Transparent Decisions: Selective Information Disclosure To Generate Synthetic Data

Carlos Gavidia-Calderon[1], Steve Harris[2,3], Markus Hauru[1], Florimond Houssiau[1], Carsten Maple[1,4], Iain Stenson[1] and May Yong[1]

[1]The Alan Turing Institute
[2]University College London
[3]NIHR University College London Hospitals BRC
[4]University of Warwick

Abstract

The UK government and the public wish to see the National Health Service (NHS) use data and Artificial Intelligence for public good [13][16]. However, there is a major challenge in making health data available for research whilst respecting patient privacy. Synthetic data generation is an emerging technique that enables access to data that, in some way, shares the characteristics of the original data. In this paper we introduce SqlSynthGen (SSG), a method for generating synthetic relational datasets. SSG offers a human-readable, risk-guided approach to refining data fidelity while managing disclosure risk. This paper presents SSG, specifically focusing on its application for generating synthetic data from NHS hospitals.

1 Introduction

Hospitals electronic health record systems are typically built using relational databases containing millions of records. While hospital staff access this data for their clinical duties, other professional communities— scientists, software engineers and educators — rightly must follow lengthy processes to be granted access. Controls are in place to ensure patient data —which is both sensitive and valuable [28]— is accessed for only legitimate reasons. Current practices involve preparing employee contracts, implementing de-identification or anonymisation mechanisms to remove personal information, and accessing data only via Trusted Research Environments [14].

While protecting patient privacy is of utmost importance, these processes impede collaboration and engagement, and introduce delays to researchers already working to arduous grant deadlines. For instance, researchers can use data to improve diagnostic accuracy, refine our understanding of diseases, or develop personalised treatments [30]. Patient data can be used to train the next generation of healthcare practitioners and researchers. Synthetic data is an accelerator: it can provide a simulcrum with the characteristics of patient data that can be shared onwardly. This can be used to support education and training, to quality control applications and code, and to test reproducible analytical pipelines in the open. This will accelerate academic progress for patient benefit.

In order to both protect user privacy and control access, current techniques employ mechanisms including data agreements, de-identification or anonymisation, aggregation over the original data, and provision of trusted

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

research environments (TRE) for access by third parties. While these techniques provide an extra layer of protection, they are not immune to vulnerabilities [21]. For example, de-identified data releases are still susceptible to linkage attacks. Aggregation requires releasing only aggregate population metrics, such as counts or averages, but outliers remain vulnerable to identification [30]. Instead of releasing real patient data —either partial or aggregate— an option is to release synthetic patient data.

Synthetic data is data that is manufactured, as opposed to real data that is collected from real-life events and people. Synthetic data generators (SDG) use algorithms to produce synthetic data entries while preserving statistical properties of the real dataset. There are multiple SDG approaches in the literature, each one targeting a specific data type, such as tabular data or time-series data [17]. SDGs can, when appropriately constructed, offer mathematical guarantees of the preservation of user privacy [19, 8] by incorporating differential privacy.

In this paper, we describe our work on developing a new SDG approach at the University College London Hospitals (UCLH) NHS Foundation Trust. Each year, UCLH admits 100,000 patients and stores their data in a relational database. Broadly, we discover that these are their requirements regarding their utilisation:

- **REQ-1:** The synthetic datasets should be in the form of relational datasets for any given relational schema
- **REQ-2:** The generator can manufacture synthetic data by utilising aggregates and statistical properties extracted from real patients
- **REQ-3:** Ensure that information disclosed about real patients are easily understandable by humans.

Listing 1: Requirements for Synthetic Data Generation at UCLH Trust

We developed SQLSYNTHGEN [12] to meet the requirements in Listing 1. SQLSYNTHGEN is an open-source Python package that can replicate the database schema of a relational database. Once the replica is in place, SQLSYNTHGEN can generate synthetic samples at different levels of fidelity: from low-fidelity random values compliant with the database schema, to high-fidelity samples from probability distributions learned from real data.

