *ACM SIGMOD*. 945–959.

[37] Carlos Garrocho-Rangel, José Antonio Álvarez Lobato, and Tania Chávez. 2013. Calculating Intraurban Agglomeration of Economic Units with Planar and Network K-Functions: A Comparative Analysis. *Urban Geography* 34, 2 (2013), 261–286.

[38] Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling. 2008. Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. In *WEA*. 319–333. https://doi.org/10.1007/978-3-540-68552-4_24

[39] Alexander G. Gray and Andrew W. Moore. 2003. Nonparametric Density Estimation: Toward Computational Tractability. In *SDM*. 203–211.

[40] Ching-Tien Ho, Rakesh Agrawal, Nimrod Megiddo, and Ramakrishnan Srikant. 1997. Range Queries in OLAP Data Cubes. In *SIGMOD*, Joan Peckham (Ed.). 73–88. https://doi.org/10.1145/253260.253274

[41] Ruoming Jin, Ning Ruan, Yang Xiang, and Victor E. Lee. 2012. A highway-centric labeling approach for answering distance queries on large sparse graphs. In *SIGMOD*. 445–456. https://doi.org/10.1145/2213836.2213887

[42] Yunfan Kang, Fangzheng Lyu, and Shaowen Wang. 2024. NetPointLib: Library for Large-Scale Spatial Network Point Data Fusion and Analysis. In *PEARC*. ACM, 43:1–43:4. https://doi.org/10.1145/3626203.3670615

[43] Shoaib Khalid, Fariha Shoaib, Tianlu Qian, Yikang Rui, Arezu Imran Bari, Muhammad Sajjad, Muhammad Shakeel, and Jiechen Wang. 2018. Network constrained spatio-temporal hotspot mapping of crimes in Faisalabad. *Applied Spatial Analysis and Policy* 11, 3 (2018), 599–622.

[44] David S. Lamb, Joni A. Downs, and Chanyoung Lee. 2016. The network K-function in context: examining the effects of network structure on the network K-function. *Transactions in GIS* 20, 3 (2016), 448–460. https://doi.org/10.1111/tgis.12157 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.12157

[45] Lei Li, Mengxuan Zhang, Wen Hua, and Xiaofang Zhou. 2020. Fast Query Decomposition for Batch Shortest Path Processing in Road Networks. In *ICDE*. 1189–1200. https://doi.org/10.1109/ICDE48307.2020.00107

[46] Wentao Li, Miao Qiao, Lu Qin, Ying Zhang, Lijun Chang, and Xuemin Lin. 2022. Distance labeling: on parallelism, compression, and ordering. *VLDB J.* 31, 1 (2022), 129–155. https://doi.org/10.1007/s00778-021-00694-1

[47] Ye Li, Leong Hou U, Man Lung Yiu, and Ngai Meng Kou. 2017. An Experimental Study on Hub Labeling based Shortest Path Algorithms. *Proc. VLDB Endow.* 11, 4 (2017), 445–457. https://doi.org/10.1145/3186728.3164141

[48] Yongmei Lu and Xuwei Chen. 2007. On the false alarm of planar K-function when analyzing urban crime distributed along streets. *Social Science Research* 36, 2 (2007), 611–632. https://doi.org/10.1016/j.ssresearch.2006.05.003

[49] Evgenia Martynova and Johannes Textor. 2024. A Uniformly Bounded Correlation Function for Spatial Point Patterns. In *SIGKDD*. ACM, 2177–2188. https://doi.org/10.1145/3637528.3671891

[50] Mehdi Moradi and Ali Sharifi. 2024. Summary statistics for spatio-temporal point processes on linear networks. *Spatial Statistics* 61 (2024), 100840. https://doi.org/10.1016/j.spasta.2024.100840

[51] M Mehdi Moradi and Jorge Mateu. 2020. First-and second-order characteristics of spatio-temporal point processes on linear networks. *Journal of Computational and Graphical Statistics* 29, 3 (2020), 432–443.

