A Privacy-Preserving Scheme With High Utility Over Data Streams in Mobile Crowdsensing

Zhimao Gong, Jiapeng Zhang, Haotian Wang, Mingxing Duan, *Member, IEEE*, Keqin Li, *Fellow, IEEE*, and Kenli Li, *Senior Member, IEEE*

***Abstract*—Both truth discovery and pattern analysis are effective methods for extracting valuable insights from data streams in mobile crowdsensing. However, existing privacy-preserving schemes either suffer from low data utility or provide high utility at the cost of weak privacy protection. To address this challenge, we introduce a robust privacy-preserving scheme that facilitates high-utility truth discovery and pattern analysis over mobile crowdsensing data streams. Concretely, we leverage the Square Wave mechanism, a randomized reporting technique, to perturb the data to prevent privacy breaches. To reduce the utility loss caused by perturbation, we design a budget allocation algorithm. This algorithm ensures that adjacent timestamps with approximate data share a perturbed value derived from their accumulated budgets. Furthermore, to facilitate robust pattern analysis, we propose a data splitting method that divides the perturbed data into two parts: one part records patterns randomly, while the other part recovers the perturbed values. Theoretical analysis confirms that our scheme satisfies ω-event ϵ-differential privacy level. Extensive experiments conducted on four real-world datasets demonstrate that our scheme outperforms existing schemes, delivering more accurate results for both truth discovery and pattern analysis under the same privacy constraints.**

***Index Terms*—Data streams, differential privacy, mobile crowdsensing, patterns, truth discovery.**

I. Introduction

MOBILE crowdsensing [1], [2] is a versatile and expansive data collection paradigm organized as a collaborative interaction between a central server and multiple users. In this model, the server publishes tasks specifying the type and format of data to be collected, and users contribute data that matches these requirements using their portable mobile devices (e.g., smartphones). The collected data then enables the server to assist users in making more informed decisions. For example, GreenGPS [3] utilizes fuel consumption data to recommend fuel-efficient routes for drivers, and AirClowd [4] integrates fused sensor data to suggest travel routes for users with lower PM2.5 exposure.

Among various data mining methods in mobile crowdsensing, truth discovery and pattern analysis are two effective approaches for extracting valuable information. The former takes cross-sectional datasets collected in each round as input, assigns a weight to each data source based on its reliability, and derives the ground truth through weighted aggregation [5], [6]. The latter, on the other hand, uses time-series datasets as input and identifies temporal correlations by analyzing the changes between consecutive data points [7]. As mobile crowdsensing continues to evolve, privacy concerns have also garnered increasing attention from both the public and the research community. Mobile crowdsensing typically involves continuous data collection, and data streams are more likely to contain sensitive information, especially when linked to personal attributes [8], [9]. If the server managing the data is compromised or untrustworthy, it could lead to personal privacy breaches. This risk may cause users to hesitate to share their data, thereby limiting the potential growth of mobile crowdsensing applications.

Many studies have discussed this issue and proposed corresponding privacy-preserving solutions. Among them, differential privacy is a widely adopted mechanism, as it provides robust mathematical security guarantees with negligible additional computational overhead. Under this mechanism, users submit perturbed data with noise instead of raw data. However, the added noise will also result in computational errors, known as utility loss, which is influenced by the privacy budget. Early studies [10], [11], [12] primarily focused on one-time data perturbation, overlooking the temporal correlation of the data. These methods often introduced excessive noise, causing the truth discovery results from the perturbed datasets to deviate significantly from the ground truth. Wang et al. [13] recognized this issue and were the first to explore privacy-preserving streaming truth discovery. In their scheme, the published data are selected from the current and previous perturbed data, with the value closest to the local truth established by edge servers. Similarly, Pang et al. [14] investigated a privacy-

Received 2 October 2024; revised 13 April 2025 and 24 May 2025; accepted 25 May 2025. Date of publication 29 May 2025; date of current version 5 June 2025. This work was supported in part by the Creative Research Groups Program of the National Natural Science Foundation of China under Grant 62321003; in part by the National Natural Science Foundation of China under Grant 62172151, Grant 62422205, Grant U24A20255, Grant 62272149, Grant 62272158, and Grant 62302160; in part by the Natural Science Foundation of Hunan Province under Grant 2023JJ40162 and Grant 2024JJ6154; and in part by the China National Postdoctoral Program for Innovative Talents under Grant BX20240111. The associate editor coordinating the review of this article and approving it for publication was Prof. Haibo Hu. *(Corresponding author: Jiapeng Zhang.)*

Zhimao Gong, Jiapeng Zhang, Haotian Wang, Mingxing Duan, and Kenli Li are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan 410082, China (e-mail: zmgong@hnu.edu.cn; zhangjp@hnu.edu.cn; wanghaotian@hnu.edu.cn; duanmingxing@hnu.edu.cn; lkl@hnu.edu.cn).

Keqin Li is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan 410082, China, also with the National Supercomputing Center in Changsha, Changsha, Hunan 410082, China, and also with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Digital Object Identifier 10.1109/TIFS.2025.3574967

TABLE I
COMPARISON WITH EXISTING SCHEMES

	BA [10]	PrivSTD [13]	PPPTD [14]	PriPTD [15]	PatternLDP [7]	PPLDP [16]	ASRT [17]	Our Scheme
Full Data Protection	✓	✓	✓	✓	✗	✗	✗	✓
Truth Discovery Utility	Low	Middle	Low	High	Very High	Very High	Very High	High
Pattern Analysis Utility	Low	Low	Low	Low	Very High	Very High	High	High

preserving truth discovery framework for crowdsourced data streams. They proposed a budget adjustment method and a deviation-aware weighted aggregation strategy to enhance truth discovery accuracy under different scales of noise. Building on these works, Gong et al. [15] further optimized the budget allocation strategy and introduced a noise-aware error adjustment approach to reduce the impact of noise. However, none of these schemes consider the preservation of data patterns, making the perturbed datasets less suitable for pattern analysis.

To fill this gap, Wang et al. [7] were the first to propose a privacy-preserving scheme that retains data patterns. They devised a pattern-aware sampling method to extract remarkable points and perturb them using differential privacy, while other data points are directly published with their true values. Gao and Zhou [16] improved the sampling approach by employing the Least Squares Segmented Linear Fit method and using the Kalman filter for post-processing optimization to enhance data utility. Similarly, unsampled data are released without protection. Li et al. [17] further designed an adaptive sampling method based on dynamic features to distinguish between short-term and stable patterns. For each sampled point, to prevent data distortion caused by perturbed data exceeding the valid data range, they employed the Bounded Laplace mechanism for data perturbation. A common characteristic of these schemes is that the final published dataset contains a certain proportion of real data. Undoubtedly, raw values can preserve data patterns to the greatest extent, but they also weaken privacy guarantees. In particular, for smooth data streams, only a small fraction of the data may be perturbed. *Therefore, there remains a need to explore how to provide robust privacy protection while ensuring high utility data for truth discovery and pattern analysis.*

In this paper, we propose a Privacy-Preserving scheme in Mobile CrowdSensing (PPMCS) to address the aforementioned challenges. Specifically, we employ the Square Wave (SW) randomization mechanism to perturb each data point to prevent any real information from being exposed. To ensure the accuracy of truth discovery, we design a budget allocation algorithm that accumulates the budgets of adjacent proximate data in a time series and then allows them to share a perturbed value. For pattern analysis, we propose a data splitting method that divides the perturbed data into two parts: one for recording data patterns and the other for restoring the perturbed data. Given that retaining the original pattern completely poses a risk of inferring other data if any real data is leaked, we use the piecewise linear approximation method to randomize the output, with the original pattern serving as a baseline. A comparison of our scheme with existing solutions in terms of full data protection, truth discovery utility, and pattern analysis utility is summarized in Table I. The contributions to this paper are as follows:

- We propose a privacy-preserving scheme in mobile crowdsensing that provides high utility for truth discovery and pattern analysis without revealing any real data.
- To ensure the accuracy of truth discovery, we design a budget allocation algorithm aimed at minimizing utility loss. The core idea of the algorithm is to accumulate the budgets of adjacent moments with proximate data and allow these moments to share a value output by the SW mechanism.
- To support pattern analysis, we propose a data splitting method that divides a data stream into two parts: one for recording the data patterns and the other for restoring the perturbed data. To prevent attackers who might infer other data through scaling relationships if they know any real data, we use the piecewise linear approximation method to randomize the original patterns.
- We conduct a theoretical analysis and extensive experiments to validate our scheme. The final experimental data demonstrate that our PPMCS can obtain more accurate results in truth discovery and pattern analysis under the same privacy constraints.

In the rest of this paper, we introduce the preliminaries in Section II and then detail the proposed scheme in Section III. After that, we present the theoretical analysis in Section IV and provide the experimental analysis in Section V. The related work is discussed in Section VI, and the conclusion is shown in Section VII.

II. PRELIMINARIES

In this section, we briefly introduce some of the important preliminaries used in this paper.

A. Mobile Crowdsensing

The mobile crowdsensing model discussed in this paper consists of a central server and multiple user entities. Without loss of generality, we assume that there are M users performing N tasks over T consecutive timestamps. At the initial stage, the server publishes a set of tasks. Subsequently, for any timestamp $t \in [1, T]$ in the time series, users uploads the sensory data $\left\{d_{i,j}^t\right\}_{j=1}^{N}$ to the server, where $d_{i,j}^t$ represents the answer to task j. The problem of truth discovery is to infer results $\left\{z_j^t\right\}_{j=1}^{N}$ that are close to the ground truths

from $\left\{\left\{d_{1,j}^t\right\}_{j=1}^N, \left\{d_{2,j}^t\right\}_{j=1}^N, \cdots, \left\{d_{N,j}^t\right\}_{j=1}^N\right\}$, where z_j^t denotes the discovered truth for task j at timestamp t. Furthermore, let a pattern $P = \left\{d_{i,j}^1, d_{i,j}^2, \cdots, d_{i,j}^t\right\}$ represents continuous data points provided by worker i for task j. The problem of pattern analysis is to learn meaningful trends from P, such as the direction and rate of growth.