SQLSYNTHGEN uses a white-box approach where information extraction from real data are expressed as SQL queries in human-readable format, rather than black-box approaches, such as deep generative models with thousands of parameters [6]. For ensuring patient privacy, SQLSYNTHGEN supports differential privacy (DP)[10] to add quantifiable noise to the information extracted from the real data.

2 Sharing Patient Data

This section starts by enumerating motivations for sharing patient data. An understanding of motivations is important because these determine the requirements of appropriate data sharing mechanisms. The reasoning for sharing data dictates what minimum data needs to be shared, and this in turn defines the requirements to be met if the data is to be shared reasonably safely.

We then survey the current privacy preservation practices currently adopted by hospitals to enable collaborators controlled access to hospital data. We show that these are a) linked to inadequate privacy protection measures [21, 30], or b) a cause of unnecessary friction to analysis [23]. While synthetic data is considered a potential solution to overcome the above challenges, many patient datasets are organised as relational databases. Current synthetic data generators have limitations: a) they do not address the unique challenges of the relational structures [22][32]; b) they require users to specify dataset schemas [29]; or c) they can achieve differentially private, explainable, high-fidelity synthetic data for relational databases but currently face limitations in scalability. [8].

2.1 On the Benefits of Sharing Patient Data

Enhancing Research Quality and Innovation: Collaboration can lead to more comprehensive research studies, allowing healthcare practitioners and researchers to test hypotheses or observe trends across a broader dataset than is available internally. How well a dataset represents the true distribution matters more than simply dataset size[2]. In the medical domain, where lack of data is a common occurrence, the amalgamation of diverse datasets has a better chance of representing true underlying distributions.

Access to Specialised Expertise: External collaborators bring specialised knowledge and skills that complement the in-house capabilities of a hospital. For example, collaborations with methodology researchers can lead to state-of-the-art data analysis and interpretation, thereby improving both method development and treatment outcomes. Software engineers and machine learning operations engineers can build customised cyber-physical infrastructure to support analysis of patient data in real time[14].

Accelerating Medical Discoveries: By pooling resources and data between hospitals, research can proceed at a faster pace[2], potentially leading to quicker discoveries in disease mechanisms, treatment effectiveness, and development of new therapies or medical technologies. Sharing patient data can facilitate the recruitment of participants for clinical trials, ensuring a diverse and adequate sample size. This can be crucial in studying rare diseases or sub-types of common diseases, especially in hospitals that offer specialisations not commonly offered elsewhere in the world.

Expanding Research Funding Opportunities: Collaborative research often has better chances of securing funding[31]. Funding bodies frequently encourage or require collaboration across institutions as a criterion for grants, viewing it as a way to maximise the impact of their investment.

Bench-marking and Quality Improvement: Comparing data across institutions can help identify best practices and areas for improvement in patient care and management. This bench-marking is used to drive quality improvement initiatives within a hospital[33].

Education and Training: Collaborations provide educational opportunities to clinical research employees at hospitals, researchers and students at universities and research institutions, exposing them to different perspectives, methodologies, and cutting-edge research through joint ventures and knowledge exchanges.

Building Networks and Reputation: Collaborations can enhance a hospital's reputation in the medical and scientific community[31]. They extend the hospital's influence and recognition, which can attract top talent and more collaborations in the future.

2.2 Current Practices For Sharing Patient Data

De-identification and Anonymisation of Patient Data: De-identification is the process of obscuring or replacing personal identifiers to prevent the direct association of data with an individual. Common de-identification methods include explicit removal, masking or pseudonymisation of direct identifiers, and aggregating data to remove specificity eg. binning.

Anonymisation aims to ensure that data cannot be linked back to an individual by any means. Anonymisation strips datasets of all personal identifying information but it is not provable when this has been achieved. Conservative measures will strip a lot of information thereby heavily affecting the value of the dataset, and we still cannot be certain that there is not some way to de-anonymise.