[52] Wataru Morioka, Mei-Po Kwan, Atsuyuki Okabe, and Sara L. McLafferty. 2023. Local Indicator of Spatial Agglomeration between Newly Opened Outlets and Existing Competitors on a Street Network. *Geographical Analysis* 55, 3 (2023), 450–465. https://doi.org/10.1111/gean.12343 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/gean.12343

[53] Jianhua Ni, Tianlu Qian, Changbai Xi, Yikang Rui, and Jiechen Wang. 2016. Spatial Distribution Characteristics of Healthcare Facilities in Nanjing: Network Point Pattern Analysis and Correlation Analysis. *International Journal of Environmental Research and Public Health* 13, 8 (2016). https://www.mdpi.com/1660-4601/13/8/833

[54] A. Okabe and K. Sugihara. 2012. *Spatial Analysis Along Networks: Statistical and Computational Methods*. Wiley. https://books.google.com.hk/books?id=48GRqj51_W8C

[55] Suman Rakshit, Adrian Baddeley, and Gopalan Nair. 2019. Efficient Code for Second Order Analysis of Events on a Linear Network. *Journal of Statistical Software, Articles* 90, 1 (2019), 1–37. https://doi.org/10.18637/jss.v090.i01

[56] Michael N. Rice and Vassilis J. Tsotras. 2010. Graph Indexing of Road Networks for Shortest Path Queries with Label Restrictions. *Proc. VLDB Endow.* 4, 2 (2010), 69–80. https://doi.org/10.14778/1921071.1921074

[57] Lior Rokach, Oded Maimon, and Erez Shmueli. 2023. *Machine Learning for Data Science Handbook*. Springer.

[58] Yikang Rui, Zaigui Yang, Tianlu Qian, Shoaib Khalid, Nan Xia, and Jiechen Wang. 2016. Network-constrained and category-based point pattern analysis for Suguo retail stores in Nanjing, China. *International Journal of Geographical Information Science* 30, 2 (2016), 186–199.

[59] Jeppe Rishede Thomsen, Man Lung Yiu, and Christian S. Jensen. 2012. Effective caching of shortest paths for location-based services. In *SIGMOD*. 313–324. https://doi.org/10.1145/2213836.2213872

[60] Teng Wang, Yandong Wang, Xiaoming Zhao, and Xiaokang Fu. 2018. Spatial distribution pattern of the customer count and satisfaction of commercial facilities based on social network review data in Beijing, China. *Computers, Environment and Urban Systems* 71 (2018), 88–97. https://doi.org/10.1016/j.compenvurbsys.2018.04.005

[61] Zhensheng Wang and Ke Nie. 2019. Measuring Spatial Patterns of Health Care Facilities and Their Relationships with Hypertension Inpatients in a Network-Constrained Urban System. *International Journal of Environmental Research and Public Health* 16, 17 (2019). https://www.mdpi.com/1660-4601/16/17/3204

[62] Zhixiao Xie and Jun Yan. 2008. Kernel Density Estimation of traffic accidents in a network space. *Computers, Environment and Urban Systems* 32, 5 (2008), 396 – 406. https://doi.org/10.1016/j.compenvurbsys.2008.05.001

[63] Ikuho Yamada and Jean-Claude Thill. 2004. Comparison of planar and network K-functions in traffic accident analysis. *Journal of Transport Geography* 12, 2 (2004), 149–158. https://doi.org/10.1016/j.jtrangeo.2003.10.006

[64] Xinyue Ye, Xiao Xu, Jay Lee, Xinyan Zhu, and Ling Wu. 2015. Space–time interaction of residential burglaries in Wuhan, China. *Applied Geography* 60 (2015), 210–216. https://doi.org/10.1016/j.apgeog.2014.11.022

[65] Rui Zhang, Bin Shuai, Pengfei Gao, and Yulong Li. 2025. Capturing signals of road traffic safety risk: based on the spatial-temporal correlation between traffic violations and crashes. *Traffic Injury Prevention* 26, 5 (2025), 557–566. https://doi.org/10.1080/15389588.2024.2427270

[66] Yan Zheng, Jeffrey Jestes, Jeff M. Phillips, and Feifei Li. 2013. Quality and efficiency for kernel density estimates in large data. In *SIGMOD*. 433–444.