In our scheme, to protect individual privacy, users are required to submit perturbed data, denoted as $\left\{\left\{\widetilde{d}_{i,j}^t\right\}_{j=1}^N\right\}_{i=1}^M$, rather than the raw data. Our goal is to ensure that the results of truth discovery and pattern analysis on $\left\{\left\{d_{i,j}^t\right\}_{j=1}^N\right\}_{i=1}^M$ and $\left\{\left\{\widetilde{d}_{i,j}^t\right\}_{j=1}^N\right\}_{i=1}^M$ are close.

B. Differential Privacy

Differential privacy is a mathematical paradigm used to quantify and limit the risk of individual privacy breaches in public data. A common manifestation of differential privacy is ϵ-differential privacy, where ϵ represents the privacy budget.

Definition 1 (ϵ-Differential Privacy [18]): A mechanism $\mathcal{M}$ satisfies ϵ-differential privacy if for all outputs $O \subseteq Range(\mathcal{M})$ on any two neighboring datasets D_1 and D_2 that differ by at most one record there are

$$Pr[\mathcal{M}(D_1) \in O] \leq exp(\epsilon) Pr[\mathcal{M}(D_2) \in O]. \tag{1}$$

Another form of differential privacy for data streams is known as ω-event ϵ-differential privacy (abbreviated as ω-event privacy) [10]. It protects events within an arbitrary window size of ω in a continuous time series. Let S_1 and S_2 represent two prefix streams of length s. We define them as ω-neighboring if, for any indices $1 \leq i_1 \leq i_2 \leq s$ satisfying $S_1[i_1] \neq S_2[i_1]$ and $S_1[i_2] \neq S_2[i_2]$, the condition $i_2 - i_1 + 1 < \omega$ holds. If a mechanism $\mathcal{M}$ can ensure ϵ-differential privacy over any two ω-neighboring prefix streams S_1, S_2 of arbitrary length, we say that $\mathcal{M}$ satisfies ω-event privacy. In general, an ϵ-differential privacy mechanism can be adapted to satisfy ω-event privacy by ensuring that the cumulative budget consumed within any window of size ω does not exceed a given ϵ.

C. Square Wave Mechanism

Square Wave is a numerical distribution domain recoverable randomized reporting mechanism that satisfies the definition of differential privacy. For any input v, SW ensures that the output value $\widetilde{v}$ lies within the interval $[v-b, v+b]$ with higher probability.

Definition 2 (Square Wave Mechanism [19]): Let $\mathcal{D}_1 = [0, 1]$ denote the input domain and $\mathcal{D}_2 = [-b, 1+b]$ denote the output domain, an instance of the Square Wave mechanism $\mathcal{M} : \mathcal{D}_1 \to \mathcal{D}_2$ ensures that for all $v \in \mathcal{D}_1$, its corresponding output probability density function $\mathbf{M}_v(\widetilde{v})$ satisfies

$$\mathbf{M}_v(\widetilde{v}) = \begin{cases} p, & \text{if } |v - \widetilde{v}| \leq b; \\ q, & \text{otherwise}, \end{cases} \tag{2}$$

TABLE II
SYMBOLS DESCRIPTION

Symbol	Decription
M, N, T	The number of users, tasks, and timestamps, respectively.
$d_{i,j}^t$	The raw data about task j collected by user i at timestamp t.
$\widehat{d}_{i,j}^t$	The fitted data of user i to task j at timestamp t.
$\widetilde{d}_{i,j}^t$	The perturbed output of $d_{i,j}^t$.
$\widetilde{d}_{i,j1}^t$, $\widetilde{d}_{i,j2}^t$	The randomized output after splitting $\widetilde{d}_{i,j}^t$.
ϵ, ω	The total budget and sliding window size, respectively.
$\epsilon_{i,j}^t$	The cost budget for perturbing $d_{i,j}^t$.
α	The decay factor in DLS.
β	The threshold for determining whether two data are approximate.
l, τ	The size and fitting error of fitted list, respectively.
k	The number of approximate data.
c^t	The reserved budget for timestamp t in the fitted list.
θ	The tolerable error threshold for slope.
l^{t_1,t_2}	The slope of the line between points d^{t_1} and d^{t_2}.
l_{low}, l_{up}	The lower and upper bounds of the slope.

where b, p, and q are calculated as

$$b = \frac{\epsilon e^\epsilon - e^\epsilon + 1}{2e^\epsilon(e^\epsilon - 1 - \epsilon)}, \; p = \frac{e^\epsilon}{2be^\epsilon + 1}, \; q = \frac{1}{2be^\epsilon + 1}. \tag{3}$$

III. PROPOSED SCHEME

In this section, we first describe the workflow of PPMCS through an overview, followed by detailed descriptions of its components. Some frequently used symbols in this paper are recorded in Table II.

A. Overview

The overview of PPMCS for user i at timestamp t is shown in Fig. 1, where each process is described in the following:

①: The server publishes a task, and user i collects data $d_{i,j}^t$ corresponding to task j using carried mobile device. To minimize the utility loss, the user accumulates the budgets of subsequent adjacent moments with approximate data and uses this accumulated budget as the current budget cost. Since it is impractical to know the future data in advance in a real-time scenario, the user uses a decay-aware least squares method to fit the next l data based on historical data. Then, the user counts the number of adjacent approximate data from the fitted list (assuming there are k) and allocates the cumulative budget ϵ_a proportionally from the remaining budget ϵ_r.

②: Due to fitting errors, k may not always be accurate. For example, the user might find that the fitted data from two adjacent timestamps (e.g., t and $t+1$) appear proximate, but the actual data may not be. If the entire budget for timestamp $t+1$ is absorbed into timestamp t, there might be insufficient budget left for timestamp $t+1$ to restart step ①. To avoid this issue, the user reserves a portion of ϵ_a as the remaining budget. This reserved proportion is calculated based on the last fitting error and recycled budget from the k involved timestamps.

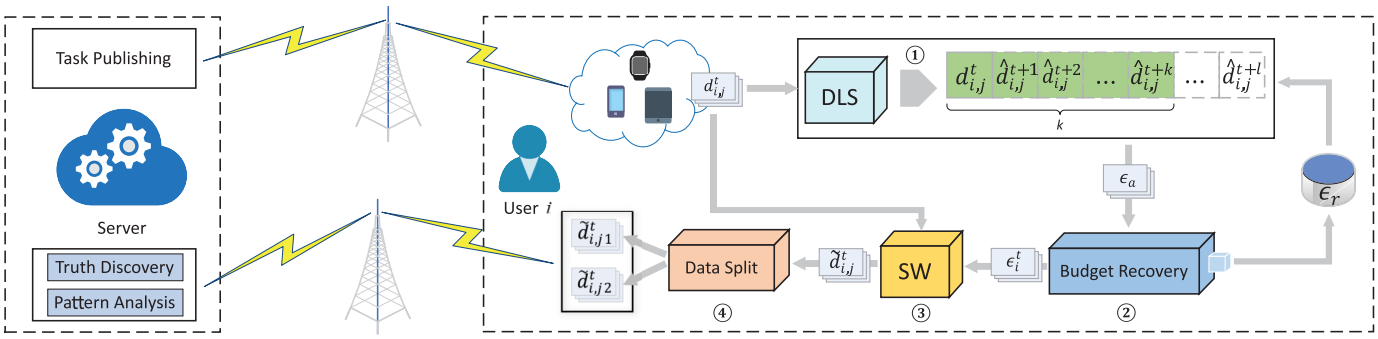


Fig. 1. Overview of user i at timestamp t in PPMCS.

The remaining portion, $\epsilon^t_{i,j}$, is then used as the cost budget for the current moment.

③: Taking $\epsilon^t_{i,j}$ and $d^t_{i,j}$ as input, the user runs the SW mechanism to obtain the perturbed data $\widetilde{d}^t_{i,j}$.

④: A data stream consisting of perturbed data can lose its original pattern features. To preserve these patterns, the user splits $\widetilde{d}^t_{i,j}$ into two parts. The first part (i.e., $\widetilde{d}^t_{i,j1}$) is a randomized output based on the original pattern, with its randomization interval constrained by the piecewise linear approximation method. The second part (i.e., $\widetilde{d}^t_{i,j2}$) is used to recover $\widetilde{d}^t_{i,j1}$ to $\widetilde{d}^t_{i,j}$ for truth discovery. Finally, the user publishes both $\widetilde{d}^t_{i,j1}$ and $\widetilde{d}^t_{i,j2}$ to the server.

In general, the truth discovery and pattern analysis algorithms executed by the server can include any existing algorithms. The primary objective is to extract valuable information from the published data to enable high-quality services.

B. Budget Allocation

A straightforward budget allocation method that satisfies ω-events privacy is to allocate the same budget $\frac{\epsilon}{\omega}$ to each timestamp. However, this approach often results in low utility when ϵ is small. Observations reveal that, in a uniformly collected data stream, adjacent values are likely to be approximate with high probability. Instead of perturbing each data point independently with a decentralized budget, we can accumulate budgets for these approximate values and allow them to share a common perturbed value. For example, consider the sequence [18.0676, 18.0382, 18.0186, 17.9696], where each point has a budget of 0.1. Assuming an approximation threshold of 0.5, we can say that [18.0676, 18.0382, 18.0186] are approximate. Consequently, we can perturb the first data point with a total budget of 0.3, allowing the second and third points to use the same perturbed value. Intuitively, this method offers higher data utility compared to perturbing each data point individually with a budget of 0.1.