For example, the removal of timestamps from a medical dataset as part of an de-identification or anonymisation process is performed because timestamps can be used to re-identify a patient by linking a patient's records over multiple de-identified datasets. The pattern of timestamps can disclose information about a patient's health, as well as their frequencies away from home.

However the stripping of timestamps from a medical dataset erases important information because medical information is highly time-contextual. Part of the richness of medical data is its time-series nature. Medical data that has been stripped of time stamps has reduced richness of data and is limited what can be learnt from it.

Effectiveness of both de-identification and anonymisation techniques is highly dependent on context, which includes the dimensionality, volume, and statistical properties of data. Other important aspects that need to be considered include which types of applications or analyses the data are to be used for, whether the data will be released publicly or with additional access control, and whether the data are tabular, relational, or have longitudinal or transactional characteristics.

Trusted Research Environments: Trusted Research Environments (TREs) are an important part of the data sharing mechanism ecosystem. TREs are the secure infrastructure and governance model that allows researchers to access and analyse data; they are often used in conjunction with other data-sharing mechanisms.

TREs play a major role in controlling data access levels. Initially, data access is controlled through secure authentication and authorisation mechanisms. This means that only approved researchers can access the data, and they can only access specific datasets approved for their role and research projects. Activities in TREs are closely monitored and logged.

In addition, TREs provide both physical and virtual security. Data in TREs are often stored in physically protected facilities. Virtual security measures such as firewalls, intrusion detection systems and regular penetration testing maximise protection against external threats. Finally, to ensure no privacy leakage, data egress from TREs is restricted. Researchers can analyse data within TREs but cannot take it out.

This means that working with data within TREs is far from a comfortable experience [23]. In order to provide security measures, computational resources can be limited and the list of approved software packages for analysis is restricted and not easily updated. There is significant process overhead generated by the need for detailed authentication into remote machines, activity logging, monitoring and compliance checks. There is a steep learning curve in working within a TRE, and new users are heavily dependent on support staff for technical assistance. Finally, the inability to egress data limits the sharing of interim findings and prevents close collaboration on ongoing data analysis.

Honorary contracts and data agreements: In order for non-hospital/clinical staff to work with medical data, they typically either need to become honorary employees of a trust or their current institution need to enter into a data sharing agreement with the trust. Both are lengthy and restrictive.

The process of obtaining an honorary contract typically begins with an initial inquiry and application to the relevant department or clinical group at the hospital. This is followed by credential verification and background checks, including border security investigations. Once these checks are satisfactorily completed, the relevant departments can grant approval.

To get a data agreement signed between two institutions, the first step is to identify the need for data sharing, specifying what data will be shared and how it will be used. Next, security requirements for storing, protecting, and accessing the data must be agreed upon by both parties. All these elements need to comply with relevant regulations. Finally, the agreement must be reviewed by the legal and compliance teams of both institutions to ensure all requirements are met and all parties are protected.

3 From Sharing Real Data to Sharing Synthetic Data

Real data is recorded from real life. Synthetic data is manufactured data, and can be created such that data elements are random, structurally or type accurate, or have distributions that mirror statistical properties of another dataset. In the last case, statistical properties can be directly or indirectly observed, to inform the data manufacturing process. When any properties of one dataset is used to guide the manufacturing process of another dataset, the first dataset is referred to as the 'real' or 'original' data. In the use case presented in this paper, 'real' data is hospital patient data. Our manufactured data is commonly referred to as 'synthetic' data.

While manufactured patient data is not about real individuals, it is a fallacy to imagine that adoption of synthetic data in data sharing practices prevents disclosure of sensitive information. This section shows how synthetic data generators can manufacture outputs which disclose more, or less sensitive information, and how this affects the ways in which outputs can be used.