[67] Yan Zheng and Jeff M. Phillips. 2015. L∞ Error and Bandwidth Selection for Kernel Density Estimates of Large Data. In *SIGKDD*. 1533–1542. https://doi.org/10.1145/2783258.2783357

[68] Yue Zhong, Tsz Nam Chan, Leong Hou U, Dingming Wu, Wei Tu, Ruisheng Wang, and Joshua Zhexue Huang. 2025. A Fast and Accurate Block Compression Solution for Spatiotemporal Kernel Density Visualization. In *SIGKDD*. ACM, 4098–4109. https://doi.org/10.1145/3711896.3736821

[69] Andy Diwen Zhu, Hui Ma, Xiaokui Xiao, Siqiang Luo, Youze Tang, and Shuigeng Zhou. 2013. Shortest path and distance queries on road networks: towards bridging theory and practice. In *SIGMOD*. 857–868. https://doi.org/10.1145/2463676.2465277

[70] Ömür Kaygisiz, Şebnem Düzgün, Ahmet Yildiz, and Metin Senbil. 2015. Spatio-temporal accident analysis for accident prevention in relation to behavioral factors in driving: The case of South Anatolian Motorway. *Transportation Research Part F: Traffic Psychology and Behaviour* 33 (2015), 128–140. https://doi.org/10.1016/j.trf.2015.07.002

7 Appendix

7.1 Detailed Description of EAR

Algorithm 1 shows the pseudocode of EAR. This algorithm first augments a range-tree for the set of data points $P(e)$ of each edge e (lines 3 and 4). Then, this algorithm iteratively (1) obtains the shortest path distances from node a and node b to other nodes with their distance values smaller than or equal to s (line 7) for each edge $\widehat{e} = (a, b)$ (line 5) and (2) computes, for each data point (p_i, τ_{p_i}) on $\widehat{e} = (a, b)$, $C_{P(e)}^{(p_i,\tau_{p_i})}(s, t)$ with respect to other edges $e = (u, v)$ (Lemma 1) and updates the spatiotemporal network K-function (lines 8 to 12). Lastly, this algorithm clears the memory space that is allocated for $SPD_s(a)$ and $SPD_s(b)$ in line 7 and moves to the next iteration (i.e., next edge $\widehat{e} = (a, b)$ in line 5).

Algorithm 1 Edge Augmentation with Range-Tree (EAR)

1: **procedure** EAR($G = (V, E)$, spatiotemporal dataset $P = \{(p_1, \tau_{p_1}), (p_2, \tau_{p_2}), ..., (p_n, \tau_{p_n})\}$, spatial threshold s, temporal threshold t)
2: $K \leftarrow 0$
3: **for** each edge $e \in E$ **do**
4: Augment the range-tree for $P(e)$ ▷ [10, 28]
5: **for** each edge $\widehat{e} = (a, b) \in E$ **do**
6: //Use the shortest path algorithm for a and b
7: Obtain $SPD_s(a)$ and $SPD_s(b)$, where ($\forall h \in V$):

$$SPD_s(a).h = \begin{cases} d_G(a, h) & \text{if } d_G(a, h) \le s \\ \infty & \text{otherwise} \end{cases} \quad (11)$$

8: **for** each $(p_i, \tau_{p_i}) \in P(\widehat{e})$ **do**
9: **for** each edge $e = (u, v) \in E$ **do**
10: Compute $d_G(p_i, u)$ ▷ Equation 6 and line 7
11: Compute $d_G(p_i, v)$ ▷ Equation 7 and line 7
12: $K \leftarrow K + C_{P(e)}^{(p_i,\tau_{p_i})}(s, t)$ ▷ Lemma 1
13: Clear $SPD_s(a)$ and $SPD_s(b)$ ▷ Clear memory
14: Return K