To count the number of proximate data points, one would usually need to traverse the sequence backward. Unfortunately, in real-time data streams, knowing future data in advance is impossible. A feasible approach is to use historical data as a sample to fit future data. The least squares method [20] is commonly used for curve fitting to provide the best estimate. However, it is important to note that data in a time series often exhibit noticeable decay characteristics, meaning that data points further from the current moment are less reliable as references. To address this issue, we introduce a decay factor $\alpha \in (0, 1)$ and propose the Decay-aware Least Squares method (DLS). The optimization problem for DLS is defined as

$$min \sum_{i}^{n} \xi_i (y_i - a * x_i - b)^2, \tag{4}$$

where $\xi_i = \alpha(1-\alpha)^{n-i}$. Correspondingly, a and b are computed as

$$a = \frac{\sum_i^n (x_i \cdot y_i \cdot \xi_i) \cdot \sum_i^n \xi_i - \sum_i^n (x_i \cdot \xi_i) \cdot \sum_i^n (y_i \cdot \xi_i)}{\sum_i^n (x_i^2 \cdot \xi_i) \cdot \sum_i^n \xi_i - \left(\sum_i^n y_i \cdot \xi_i\right)^2},$$
$$b = \frac{\sum_i^n (y_i \cdot \xi_i) - a \cdot \sum_i^n (x_i \cdot \xi_i)}{\sum_i^n \xi_i}. \tag{5}$$

With Eq. (5), at timestamp t, taking the time series as x and the historical data as y, we fit the data at timestamp $(t+1)$ as $\widehat{d}^{t+1}_{i,j}$. After that, using $\widehat{d}^{t+1}_{i,j}$ and the historical data as new inputs, we can get the fitted data $\widehat{d}^{t+2}_{i,j}$. Let $l \in [1, \omega - 1]$ represent the number of iterations, and we obtain the fitted list $\{\widehat{d}^{t+1}_{i,j}, \widehat{d}^{t+2}_{i,j}, \cdots, \widehat{d}^{t+l}_{i,j}\}$. Define β as the threshold, and then we can find a prefix list $\{\widehat{d}^{t+1}_{i,j}, \widehat{d}^{t+2}_{i,j}, \cdots, \widehat{d}^{t+k}_{i,j}\} (0 \le k \le l)$, where any data $\widehat{d}^{t'}_{i,j}$ within the list satisfies $|d^t_{i,j} - \widehat{d}^{t'}_{i,j}| \le \beta$. At this point, we consider the data for the next k timestamps to be approximate to $d^t_{i,j}$ and compute the cumulative budget ϵ_a to be

$$\epsilon_a = \frac{(k+1) \cdot \epsilon_r}{l+1}, \tag{6}$$

where $\epsilon_r = \left(\epsilon - \sum_{o=1}^{\omega-1} \epsilon^{t-o}_{i,j}\right)$. Admittedly, ϵ_a can be directly used as the cost budget $\epsilon^t_{i,j}$ for the current timestamp, but it is risky due to the inevitable fitting errors. One possible scenario is that we observe $|d^t_{i,j} - d^{t'}_{i,j}| > \beta$ at timestamp $t' = t + k' (1 \le k' \le k)$. In such cases, the fitting process needs to be re-executed to select a new value for k. If we allocate the entire ϵ_a to the current moment, there might be insufficient budget remaining, or even none at all, for timestamp t'. To solve this problem, we design a budget reservation method. Specifically, for any data $\widehat{d}^{t'}_{i,j}$ in the prefix list, we initialize its reserved budget $c^{t'}$ as $\epsilon_r/(2 \cdot (l+1))$. After that, we update $c^{t'}$ according to the recycled budget $\epsilon^{t'-\omega}_{i,j}$ to $(c^{t'} - \epsilon^{t'-\omega}_{i,j})$. The exceeded recycled budget is denoted as δ and will be carried over to the next timestamp. For example, suppose the recycled budgets at time t_1 and t_2 are 0.9 and 0.1, respectively, and

$c^{t_1} = c^{t_2} = 0.5$. We first withdraw 0.5 from 0.9 and update $c^{t_1} = 0$ and $\delta^{t_1} = 0.4$. Then, for time t_2, the available budget is $\delta^{t_1} + 0.1 = 0.5$, so we also have $c^{t_2} = 0$. Based on this, we define the reserved budget for the k'-th fitted point as

$$c^{t'} = max\left(0,\ \frac{\epsilon_r}{2(l+1)} - \epsilon_{i,j}^{t'-\omega} - \delta^{t'-1}\right). \tag{7}$$

Additionally, we further adjust the reserved budget based on the fitting error. Generally, a larger fitting error indicates that a more reserved budget is needed. Otherwise, we would prefer to allocate more of the budget to the current time to enhance utility. We use the Proportional-Integral-Derivative (PID) formula [21] with a scaling factor to represent the fitting error

$$\tau = \frac{k}{k'}\left(K_p F_i^{t'} + K_i \frac{\sum_{o=1}^{k'-1} F_i^{t+o}}{k'-1} + K_d (F_i^{t'} - F_i^{t'-1})\right), \tag{8}$$

where $F_i^t = |d_i^t - \widehat{d}_i^t|$ and K_p, K_i, and K_d are three proportional factors. The first part of the formula represents the recent error, the second part represents the integral of the error, and the third part represents the derivative of the error with respect to time. The coefficient k/k' scales the observable error value to the total error of the fitting list, based on the principle that the closer $t_{k'}$ is to t, the greater the overall error. We then use an exponential function as the proportional function and compute $\epsilon_{i,j}^t$ as

$$\epsilon_{i,j}^t = \epsilon_a - (1 - exp(-\tau)) \cdot \sum_{o=1}^{k} c^{t+o}. \tag{9}$$

Furthermore, selecting an appropriate l is crucial. In general, there is no universal optimal value for l in complex application scenarios. To handle this, we initialize $l = 1$ and dynamically adjust it in a linear manner. For example, starting at timestamp t, we check if $|d_i^t - d_i^{t+k'}| \leq \beta (1 \leq k' \leq k)$ holds for the next k timestamps. If this condition does not hold, we consider the fitting process inaccurate and adjust l to $l = max(1, \lfloor k/2 \rfloor)$. In other cases, if $k = l$, we increase l to $l = min(l+1, \omega - 1)$; otherwise, we keep l unchanged. The budget allocation process for user i at timestamp t for task j is described in Algorithm 1.

Algorithm 1 Budget Allocation Algorithm

Require: The start timestamp t_1 and count k_1 of the last budget allocation.
Ensure: The cost budget $\epsilon_{i,j}^t$.

```
Initialize ε_{i,j}^t ← 0;
if t_1 + k_1 ≥ t and |d_{i,j}^{t_1} − d_{i,j}^t| ≤ β then
    return ε_{i,j}^t.
else
    if t_1 + k_1 < t and k_1 == l then
        l ← min(l + 1, ω − 1);
    else if t_1 + k_1 ≥ t then
        l ← max(1, ⌊k_1/2⌋);
    end if
    Initialize ε_a ← 0;
    Initialize c_temp ← 0;
    for i ∈ [1, l] do
        Calculate d̂_{i,j}^{t+i} from Eq. (5);
        if |d_{i,j}^t − d̂_{i,j}^{t+i}| > β then
            break;
        else
            ε_a ← ε_a + ε_r/(l+1);
            Calculate c^{t+i} by using Eq. (7);
            c_temp ← c_temp + c^{t+i};
        end if
    end for
    Calculate τ by using Eq. (8);
    ε_{i,j}^t ← ε_a − (1 − exp(−τ)) · c_temp;
end if
return ε_{i,j}^t.
```

C. Data Perturbation

With $\epsilon_{i,j}^t$, we employ SW to randomize the data. Assuming that the spatial domain of task j is $[d_{low}, d_{up}]$, for any data d, we first normalize it as $d' = (d - d_{low})/(d_{up} - d_{low})$. Then, we compute b, p, and q from the allocated budget $\epsilon_{i,j}^t$ and choose a random value $r \in [0, 1]$ to compute the output value $\widetilde{d'}$ as

$$\widetilde{d'} = \begin{cases} \dfrac{r}{q} - b, & \text{if } 0 \leq r \leq d' \cdot q; \\ \dfrac{r - q \cdot d'}{p} + d' - b, & \text{if} d' \cdot q < r \leq d' \cdot q \\ & +2b \cdot p; \\ \dfrac{r - q \cdot d' - 2b \cdot p}{q} + d' + b, & \text{otherwise.} \end{cases} \tag{10}$$

After that, we map $\widetilde{d'}$ to the original field $\widetilde{d} = d - (d_{up} - d_{low})(d' - \widetilde{d'})$. With these operations, the raw data list $\{d_{i,j}^t\}_{j=1}^N$ of worker i is perturbed into a randomized list $\{\widetilde{d}_{i,j}^t\}_{j=1}^N$.

D. Data Splitting

The SW mechanism effectively protects data privacy but, unfortunately, disrupts patterns, making the published data unusable for pattern analysis. Thus, we propose a pattern-aware data splitting method. This method splits the perturbed data into two parts: one part retains the data patterns for pattern analysis, while the other part is used to recover the perturbed data for truth discovery. Inspired by work [7], we utilize the Piecewise Linear Approximation method to define the pattern. Let l^{t_1,t_2} represents the slope of the line connecting data points d^{t_1} and d^{t_2}, with l_{low} and l_{up} representing the lower and upper bounds of the slope, respectively. Suppose the data points within the time interval $[t_1, t-1]$ belong to pattern P. If the data point d^t at timestamp t satisfies

$$l_{low} \leq l^{t_1,t} \leq l_{up}, \tag{11}$$

we consider d^t to also belong to pattern P. Meanwhile, we compute l_1 as the slope between d^{t_1} and $d^t + \theta$, and l_1 as the slope between d^{t_1} and $d^t + -\theta$, and subsequently update l_{low} and l_{up} as

$$\begin{aligned} l_{low} &= _{max}\ (l_{low}, l_1), \\ l_{up} &= _{min}\ (l_{up}, l_2), \end{aligned} \tag{12}$$

where θ is the tolerable error threshold. Otherwise, if Eq. (11) does not hold, d^{t+1} is regarded as a new starting point. Through this process, the original data stream is segmented into multiple pattern fragments. Our goal is to ensure that analysts obtain the same segmentation over the perturbed data stream.

Concretely, for the perturbed data $\widetilde{d}^t_{i,j}$, we split it into $\widetilde{d}^t_{i,j1}$ and $\widetilde{d}^t_{i,j2}$, which satisfy $\widetilde{d}^t_{i,j1}+\widetilde{d}^t_{i,j2} = \widetilde{d}^t_{i,j}$. A straightforward way is to compute $\widetilde{d}^t_{i,j1}$ based on the scaling relationship of raw data as $(\widetilde{d}^{t-1}_{i,j1} + d^t_{i,j} - d^{t-1}_{i,j})$. However, in this case, just knowing any one of the real data allows one to infer all the others. To avoid this risk, we use randomization to compute $\widetilde{d}^t_{i,j1}$. Similarly, assuming that the data points within the time interval $[t_1, t-1]$ belong to pattern P. Then, we first determine whether $d^t_{i,j}$ also belongs to pattern P using Eq. (11). If so, we ensure that $\widetilde{d}^t_{i,j1}$ and the previous split data also belong to the same pattern, and that the relative trend is preserved. In this case, the slope $\widetilde{l}^{t_1,t}$ between the randomly selected $\widetilde{d}^t_{i,j1}$ and $\widetilde{d}^{t_1}_{i,j1}$ satisfies

$$\widetilde{l}^{t_1,t} \in \begin{cases} \left[\widetilde{l}_{low}, \widetilde{l}^{t_1,t-1}\right), & \text{if } l^{t_1,t-1} > l^{t_1,t}; \\ \left[\widetilde{l}^{t_1,t-1}, \widetilde{l}_{up}\right], & \text{otherwise,} \end{cases} \tag{13}$$

where $\widetilde{l}_{low}$ and $\widetilde{l}_{up}$ are the lower and upper bounds maintained on the released split data.