3.1 Synthetic Data Generators

Synthetic data generators (SDG) manufacture data. There is a tension observed in the process of manufacturing synthetic data which involves three factors: fidelity, utility and privacy. Fidelity measures the extent to which synthetic data resembles the real dataset. Utility is the measure of the usefulness of synthetic data to a given task. Privacy is a measure of the information disclosed about the real dataset during generation of the synthetic dataset. These three factors inform the manufacturing process and limit the ways its outputs can be used. Synthetic data which is very similar to the real dataset (high fidelity) risk leaking information about real patients (low privacy). Conversely, low fidelity datasets typically contain little information relating to the real data, so individuals are unlikely to be identified. However, this low fidelity also limits the dataset's utility. For instance, medical data stripped of personal identifiers such as timestamps loses its richness and reduces the scope of insights that can be derived from it.

However, low-fidelity or coarse-grained datasets can still be useful, as utility is dependent on the context or task. In some cases, low-fidelity datasets are valuable if they provide sufficient information for engineering applications e.g. software testing. When paired with real data, multi-fidelity datasets can reduce computational costs and prevent over-fitting in machine learning tasks [26][27][5]. Low fidelity datasets can remove blockers at the beginning of research for initial exploration, building pipelines, and testing models. These tasks can be conducted in a secure environment restricted to students and researchers, with scripts later ported to the hospital for training on real data if the initial analysis proves promising.

This means that there is a class of low-fidelity datasets that is useful in common research and engineering tasks. The benefits of using these datasets can be realised with little cost to patient privacy.

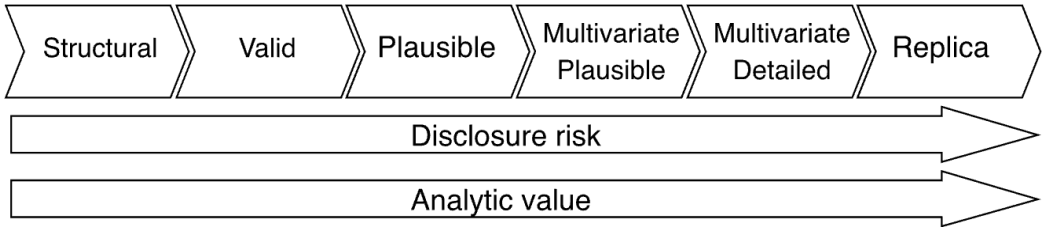

Figure 1: Shows the range of fidelity for synthetic data. High fidelity data can result in higher utility, but also increased risk of identification. Sourced from UK Office of National Statistics[24].

The UK Office of National Statistics [24] have defined a spectrum of fidelity for synthetic data, shown in Figure 1. In the context of healthcare relational datasets:

- **Structurally correct datasets** have the same column names, tables and relationships as real data.

- **Valid datasets** imply that the values in the synthetic dataset are correct and valid, e.g. date of births are valid dates.
- **Plausible datasets** imply that the relationship between values are realistic, e.g. a patient's date of death is not before their date of birth.
- **Multivariate plausible datasets** implies that the values are correlated across different variables, e.g. a male patient is likely to be both heavier and taller than a female patient.
- **Multivariate detailed datasets** are more realistic than a multivariate plausible data set, but less than a replica of the real data. An example are rows of data showing that a patient with a diabetes diagnosis has more records of blood sugar readings than a patient with a broken bone.

3.2 Synthetic Data For UCLH NHS Trust

University College London Hospitals National Health Services Foundation (UCLH NHS) Trust is a pioneering institution within the UK, renowned for its treatment care and specialist services not widely available in other NHS Trusts. It is closely affiliated with University College London; this is a partnership that emphasises research and education, integrating medical research and teaching at the undergraduate and postgraduate levels directly into the clinical environment. As an institute that emphasises medical care, research and education, and as custodians of highly sensitive medical data, UCLH NHS Trust are in a position to leverage research capabilities to supercharge innovation if they can develop a process for thoughtful access to this data. However, consequences of unintentionally releasing identifiable information include loss of individuals' privacy, loss of institutional prestige, as well as substantial legal fines.

3.2.1 Problem Statement

Machine learning (ML) infrastructure are deployed in hospitals to enable AI in healthcare delivery and administration. ML infrastructure supports tasks such as structuring data from electronic health records into a format that can be used as inputs to AI algorithms, deploying image analysis and predictive analysis tools, and presenting the results to healthcare practitioners in a timely and useful format.