7.2 Proof of Theorem 1

Proof. In Algorithm 1, we first augment a range-tree for each edge e, which takes $O(|P(e)| \log |P(e)|)$ time [10, 28]. Therefore, lines 3 and 4 take $O(\sum_{e \in E} |P(e)| \log |P(e)|)$ time. Then, for each edge $\widehat{e} = (a, b)$ in E, we compute the shortest path distances from node a and node b to other nodes. As such, line 7 takes $O(|E|\mathcal{T}_{\text{SP}})$ time throughout the loop in line 5. With these shortest path distances, we use $O(1)$ time to compute $d_G(p_i, u)$ (Equation 6) and $d_G(p_i, v)$ (Equation 7) for each edge $e = (u, v)$ and use $O(\log |P(e)|)$ time to evaluate $C_{P(e)}^{(p_i,\tau_{p_i})}(s, t)$ (Lemma 1). Therefore, lines 10 to 12 take $O(\sum_{(p_i,\tau_{p_i}) \in P} \sum_{e \in E} \log |P(e)|)$ time throughout the loop in line 5. Hence, the time complexity of the EAR method is:

$$O\Big(\sum_{e \in E} |P(e)| \log |P(e)| + |E|\mathcal{T}_{\text{SP}} + \sum_{(p_i,\tau_{p_i}) \in P} \sum_{e \in E} \log |P(e)|\Big)$$
$$= O\Big(|E|\mathcal{T}_{\text{SP}} + n|E| \log \Big(\frac{n}{|E|}\Big) + n \log n\Big)$$

The equality holds because $\sum_{e \in E} |P(e)| \log |P(e)| \le n \log n$ and $\sum_{e \in E} \log |P(e)| \le |E| \log \big(\frac{n}{|E|}\big)$ (which has been proved in Theorem 2 of [21]). □

7.3 Generating a Spatiotemporal Network K-function Plot with EAR

We can directly adopt Algorithm 1 to compute the spatiotemporal network K-functions with $M \times T$ (s_α, t_β)-pairs for $L + 1$ datasets, which takes $O\big(LMT\big(|E|\mathcal{T}_{\text{SP}} + n|E| \log \big(\frac{n}{|E|}\big) + n \log n\big)\big)$ time (based on Theorem 1). To further reduce the time complexity for solving this problem, we incorporate the recent approach, called advanced shortest path sharing (see Section 3.4 in [23]), into the EAR method. The general idea is to first compute the shortest path distances from node a and node b on the edge $\widehat{e} = (a, b)$ to other nodes with the largest spatial threshold s_M, i.e., $SPD_{s_M}(a)$ and $SPD_{s_M}(b)$ (Equation 11), and then share these shortest path distances with all datasets, spatial thresholds, and temporal thresholds. By adopting this approach, the total number of shortest path computations remains in $O(|E|)$ (i.e., $O(|E|\mathcal{T}_{\text{SP}})$ time). In addition, since there are $L + 1$ datasets, the time complexity for augmenting range-trees in a road network is $O(Ln \log n)$. Hence, we conclude in Theorem 2 that generating a spatiotemporal network K-function plot only takes $O\big(|E|\mathcal{T}_{\text{SP}} + LMTn|E| \log \big(\frac{n}{|E|}\big) + Ln \log n\big)$ time.

7.4 Detailed Description of MTS

In Algorithm 2, it first invokes the shortest path algorithm with the largest spatial threshold s_M for nodes a and b on each edge $\widehat{e} = (a, b)$ (line 5). Then, for every data point (p_i, τ_{p_i}) on this edge $\widehat{e} = (a, b)$ with each dataset R_l (line 7), where $0 \le l \le L$, this algorithm (1) obtains the shortest path distances from (p_i, τ_{p_i}) to the nodes, say u and v, on other edges $e = (u, v)$ (lines 9 and 10), (2) computes $C_{R_l(e)}^{(p_i,\tau_{p_i})}(s_\alpha, t_\beta)$ with respect to all (s_α, t_β)-pairs based on the core ideas 3 and 4 (line 12), and (3) updates the spatiotemporal network K-function values for all (s_α, t_β)-pairs with the dataset R_l (lines 13 to 15). After that, MTS iteratively processes those data points on the edge $\widehat{e} = (a, b)$ for the next dataset (line 6). Once all datasets on this edge $\widehat{e} = (a, b)$ have been processed, MTS clears the memory space that is allocated for computing the shortest path distances (line 16) and iteratively moves to the next edge $\widehat{e}$ (line 4) until all edges have been accessed.