Algorithm 2 Data Splitting Algorithm

Require: The timestamp t_1 of the last starting point, the upper and lower boundaries l_{up} and l_{low} over the raw data stream, and the upper and lower boundaries $\widetilde{l}_{up}$ and $\widetilde{l}_{low}$ over the splitting data stream.

Ensure: The splitting data $\widetilde{d}^t_{i,j1}$ and $\widetilde{d}^t_{i,j2}$.

```
Calculate the slope l^{t1,t};
if l^{t1,t} < l_low then
    Initialize d ← d̃^{t1}_{i,j} + l̃_low · (t − t1);
    Choose a random d̃^t_{i,j1} from [d − θ, d);
else if l^{t1,t} > l_up then
    Initialize d ← d̃^{t1}_{i,j} + l̃_up · (t − t1);
    Choose a random d̃^t_{i,j1} from [d, d + θ);
else
    Initialize l̃ ← 0;
    Calculate the slopes l^{t1,t−1} and l̃^{t1,t−1};
    if l^{t1,t} > l^{t1,t−1} then
        Choose a random l̃ from [l̃^{t1,t−1}, l̃_up];
    else
        Choose a random l̃ from [l̃_low, l̃^{t1,t−1}];
    end if
    Calculate d̃^t_{i,j1} as d̃^{t1}_{i,j} + l̃ · (t − t1);
end if
Update l_low, l_up, l̃_low, and l̃_up by using Eq. (12);
Calculate d̃^t_{i,j2} as d̃^t_{i,j} − d̃^t_{i,j1};
return d̃^t_{i,j1} and d̃^t_{i,j2}.
```

Conversely, if $d^{t_1}_{i,j}$ and $d^t_{i,j}$ are not within the same pattern, we will select $\widetilde{d}^t_{i,j1}$ from outside the boundaries and regard this point as a new start point. The data splitting process for user i at timestamp t for task j is shown in Algorithm 2. Ultimately, the perturbed data list $\{\widetilde{d}^t_{i,j}\}^N_{j=1}$ of worker i is split into $\{\widetilde{d}^t_{i,j1}\}^N_{j=1}$ and $\{\widetilde{d}^t_{i,j2}\}^N_{j=1}$.

IV. Theoretical Analysis

In this section, we provide theoretical analyses of privacy and efficiency.

A. Privacy Analysis

Theorem 1 (Sequential Composition [22]): Let $\mathcal{M}$ be a combinatorial mechanism that performs $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_k$ independently in turn, where $\mathcal{M}_i$ satisfies ϵ_i-differential privacy, then we can say $\mathcal{M}$ satisfies $\sum_{i=1}^k \epsilon_i$-differential privacy.

Theorem 2 (Parallel Composition [22]): Let D_1, D_2, $\cdots$, D_k be subsets of the dataset D that satisfy $\cup_{i=1}^k D_i = D$ and $D_i \cap D_j = \emptyset (\forall i \neq j)$, and $\mathcal{M}$ be a combinatorial mechanism that executes $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_k$ in parallel, where $\mathcal{M}_i$ satisfies ϵ_i-differential privacy and takes D_i as input, then we can say that $\mathcal{M}(D)$ satisfies $max\{\epsilon_1, \epsilon_2, \cdots, \epsilon_k\}$-differential privacy.

Theorem 3 (Post Processing [23]): Let $\mathcal{M}$ be a mechanism that satisfies ϵ-differential privacy, and f denote an arbitrary randomized mapping function, then $f \circ \mathcal{M}$ also satisfies ϵ-differential privacy. That is, post-processing operations on the output do not result in any privacy loss.

Theorem 4: Our proposed PPMCS scheme satisfies $\omega-$ event privacy.

Proof: First, we analyze the security of the data published by user i for task j at timestamp t. Our scheme employs SW as the randomization mechanism. Without loss of generality, SW can be defined as a family of probability density functions distributed over the output domain, where the probability density function of any mapping $v \to \widetilde{v}$ can be expressed as $M_v(\widetilde{v}) = Pr[\Psi(v) = \widetilde{v}]$. For any possible two values (v_1, v_2) in the input domain and any possible set T in the output domain, we have

$$\frac{Pr[SW(v_1) \in T]}{Pr[SW(v_2) \in T]} = \frac{\int_{\widetilde{v}\in T} M_{v_1}(\widetilde{v})d\widetilde{v}}{\int_{\widetilde{v}\in T} M_{v_2}(\widetilde{v})d\widetilde{v}} \leq \frac{\int_{\widetilde{v}\in T} p\, d\widetilde{v}}{\int_{\widetilde{v}\in T} q\, d\widetilde{v}} = e^{\epsilon}. \tag{14}$$

This formula proves that the randomized output of data $d^t_{i,j}$ with a privacy budget of $\epsilon^t_{i,j}$ satisfies $\epsilon^t_{i,j}$-differential privacy. To retain the data pattern, the randomized output will be further divided into two parts. According to Theorem 3, post-processing operations on the output do not cause additional privacy loss, so the final published data of task j at timestamp t still satisfies $\epsilon^t_{i,j}$-differential privacy.

Then, for task j, we define its SW mechanisms used within any window of w timestamps as $\mathcal{M}_1$, $\mathcal{M}_2$, $\cdots$, $\mathcal{M}_\omega$. According to Theorem 1, the privacy guarantee of mechanisms $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_\omega$ is equivalent to a combinatorial mechanism $\mathcal{M}_s$ that executes them sequentially. For any two neighboring datasets D and D', and all possible outputs $T \in Range(\mathcal{M}_s)$, it holds that

$$\begin{aligned} &Pr[\mathcal{M}_s(D) \in T] \\ &\quad = Pr[(\mathcal{M}_1(D), \mathcal{M}_2(D), \cdots, \mathcal{M}_\omega(D)) \in T] \\ &\quad = \sum_{T_t \in T} \prod_{t}^{\omega} Pr[\mathcal{M}_t(D) = T_t] \end{aligned}$$

$$\leq \sum_{T_t \in T} \prod_{t}^{\omega} e^{\epsilon_{i,j}^t} Pr[\mathcal{M}_t(D') = T_t]$$
$$= e^{\sum_{t=1}^{\omega} \epsilon_{i,j}^t} \sum_{T_t \in T} \prod_{t}^{\omega} Pr[\mathcal{M}_t(D') = T_t]$$
$$= e^{\sum_{t=1}^{\omega} \epsilon_{i,j}^t} Pr[\mathcal{M}_s(D') \in T]. \quad (15)$$

In our budget allocation algorithm, we strictly ensure that the total consumption budget for any consecutive ω timestamps does not exceed ϵ, i.e., $\sum_{t=1}^{\omega} \epsilon_{i,j}^t \leq \epsilon$. Combined with Eq. (15), we can say that the perturbation mechanisms of task j within any window of w timestamps satisfy ϵ-differential privacy.

Once again, by leveraging Theorem 1, we represent the perturbation mechanisms for all tasks within any window of ω timestamps as $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_q$, where $q = M \times N$ and $\mathcal{M}_j$ ($j \in [1, q]$) denotes the sequential composition mechanism for task j. Then, according to Theorem 2, $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_q$ can be reduced to a parallel combinatorial mechanism $\mathcal{M}_p$. Let D_j denote the dataset of mechanism $\mathcal{M}_j$, $D = \bigcup_{j=1}^{q} D_j$ be the union, and D' be an adjacent dataset differing from D in the k-th subset. For any possible output set $T \in Range(\mathcal{M}_p)$, there exists

$$Pr[\mathcal{M}_p(D) \in T]$$
$$= Pr[(\mathcal{M}_1(D_1), \mathcal{M}_2(D_2), \cdots, \mathcal{M}_q(D_q)) \in T]$$
$$= Pr[\mathcal{M}_k(D_k) \in T_k] \prod_{j \neq k}^{q} Pr[\mathcal{M}_j(D_j) \in T_j]$$
$$\leq e^{\epsilon_k} Pr[\mathcal{M}_k(D'_k) \in T_k] \prod_{j \neq k}^{q} Pr[\mathcal{M}_j(D_j) \in T_j]$$
$$= e^{\epsilon_k} Pr[(\mathcal{M}_1(D_1), \cdots, \mathcal{M}_k(D'_k), \cdots, \mathcal{M}_q(D_q)) \in T]$$
$$= e^{\epsilon_k} Pr[\mathcal{M}_p(D') \in T]$$
$$\leq e^{max(\epsilon_1, \epsilon_2, \cdots, \epsilon_q)} Pr[\mathcal{M}_p(D') \in T]. \quad (16)$$

Based on the above analysis, we have $\epsilon_1 = \epsilon_2 = \cdots = \epsilon_q = \epsilon$. This means that $\mathcal{M}_p$ satisfies ϵ-differential privacy as well. Consequently, we can say that the perturbation mechanisms applied to all tasks within any window of ω timestamps satisfy ϵ-differential privacy, which aligns with the definition of ω-event ϵ-differential privacy. Therefore, we conclude that our scheme satisfies ω-event privacy.

B. Efficiency Analysis

The operations on any sensing data $d_{i,j}^t$ primarily consist of budget allocation, data perturbation, and data splitting. During budget allocation, the user first fits the next l data points, where each fitting step takes historical samples as input. Thus, this part has a time complexity of $O(lT)$. Then, based on the fitted list and the PID error, the user determines the privacy budget for the current timestamp from the remaining budget, with a time complexity of $O(k)$. Notably, k ranges from 0 to l, while l is dynamically adjusted within the range from 1 to ω. As a result, the overall time complexity of budget allocation can be expressed as $O(\omega T)$. For data perturbation and data splitting, since no loops are involved, their time complexity can be considered constant. Particularly, if the previously perturbed data remains available for the current timestamp, the budget allocation and data perturbation steps can be skipped. Since each user processes sensing data for N tasks simultaneously, the time complexity at a given timestamp t is $O(\omega NT)$, while the total time complexity over the entire timeline is $O(\omega NT^2)$. Furthermore, in our experiments, for simplicity, we simulate each user using a sequential processing approach, leading to an experimental time complexity of $O(\omega NMT^2)$. However, in practical applications, as users operate independently, the overall time complexity remains $O(\omega NT^2)$.

V. Experiments

In this section, we evaluate our PPMCS on multiple datasets and provide a comprehensive comparative analysis with existing works.

A. Experiment Settings

1) Datasets: We construct experimental datasets based on four real-world datasets. The details are as follows.

- Intel Lab Data.[1] The Intel Lab Data (abbreviated as Intel) dataset consists of data collected by 54 sensors at the Intel Berkeley Research Lab from February 28th to April 5th. We extract a subset from this dataset as our experimental dataset, which contains the temperature and humidity collected by 27 sensors at 851 timestamps.
- Weather.[2] The Weather dataset comprises weather data for 30 major U.S. cities as reported by 18 websites in March 2010. Excluding one invalid website, we select the temperature for 23 cities from 17 websites at 144 timestamps as our experimental dataset.
- Stock.[3] The Stock dataset records market data for 505 stocks from 2013 through 2018. Restricting the data stream to the same length, we obtain our experimental dataset containing the opening prices of 468 stocks at 1259 timestamps.
- PEMS08.[4] The PEMS08 dataset contains traffic data collected at 5-minute intervals over a 62-day period from 170 detectors in California. Each probe packet includes three dimensions: flow rate, average speed, and average occupancy. We intercept the first 1000 timestamps as our experimental dataset.

2) Metrics: We evaluate the utility of truth discovery using Mean Absolution Error (MAE) and Mean Relative Error (MRE), which are formulated as

$$MAE(Z, \widetilde{Z}) = \frac{1}{T \cdot N} \sum_{t=1}^{T} \sum_{j=1}^{N} |z_j^t - \widetilde{z}_j^t|,$$
$$MRE(Z, \widetilde{Z}) = \frac{1}{T \cdot N} \sum_{t=1}^{T} \sum_{j=1}^{N} \frac{|z_j^t - \widetilde{z}_j^t|}{max(z_j^t, \gamma)}, \quad (17)$$

[1] https://db.csail.mit.edu/labdata/labdata.html
[2] https://lunadong.com/fusionDataSets.htm
[3] https://www.kaggle.com/datasets/camnugent/sandp500
[4] https://www.kaggle.com/datasets/elmahy/pems-dataset

where z_j^t and $\widetilde{z}_j^t$ denote the truth discovery results with and without differential privacy, respectively, and γ is a constraint to mitigate the effects when values are too small. To demonstrate the adaptability of our scheme, we will use two truth discovery algorithms, namely CRH [24] and CATD [25], to compute $\widetilde{z}_j^t$, respectively. When discussing patterns, we are more concerned with the trends in data rather than the data values themselves. Therefore, we use the Mean Absolute Difference Error (MADE) as a utility metric for pattern analysis. The formula for MADE is

$$MADE(D, \widetilde{D}) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{t=2}^{T}|(d_{i,j}^t - d_{i,j}^{t-1}) - (\widetilde{d}_{i,j}^t - \widetilde{d}_{i,j}^{t-1})|}{M \cdot N \cdot (T-1)}. \quad (18)$$

To quantify the extent of real data leakage, we design a simple data inference method. Let $S = \{d_1, d_2, \cdots, d_n\}$ denote a perturbed data sequence, and DF_i denote the first-order difference between two neighboring numbers, i.e., $DF_i = d_i - d_{i-1}$. We identify all remarkable points with the following characteristics

$$\begin{cases} DF_i > 0 \text{ and } DF_{i-1} < 0 \\ DF_i < 0 \text{ and } DF_{i-1} > 0, \end{cases} \quad (19)$$

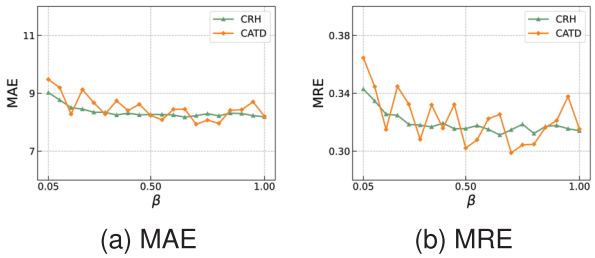
and then mark all other data as true. By comparing these with the raw data sequence, we obtain the Correct Labeling Percentage (CLP). Intuitively, the larger the CLP value, the lower the privacy guarantees.

3) Comparison: To validate the effectiveness of our budget allocation and data splitting methods, we design two variants of PPMCS, called PPMCS* and PPMCS†. In PPMCS*, we allocate the budget equally, with each data perturbed with budget $\frac{\epsilon}{\omega}$ at every timestamp. In PPMCS†, we publish the perturbed data without splitting. In addition, we also compare PPMCS with BA [10], PPPTD [14], PriSTD [13], PriPTD [15], PatternLDP [7], PPLDP [16], and ASRT [17]. In these schemes, BA, PPPTD, PriPTD, and PriSTD use the Laplace mechanism for data perturbation, whereas ASRT employs the Bounded Laplace mechanism, and PatternLDP and PPLDP adopt the randomization mechanism. Additionally, PPPTD is designed to satisfy user-level differential privacy. For fairness, we impose an upper bound on its budget to ensure that all schemes adhere to ω-event privacy.

All programs are implemented by Python3. For more details, please refer to our GitHub repository.[5] To avoid chance, all results are averages of data from 100 experiments.

B. Experiment Analysis

In this part, we analyze the effect of parameters α, β, ϵ, ω, and θ on PPMCS across different datasets. The first four parameters influence to the utility of truth discovery, while the last one pertains to the utility in pattern analysis. Following that, we compare the performance of PPMCS with that of existing schemes. Considering that fitting based on a small sample size introduces significant bias, we designate the first 10 timestamps as a preparation phase, during which the budget of each timestamp is set to $\frac{\epsilon}{\omega}$. Additionally, following previous works, we set $K_p = 0.8$, $K_i = 0.1$, and $K_d = 0.1$.



(a) MAE (b) MRE

Fig. 2. The (a) MAE and (b) MRE metrics for β on the Intel dataset under CRH and CATD algorithms.

1) Effect of α: We evaluate the average fitting error of the Decay-aware Least Squares (DLS) method for different values of α across four datasets. The parameter α is varied incrementally from 0.1 to 0.9. For the n-th data, we use the first $n-1$ data as input for fitting. We also apply the same procedure to the classical Least Squares (LS) method for comparison. The results for both methods are shown in Table III, from which we can clearly see that (1) DLS significantly improves fitting accuracy compared to LS. In the following experiments, we use the optimal values of α for each dataset, as highlighted in bold in Table III.

2) Effect of β: Since the data space size $|D|$ differs across tasks, using the same β value for all tasks is inappropriate. Thus, we define β as a percentage, and the approximation threshold is calculated as $|D| \cdot \beta$. Fig. 2 illustrates the effect of β ranging from 0.05 to 1 on the Intel dataset, with $\epsilon = 1$ and $\omega = 50$. From the results, we first observe that the trends of MAE and MRE are similar, and the fluctuations in MAE and MRE for the CATD algorithm are larger than those for the CRH algorithm. Furthermore, we note that MAE and MRE do not show a strong correlation with β. This indicates that β does not have a universally optimal value across datasets. In subsequent experiments, for the Intel dataset, we set β to the best observed value from the results, which is 0.65 for CRH and 0.7 for CATD.

We also test the effect of β on the remaining three datasets. Due to space limitations, we do not present the detailed data here. As a result, the optimal values of β for the Weather, Stock, and PEMS08 datasets under the CRH algorithm are 0.7, 0.8, and 0.6, respectively, while for the CATD algorithm, they are 0.8, 0.35, and 0.3, respectively.

3) Effect of ϵ: We examine the utility effects of ϵ within the range [0.05, 1], with ω set to 50 and ϵ increasing by 0.05 each time. To highlight the impact of the budget allocation algorithm, we also test PPMCS* under the same settings. Fig. 3 presents the MAE results under the CRH algorithm across four datasets. From these data, we can learn that (1) PPMCS exhibits a significant utility improvement over PPMCS*, indicating that our budget allocation algorithm plays a crucial role in enhancing the utility of truth discovery compared to a simple uniform allocation approach. Furthermore, we observe that (2) the utility of both schemes improves with increasing ϵ. This reflects the nature of differential privacy: a larger budget allows for higher utility but lower privacy

[5]https://github.com/javadazhancjiajia/ppmcs.git

TABLE III
COMPARISON OF AVERAGE FITTING ERRORS FOR LS AND DLS

Dataset	LS	DLS (α=)								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Intel	3.01	0.395	0.254	0.215	0.202	**0.2**	0.206	0.217	0.234	0.256
Weather	4.802	3.652	3.042	2.581	2.355	**2.28**	2.316	2.418	2.571	2.786
Stock	7.472	1.546	1.238	**1.156**	1.16	1.215	1.311	1.443	1.615	1.834
PEMS08	28.1	5.455	**5.09**	5.376	5.863	6.534	7.41	8.54	9.978	11.821

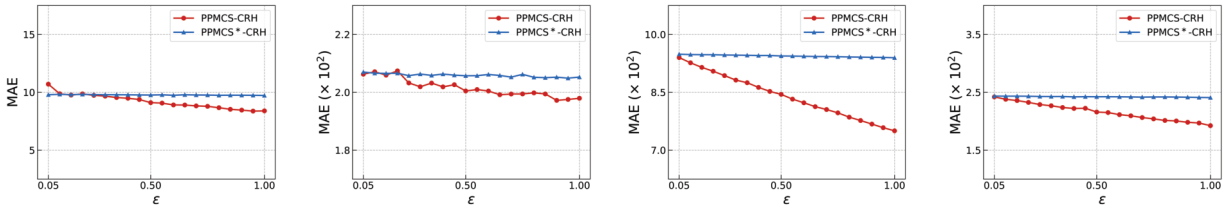


Fig. 3. The MAE metric for different values of ϵ under the CRH algorithm across four datasets.