7.5 Proof of Theorem 3

Proof. In Algorithm 2, we invoke the shortest path algorithm twice for nodes a and b on each edge $\widehat{e} = (a, b)$ (line 5). As such, the time complexity of line 5 for all iterations is $O(|E|\mathcal{T}_{\text{SP}})$. With these shortest path distances, $SPD_{s_M}(a)$ and $SPD_{s_M}(b)$, we can first obtain $d_G(p_i, u)$ (line 9) and $d_G(p_i, v)$ (line 10) in $O(1)$ time using Equations 6 and 7, respectively. Based on $d_G(p_i, u)$ and $d_G(p_i, v)$,

Algorithm 2 Multi-Threshold Sharing (MTS)

1: **procedure** MTS(Datasets $\{P, R_1, R_2, ..., R_L\}$, $G = (V, E)$, spatial thresholds s_1, s_2,..., s_M, temporal thresholds t_1, t_2,..., t_T)
2: Let $R_0 = P$ and $K^{(l)}_{\alpha,\beta}$ be the spatiotemporal network K-function for the dataset R_l with the spatial threshold s_α and the temporal threshold t_β.
3: $K^{(l)}_{\alpha,\beta} \leftarrow 0$ $(1 \le \alpha \le M, 1 \le \beta \le T, 0 \le l \le L)$
4: **for** each edge $\widehat{e} = (a, b) \in E$ **do**
5: Obtain $SPD_{s_M}(a)$ and $SPD_{s_M}(b)$ ▷ Equation 11
6: **for** $l \leftarrow 0$ to L **do**
7: **for** each $(p_i, \tau_{p_i}) \in R_l(\widehat{e})$ **do** ▷ Definition 1
8: **for** each edge $e = (u, v) \in E$ **do**
9: Compute $d_G(p_i, u)$ ▷ Equation 6
10: Compute $d_G(p_i, v)$ ▷ Equation 7
11: //Adopt the core ideas 3 and 4
12: Compute $C^{(p_i,\tau_{p_i})}_{R_l(e)}(s_\alpha, t_\beta)$, $\forall(s_\alpha, t_\beta)$
13: **for** $\alpha \leftarrow 1$ to M **do**
14: **for** $\beta \leftarrow 1$ to T **do**
15: $K^{(l)}_{\alpha,\beta} \leftarrow K^{(l)}_{\alpha,\beta} + C^{(p_i,\tau_{p_i})}_{R_l(e)}(s_\alpha, t_\beta)$
16: Clear $SPD_{s_M}(a)$ and $SPD_{s_M}(b)$
17: Return $K^{(l)}_{\alpha,\beta}$ $(1 \le \alpha \le M, 1 \le \beta \le T, 0 \le l \le L)$

we can then compute $C^{(p_i,\tau_{p_i})}_{R_l(e)}(s_\alpha, t_\beta)$ for all (s_α, t_β)-pairs (line 12) in $O(TM + |R_l(e)|)$ time (see Lemma 3) and update these spatiotemporal network K-function values (lines 13 to 15) in $O(TM)$ time. Therefore, the time complexity of lines 9 to 15 is $O(TM + |R_l(e)|)$ for each iteration. Hence, we can conclude that the time complexity of lines 9 to 15 for all iterations is:

$$O\Big(\sum_{\widehat{e}\in E}\sum_{l=0}^{L}\sum_{(p_i,\tau_{p_i})\in R_l(\widehat{e})}\sum_{e\in E}(TM + |R_l(e)|)\Big) = O(Ln^2 + LMTn|E|)$$

Based on the above discussion, we conclude that the time complexity of MTS is $O(|E|\mathcal{T}_{\text{SP}} + Ln^2 + LMTn|E|)$. □