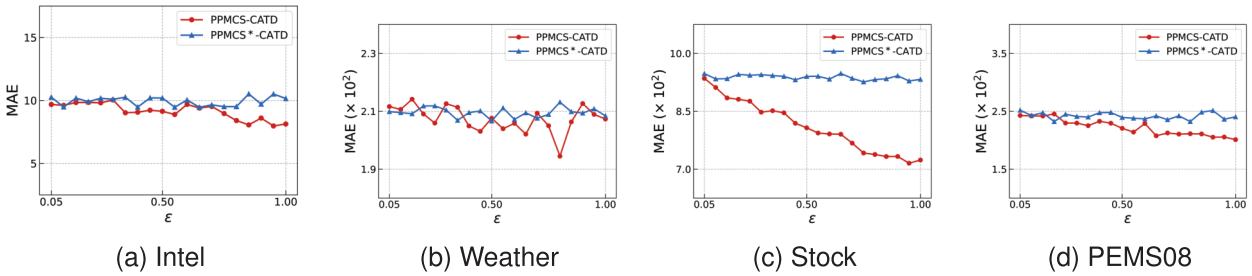


Fig. 4. The MAE metric for different values of ϵ under the CATD algorithm across four datasets.

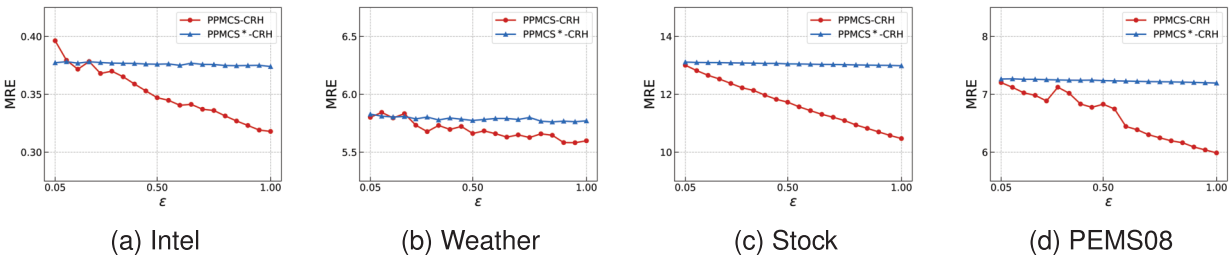


Fig. 5. The MRE metric for different values of ϵ under the CRH algorithm across four datasets.

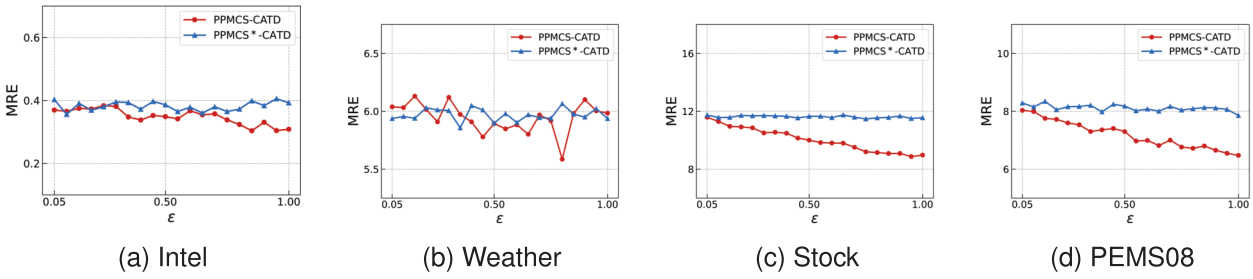


Fig. 6. The MRE metric for different values of ϵ under the CATD algorithm across four datasets.

guarantees. A similar experiment is conducted under the CATD algorithm, with the results shown in Fig. 4, where PPMCS also maintains a clear advantage. Figs. 5 and 6 illustrate the MRE results under CRH and CATD, respectively,

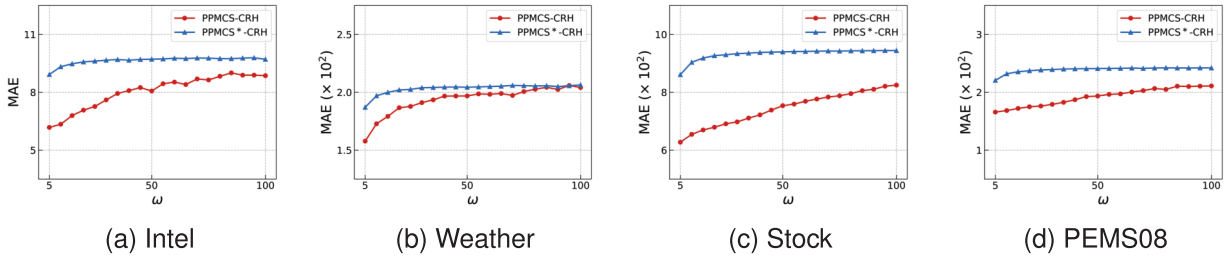


Fig. 7. The MAE metric for different values of ω under the CRH algorithm across four datasets.

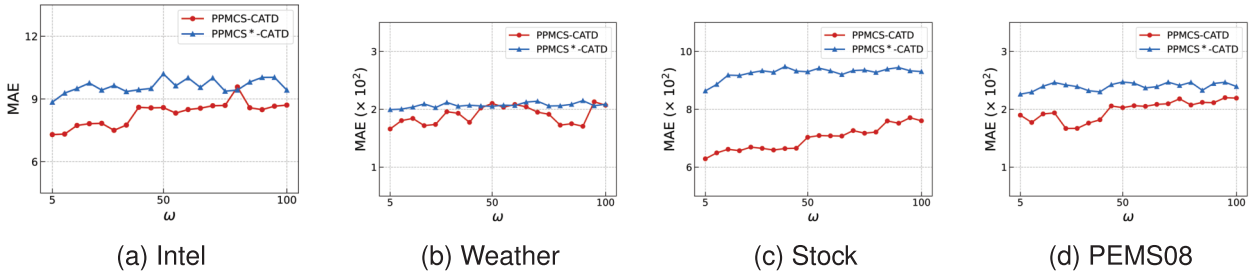


Fig. 8. The MAE metric for different values of ω under the CATD algorithm across four datasets.

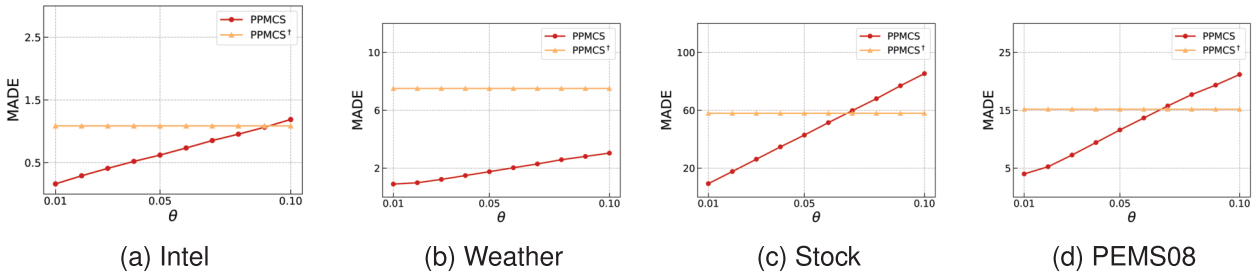


Fig. 9. The MADE metric for different values of θ across four datasets.

showing trends and conclusions consistent with those of MAE. In the rest of experiments, we set $\epsilon = 1$ unless otherwise specified.

4) Effect of ω: According to the definition of ω-event privacy, the parameter ω determines the size of the event window requiring continuous protection and also constrains the maximum value of l in our scheme. Here, we test the effects of ω on four datasets, varying ω from 5 to 100 in increments of 5. Again, PPMCS* is used as a baseline for comparison. Figs. 7 and 8 depict the variations in the MAE metric under the CRH and CATD algorithms, respectively. From these figures, we can see that (1) PPMCS repeatedly outperforms PPMCS*, especially when ω is smaller. Theoretically, ϵ and ω have opposing effects on the trade-off between utility and privacy. When ϵ is fixed, a larger ω reduces the average budget available per timestamp, thereby lowering utility. Since the MRE metric still follows a similar pattern to the MAE metric, its results are omitted for brevity. In the subsequent experiments, we set $\omega = 50$.

5) Effect of θ: Similar to β, we define θ as a percentage to account for task variability and use it to calculate the slope tolerance error as $\theta \times C$, where C represents the maximum difference between adjacent values. The experimental range for θ is set between [0.01, 0.1]. We use PPMCS† as a benchmark to highlight the data splitting method. The final results are presented in Fig. 9. Notably, since PPMCS† is independent of θ, its MADE results form a constant line across all four datasets. From these curves, we observe that (1) our PPMCS achieves better MADE results than PPMCS† at smaller values of θ. This is because a smaller θ indicates a smaller randomization interval, which helps to avoid abrupt changes in the split data. However, it also increases the risk that an attacker with knowledge of any real data could infer a curve closer to the original, based on scaling relationships. For the rest of experiments, we fix $\theta = 0.01$.

TABLE IV
AVERAGE FITTING ERROR ON DLS

	Intel	Weather	Stock	PEMS08
PPMCS†	17.8	32.22	35.8	30.48
PPMCS	**10.54**	**12.65**	**13.28**	**17.74**

Furthermore, to provide a more intuitive demonstration of the improvement in pattern analysis accuracy achieved by our method, we apply the DLS algorithm to the perturbed data generated by PPMCS and PPMCS† for data fitting. Table IV presents the average error between the fitting results and the

TABLE V
COMPARISON OF SECURITY

	Intel			Weather			Stock			PEMS08		
	CLP	MAE	MADE	CLP	MAE	MADE	CLP	MAE	MADE	CLP	MAE	MADE
PatternLDP	92.13%	0.0076	0.00068	98.08%	0.203	0.022	98.01%	0.0016	0.00014	89.3%	0.0025	0.0011
PPLDP	92.12%	0.0011	0.0014	96.15%	6.41	0.66	98%	0.45	1.01	89.5%	0.54	0.43
ASRT	43.6%	0.22	0.59	62.5%	5.16	0.96	43.53%	5.58	10.73	65.2%	1.84	3.26
PPMCS	**0%**	8.39	0.16	**0%**	197.93	0.9	**0%**	750.13	9.29	**0%**	192.33	4

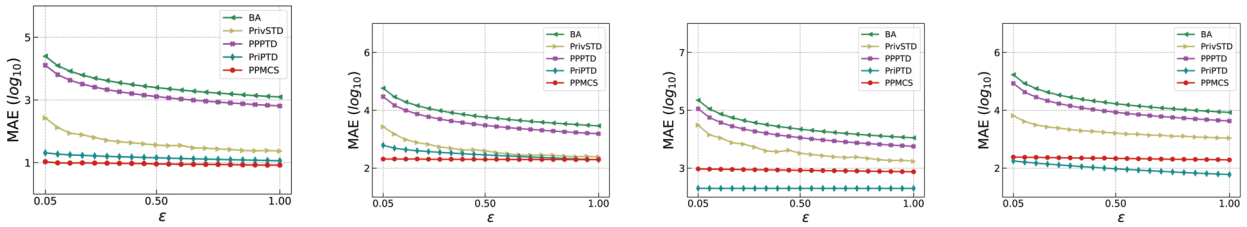


Fig. 10. The MAE metric of ϵ under different schemes across four datasets. In particular, all data are results from a logarithmic transformation log_{10}.

TABLE VI
COMPARISON OF UTILITY

	Intel			Weather			Stock			PEMS08		
	MAE	MRE	MADE	MAE	MRE	MADE	MAE	MRE	MADE	MAE	MRE	MADE
BA	1244.84	43.32	407.09	2900.04	87.91	691.22	11157.97	154.01	3055.67	8452.33	170	4707.55
PPPTD	645.16	22.47	207.9	1546.14	47.5	368.78	5670.64	78.43	1540.16	4281.53	86.73	2371.04
PrivSTD	23.44	0.87	3.52	242.59	7.49	61.11	1720.13	28.43	27.61	1095.84	68.91	99.05
PriPTD	11.55	0.45	75.36	199.87	5.85	75.45	**46.01**	**0.8**	75.4	**60.16**	711.16	75.36
PPMCS	**8.39**	**0.37**	**0.16**	**197.93**	**5.6**	**0.9**	750.13	10.47	**9.29**	192.33	**5.99**	**4**

real data. Evidently, our PPMCS yields more accurate fitting results, thereby validating the effectiveness of our approach.

6) Comparison: Since that PPPTD and PriPTD are specifically designed for the CRH algorithm, all MAE and MRE metrics in this part are computed under the CRH algorithm. We first compare our PPMCS with PatternLDP, PPLDP, and ASRT in Table V, where CIP takes the maximum value among the $M \times N$ data streams. As previously mentioned, a common characteristic of these three schemes is that the released data contains a certain proportion of raw data. Undoubtedly, this results in high data utility, as reflected by the MAE metric in the table. However, they are not secure. The CLP indicates that an attacker could interpret the published data as near-real datasets with minor anomalies, potentially compromising user privacy, especially in PatternLDP and PPLDP. Although ASRT reduces data leakage, it also substantially diminishes utility in pattern analysis, as reflected in the MADE metric, which is comparable to our proposed PPMCS. In summary, we conclude that (1) our PPMCS provides better privacy guarantees than existing high-utility schemes.

Next, we compare our PPMCS with BA, PrivSTD, PPPTP, and PriPTD. All these schemes provide the same theoretical privacy guarantees; therefore, we focus on evaluating their utility in truth discovery and pattern analysis. Fig. 10 illustrates the variation in MAE for different schemes as parameter ϵ increases from 0.05 to 1. To better represent data with a wide range of values, we apply a logarithmic transformation log_{10}. The results demonstrate that our PPMCS achieves the best performance on the Intel and Weather datasets and the second-best performance on the Stock and PEMS08 datasets. To further clarify, Table VI presents the results of these schemes under $\epsilon = 1$, including MAE, MRE, and MADE metrics. The data reveal that our scheme consistently achieves the highest utility in terms of MAD and MRE in most cases. Furthermore, in terms of MDRE, our approach consistently outperforms all other schemes. These findings confirm that (2) our scheme provides superior data utility than existing schemes while maintaining the same level of privacy protection. In other words, the PPMCS achieves a better trade-off between privacy and utility.

VI. RELATED WORK

A. Privacy in Truth Discovery

The truth discovery problem was first formalized by Yun et al. [26], in a scenario of choosing more trustworthy websites than authority-based search engines. Numerous

studies have since addressed this problem, proposing various iterative algorithms for different application scenarios [5], [6], [27]. As privacy concerns have become increasingly prominent, researchers have turned their attention to privacy-preserving truth discovery. Depending on the privacy protection technique employed, existing works can be broadly categorized into two groups: (1) encryption-based schemes and (2) differential privacy-based schemes.

In encryption-based schemes [28], [29], [30], [31], users typically encrypt data locally and submit only the ciphertext, while the server runs a specified truth discovery algorithm using the ciphertext as input. However, cryptographic operations often introduce costly computation overhead, making their schemes less practical for real-time and large-scale data applications. Compared to cryptography, differential privacy offers robust security with negligible additional overhead, making it more widely adopted. In early studies on differential privacy [18], [32], raw data were centrally perturbed by a trusted server. However, constructing a fully trustworthy entity proved to be a significant challenge. As an alternative, local differential privacy was introduced to eliminate the need for a central server by requiring users to locally perturb their data [33], [34]. With this goal, Li et al. [11] proposed a two-layer mechanism that allows users to sample their own probabilities from a hyperdistribution to randomize their reported answers. To reduce utility loss, they then extended their work using the Gaussian mechanism [35]. Sun et al. [12] highlighted the sparsity of uploaded data, where most workers only answered a small subset of tasks. To address this, they designed an efficient matrix factorization algorithm using the Laplace mechanism for truth inference. Wang et al. [13] first explored the privacy concerns of streaming truth discovery. They proposed a budget recycling mechanism that allows only a subset of points to consume a limited budget for perturbation, while other points reuse the previously perturbed values. Pang et al. [14] also focused on data streams and proposed a personalized privacy-preserving framework. In their scheme, users initialized and adjusted their privacy budgets based on individual needs. Additionally, Zhang et al. [36] devised a scheme with solid privacy and utility guarantees. Users submitted only partially perturbed data, while the server employed matrix factorization models to infer the remaining data before conducting truth discovery. In recent work, Gong et al. [15] proposed an outlier-aware privacy-preserving scheme that evaluates user credibility weights by analyzing data fluctuations and allocates the limited privacy budget to more critical data points. Furthermore, they devised a noise-aware error adjustment method to mitigate the bias introduced by noise. However, none of these schemes take into account the preservation of data patterns. The original temporal patterns are disrupted after perturbation, preventing the server from performing effective pattern analysis on the released data.

B. Privacy in Pattern Analysis

Patterns are defined as the temporal correlations among data points in a time series. Early studies on differentially private time-series data primarily focused on statistical analysis, such as mean estimation [37], [38] and frequency estimation [39], [40]. These approaches typically perturb each data point independently, without considering the preservation of patterns. Wang et al. [7] were the first to identify this limitation and proposed a pattern-aware privacy-preserving data collection framework. They introduced an importance-aware remarkable point sampling method to capture pattern features and applied a randomization mechanism to perturb these remarkable points, while unselected data points were uploaded in their original form. Gao and Zhou [16] improved this framework. They leveraged both the fluctuation trend and the fluctuation speed of real-time data streams to determine whether a data point could accurately represent the underlying pattern. The selected points were perturbed using a randomized mechanism and then refined via Kalman filtering, while the remaining points were directly released without perturbation. Li et al. [17] designed a sampling method based on the randomized response mechanism to capture sensitive patterns in time-series data for mobile crowdsensing. Additionally, they applied the Bounded Laplace mechanism to ensure that the perturbed data values remained within the valid range. However, all three schemes offer weak security, as a certain proportion of the data remains unperturbed. How to perturb all data points while still preserving the data patterns remains an open problem worth exploring.

In addition to the aforementioned works, Ye et al. [41] attempted to preserve temporal correlations by perturbing the time sequence rather than the data values. Subsequently, they improved their scheme with a two-way atomic operation, which can eliminate missing, empty, or repeated values in the released time series [42]. To extract frequent shapes from perturbed data, Mao et al. [43] considered using time-series transformation techniques to preserve data patterns. However, these methods are not designed for real-time release, and therefore cannot be directly applied to mobile crowdsensing.

VII. Conclusion

In mobile crowdsensing, to prevent untrustworthy entities from learning sensitive information embedded in the data stream, we employ the Square Wave mechanism to perturb the raw data. However, this results in a reduction in data utility, leading to lower accuracy in truth discovery and pattern analysis on the perturbed datasets. To mitigate this issue, we design a budget allocation algorithm to reduce the utility loss and propose a data splitting method to preserve patterns. Theoretical analysis proves that our scheme satisfies ω-event privacy. Extensive experiments based on four real-world datasets show that, while maintaining strong privacy guarantees, our scheme produces more accurate results in truth discovery and pattern analysis compared to existing schemes.

We note that despite the significant improvement in the accuracy of aggregated results, they still fall short of meeting the requirements of practical applications, especially in scenarios where data changes lack clear patterns. Moving forward, we aim to investigate more efficient approaches to further reduce utility loss in numerical computations.

REFERENCES

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, Nov. 2011.

[2] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2419–2465, 3rd Quart., 2019.

[3] R. K. Ganti, N. Pham, H. Ahmadi, S. Nangia, and T. F. Abdelzaher, "GreenGPS: A participatory sensing fuel-efficient maps application," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, Jun. 2010, pp. 151–164.

[4] Y. Cheng et al., "AirCloud: A cloud-based air-quality monitoring system for everyone," in *Proc. 12th ACM Conf. Embedded Netw. Sensor Syst.*, New York, NY, USA, 2014, pp. 251–265.

[5] H. Xiao et al., "Towards confidence in the truth: A bootstrapping based truth discovery approach," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Min.*, 2016, pp. 1935–1944.

[6] P. Wang, D. Jiao, L. Yang, B. Wang, and R. Yu, "Hypergraph-based truth discovery for sparse data in mobile crowdsensing," *ACM Trans. Sensor Netw.*, vol. 20, no. 3, pp. 1–23, May 2024.