7.6 Proof of Theorem 4

Proof. Since this method does not affect the total number of shortest path computations, the time complexity of this part remains in $O(|E|T_{\text{SP}})$. Recall that we choose either EAR or MTS to compute $C^{(p_i,\tau_{p_i})}_{R_l(e)}(s_\alpha, t_\beta)$ on an edge e with MT (s_α, t_β)-pairs for all data points (p_i, τ_{p_i}) in R_l based on the cost values (see Equations 9 and 10). Since there are $L + 1$ datasets and $|E|$ edges, the time complexity of EARTH is:

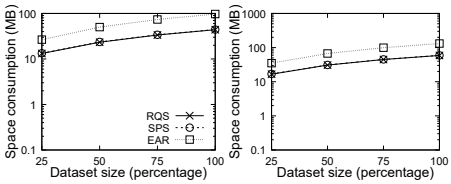
$$\begin{aligned}
&O\Big(|E|T_{\text{SP}} + \sum_{l=0}^{L}\sum_{e\in E}\min(\mathbb{C}_{\text{EAR}}(R_l, e), \mathbb{C}_{\text{MTS}}(R_l, e))\Big)\\
&\le O\Big(|E|T_{\text{SP}} + \min\Big(\sum_{l=0}^{L}\sum_{e\in E}\mathbb{C}_{\text{EAR}}(R_l, e), \sum_{l=0}^{L}\sum_{e\in E}\mathbb{C}_{\text{MTS}}(R_l, e)\Big)\Big)\\
&= O\Big(|E|T_{\text{SP}} + \min\Big(LMTn|E|\log\Big(\frac{n}{|E|}\Big) + Ln\log n, Ln^2 + LMTn|E|\Big)\Big)
\end{aligned}$$

□

7.7 Space-Efficiency for Spatiotemporal Network K-function and its Plot

In this section, we investigate the space efficiency of all methods for computing a spatiotemporal network K-function and its plot. We adopt the two largest datasets in Table 2, which are New York and Los Angeles, for conducting the following experiments.

Varying the dataset size. In the first experiment, we measure the memory space consumption of all methods, i.e., RQS, SPS, and EAR, for computing a spatiotemporal network K-function, which follows the same settings in Section 5.2 (i.e., choosing the default spatial threshold and the default temporal threshold to be 1000m and 7 days, respectively, and using 25%, 50%, 75%, and 100% as four sampling ratios).

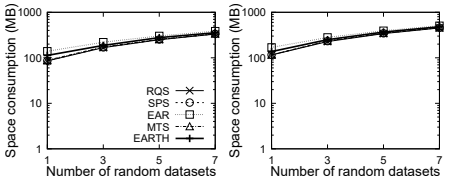


(a) New York (b) Los Angeles

Figure 15: Space consumption of all methods for computing a spatiotemporal network K-function, varying the dataset size.

Observe from Figure 15 that the larger the dataset size, the larger the memory space consumption of all methods. The main reason is that the space complexities of all methods, RQS, SPS, and EAR, are proportional with respect to the dataset size n (see Table 1). With the same space complexities of all methods, EAR only incurs a small amount of space overheads compared with RQS and SPS.

Varying the number of random datasets L. In the second experiment, we follow the same settings in Section 5.3 (i.e., setting the default number of spatial thresholds M and the default number of temporal thresholds T to be 4 and varying the number of random datasets L from 1 to 7) for measuring the memory space consumption of all methods.



(a) New York (b) Los Angeles

Figure 16: Space consumption of all methods for generating a spatiotemporal network K-function plot with $M = 4$ spatial thresholds and $T = 4$ temporal thresholds, varying the number of random datasets L.

Figure 16 shows the results of all methods. Observe that the memory space consumption of all methods increases with respect to L. The main reason is that the space complexities of all methods (see Table 1) are proportional to L. With the same space complexities of all methods, EAR, MTS, and EARTH, only incur slight space overhead compared with the existing methods, RQS and SPS.