[7] Z. Wang, W. Liu, X. Pang, J. Ren, Z. Liu, and Y. Chen, "Towards pattern-aware privacy-preserving real-time data collection," in *Proc. 39th IEEE Conf. Comput. Commun. INFOCOM*, Jul. 2020, pp. 109–118.

[8] Z. Wang et al., "When mobile crowdsensing meets privacy," *IEEE Commun. Mag.*, vol. 57, no. 9, pp. 72–78, Sep. 2019.

[9] D. Chen, P. Bovornkeeratiroj, D. Irwin, and P. Shenoy, "Private memoirs of IoT devices: Safeguarding user privacy in the IoT era," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 1327–1336.

[10] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias, "Differentially private event sequences over infinite streams," *Proc. VLDB Endow.*, vol. 7, no. 12, pp. 1155–1166, Aug. 2014.

[11] Y. Li et al., "An efficient two-layer mechanism for privacy-preserving truth discovery," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1705–1714.

[12] H. Sun, B. Dong, H. Wang, T. Yu, and Z. Qin, "Truth inference on sparse crowdsourcing data with local differential privacy," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 488–497.

[13] D. Wang, J. Ren, Z. Wang, X. Pang, Y. Zhang, and X. Shen, "Privacy-preserving streaming truth discovery in crowdsourcing with differential privacy," *IEEE Trans. Mobile Comput.*, vol. 21, no. 10, pp. 3757–3772, Oct. 2022.

[14] X. Pang, Z. Wang, D. Liu, J. C. S. Lui, Q. Wang, and J. Ren, "Towards personalized privacy-preserving truth discovery over crowdsourced data streams," *IEEE/ACM Trans. Netw.*, vol. 30, no. 1, pp. 327–340, Feb. 2022.

[15] Z. Gong, Z. Yang, S. Yang, S. Yu, K. Li, and M. Duan, "Towards accurate truth discovery with privacy-preserving over crowdsourced data streams," *IEEE Trans. Knowl. Data Eng.*, vol. 37, no. 4, pp. 2155–2168, Apr. 2025.

[16] W. Gao and S. Zhou, "Privacy-preserving for dynamic real-time published data streams based on local differential privacy," *IEEE Internet Things J.*, vol. 11, no. 8, pp. 13551–13562, Apr. 2024.

[17] Z. Li, X. Zeng, Y. Xiao, C. Li, W. Wu, and H. Liu, "Pattern-sensitive local differential privacy for finite-range time-series data in mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 1, pp. 1–14, Jan. 2025.

[18] C. Dwork, "Differential privacy," in *Proc. 33rd Int. Colloq. Automata, Lang., Program.* Cham, Switzerland: Springer, 2006, pp. 1–12.

[19] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škoric, "Estimating numerical distributions under local differential privacy," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2020, pp. 621–635.

[20] D. Eberly. (2000). *Least Squares Fitting of Data*. [Online]. Available: https://ncorr.com/download/publications/eberlyleastsquares.pdf

[21] M. King, *Process Control: A Practical Approach*. Hoboken, NJ, USA: Wiley, 2010.

[22] F. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2009, pp. 19–30.

[23] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.

[24] Q. Li et al., "A confidence-aware approach for truth discovery on long-tail data," *Proc. VLDB Endow.*, vol. 8, no. 4, pp. 425–436, 2014.

[25] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han, "Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2014, pp. 1187–1198.

[26] X. Yin, J. Han, and P. S. Yu, "Truth discovery with multiple conflicting information providers on the Web," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 6, pp. 796–808, Jun. 2008.

[27] Y. Li, H. Sun, and W. H. Wang, "Towards fair truth discovery from biased crowdsourced answers," in *Proc. 26th ACM SIGKDD Conf. Knowl. Discov. Data Min.*, 2020, pp. 599–607.

[28] C. Miao et al., "Cloud-enabled privacy-preserving truth discovery in crowd sensing systems," in *Proc. 13th ACM Conf. Embedded Netw. Sens. Syst.*, 2015, pp. 183–196.

[29] Y. Zheng, H. Duan, and C. Wang, "Learning the truth privately and confidently: Encrypted confidence-aware truth discovery in mobile crowdsensing," *IEEE Trans. Inf. Forensics Security*, vol. 13, pp. 2475–2489, 2018.

[30] C. Zhang, M. Zhao, L. Zhu, T. Wu, and X. Liu, "Enabling efficient and strong privacy-preserving truth discovery in mobile crowdsensing," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 3569–3581, 2022.

[31] J. Bai, J. Gui, T. Wang, H. Song, A. Liu, and N. N. Xiong, "ETBP-TD: An efficient and trusted bilateral privacy-preserving truth discovery scheme for mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 2203–2219, Mar. 2025.

[32] H. To, G. Ghinita, and C. Shahabi, "A framework for protecting worker location privacy in spatial crowdsourcing," *Proc. VLDB Endowment*, vol. 7, no. 10, pp. 919–930, Jun. 2014.

[33] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proc. IEEE Symp. Found. Comput. Sci.*, Oct. 2013, pp. 429–438.

[34] S. Xiong, A. D. Sarwate, and N. B. Mandayam, "Randomized requantization with local differential privacy," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 2189–2193.

[35] Y. Li et al., "Towards differentially private truth discovery for crowd sensing systems," in *Proc. 40th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Nov. 2020, pp. 1156–1166.

[36] P. Zhang, X. Cheng, S. Su, and B. Zhu, "PrivTDSI: A local differentially private approach for truth discovery via sampling and inference," *IEEE Trans. Big Data*, vol. 9, no. 2, pp. 471–484, Apr. 2023.

[37] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Proc. 31st Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 3574–3583.

[38] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *Proc. 26th USENIX Secur. Symp.*, Aug. 2017, pp. 729–745.

[39] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries," *Proc. Privacy Enhancing Technol.*, vol. 2016, no. 3, pp. 41–61, Jul. 2016.

[40] X. Ren et al., "LoPub: High-dimensional crowdsourced data publication with local differential privacy," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 9, pp. 2151–2166, Sep. 2018.

[41] Q. Ye, H. Hu, N. Li, X. Meng, H. Zheng, and H. Yan, "Beyond value perturbation: Local differential privacy in the temporal setting," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2021, pp. 1–10.

[42] Q. Ye, H. Hu, K. Huang, M. H. Au, and Q. Xue, "Stateful switch: Optimized time series release with local differential privacy," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, May 2023, pp. 1–10.

[43] Y. Mao, Q. Ye, H. Hu, Q. Wang, and K. Huang, "PrivShape: Extracting shapes in time series under user-level local differential privacy," in *Proc. IEEE 40th Int. Conf. Data Eng. (ICDE)*, May 2024, pp. 1739–1751.

Zhimao Gong received the B.S. degree in software engineering and the M.S. degree in computer science and technology from Hunan University, Changsha, China, in 2020 and 2023, respectively, where he is currently pursuing the Ph.D. degree in computer science and technology with the College of Computer Science and Electronic Engineering. His research interests include cloud computing security and blockchain.

Jiapeng Zhang received the B.E. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China, in 2017, and the Ph.D. degree in electronic engineering from Shanghai Jiao Tong University, China, in 2022. He is currently an Assistant Professor with the College of Computer Science and Electronic Engineering, Hunan University. His research interests include high performance parallel algorithms, resource scheduling design, and network performance analysis.

Haotian Wang received the B.S. degree from the School of Information Engineering, Nanchang University, China, in 2018, and the Ph.D. degree in computer science from Hunan University, China, in 2023. He is currently a Post-Doctoral Fellow at Hunan University. He previously completed a one-year joint Ph.D. Program at Nanyang Technological University. His research interests include parallel computing, tensor compilation, and artificial intelligence. He is an ACM Member.

Mingxing Duan (Member, IEEE) received the Ph.D. degree from the School of Computer Science, National University of Defense Technology, China. He is currently an Associate Professor at the School of Information Science and Engineering, Hunan University. He has been a Post-Doctoral Fellow with the Department of Computer Engineering, College of Computer Science and Electronic Engineering, Hunan University, and a Post-Doctoral Fellow with the Department of Computing, Hong Kong Polytechnic University. He has been supported by Chinese Hong Kong Scholars Program. His research interests include big data and machine learning. He has served on the Editorial Board for *American Journal of Neural Networks and Applications*.

Keqin Li (Fellow, IEEE) received the B.S. degree in computer science from Tsinghua University in 1985 and the Ph.D. degree in computer science from the University of Houston in 1990. He is a SUNY Distinguished Professor with the State University of New York and a National Distinguished Professor with Hunan University, China. He has authored or co-authored more than 1000 journal articles, book chapters, and refereed conference papers. He is among the world's top five most influential scientists in parallel and distributed computing in terms of single-year and career-long impacts based on a composite indicator of the Scopus citation database. He is a member of the SUNY Distinguished Academy. He is an AAAS Fellow, an AAIA Fellow, and an ACIS Founding Fellow. He is an Academician Member and a fellow of the International Artificial Intelligence Industry Alliance. He is a member of Academia Europaea (Academician of the Academy of Europe). He was a 2017 recipient of the Albert Nelson Marquis Lifetime Achievement Award for being listed in Marquis *Who's Who in Science and Engineering*, *Who's Who in America*, *Who's Who in the World*, and *Who's Who in American Education* for more than 20 consecutive years. He received the Distinguished Alumnus Award from the Computer Science Department, University of Houston, in 2018. He received the IEEE TCCLD Research Impact Award from the IEEE CS Technical Committee on Cloud Computing in 2022 and the IEEE TCSVC Research Innovation Award from the IEEE CS Technical Community on Services Computing in 2023. He received the IEEE Region 1 Technological Innovation Award (Academic) in 2023.

Kenli Li (Senior Member, IEEE) received the Ph.D. degree in computer science from the Huazhong University of Science and Technology, China, in 2003. He is currently a Full Professor of computer science and technology with Hunan University and the Deputy Director of the National Supercomputing Center, Changsha. He has published more than 300 papers in international conferences and journals. His research interests include parallel computing, cloud computing, and big data computing. He is a fellow of CCF. He serves on the editorial boards for IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON SERVICES COMPUTING, and IJPRAI.