To achieve these tasks, engineers who build the infrastructure need to gain an understanding of the data structures and data flow within the hospital. Researchers need to evaluate if target datasets meet their purposes for hypothesis testing, and are adequate in terms of quality and quantity. It is onerous to issue contracts to entire teams of engineers, researchers and students, but there are no other ways to share data with external collaborators.

However, what engineers and researchers need when working on early stages of exploratory analysis to understand data in terms of content, structure and data flow is information about the data, rather than having access to individual rows of data itself. Here is an opportunity to frame the problem as: What information can be released about sensitive data, which is maximally beneficial to engineers and researchers, with minimal cost to patient privacy?

3.2.2 Requirements

Listing 1 enumerates the requirements of building synthetic data generators for UCLH Trust. This section expands on each requirement; the following section demonstrates how the design of SSG fulfils these requirements.

Produce relational datasets for any given schema: Many data holders, including hospitals, store patient electronic health records in relational databases. Data is often structured within complex schema that capture both single observations and time series data. These relational databases also include tables for vocabularies such as definitions of drugs, observations and diagnoses.

Under this requirement, a minimally useful synthetic dataset must at the very least a) be structurally correct. That is, it will contain the same tables, columns, and data types as the real data, and b) meet foreign key constraints. In order to increase analytical value as shown in Figure 1, the synthetic generator will need to generate values which are valid and plausible, e.g. valid gender values and a plausible distribution of height and weight. A multivariate plausible dataset will have values that correlate across multiple tables, e.g. the correlation between gender and height are represented across the 'Demographic' and 'Observation' tables.

An additional complexity here is in generating synthetic time series data, e.g. blood pressure values every ten minutes for a patient in intensive care unit. In order to be multivariate plausible, the data needs to contain the correct frequencies for data collection as well as plausible values that depend on a patient's physiology. This is generated across multiple tables as well.

Generate synthetic data using statistical properties computed from real patients Hospitals are mandated or encouraged by various information acts to release hospital information to the public. The main reasons for this are a) allowing insights into quality of care provided by public or insurance funds and b) to enable patients to make informed decisions regarding where to seek care based on hospital performance and specialisations[33].

The type of information that is released in the public domain includes quality of care indicators, patient safety data, readmission rates and service availability. This includes aggregate data about patient outcomes, infection rates, details on specialised services, bed occupancy, Accidents and Emergency (A&E) wait times as well as statistical properties on patients returning for treatment within a period of discharge. This information is published regularly and does not compromise individual patient privacy.

Synthetic data generators can use aggregate data and statistical properties of real data to generate datasets which are measurably closer to real data. A synthetic dataset generated using public information is unlikely to reveal any additional patient information beyond what is already publicly available.

Ensure that information disclosed about real patients are easily understandable by humans. Aggregates and statistical properties are well-understood mathematical concepts. A comprehensive explanation of such information extracted from real patients datasets for the purpose of generating synthetic data should cover the following three points:

1. **Extracted Information**: Detail what specific information about patients has been extracted.
2. **Computation Process**: Explain how this information is computed.
3. **Usage for Synthetic Data**: Describe how this information is used to shape the synthetic data.

Providing this explanation in a single, human-readable source ensures consistency and prevents obsolescence across multiple data generation iterations. This offers a clear audit trail of the generation process and helps identify the disclosure risks of its outputs.

The concept of synthetic data is complex, people may not understand how data that does not represent real individuals still needs privacy considerations. It is furthermore difficult to understand how the application of differential privacy to aggregates and statistics can provide additional protection.

Differential privacy (DP) [10] is the gold standard that protects individuals within a dataset while still allowing for the useful analysis of the aggregate data. Its internal mechanics of noise addition for the purpose of privacy preservation can leave users without a clear understanding of its outputs and how to interpret them correctly[9].

The application of differential privacy to synthetic data compounds the explanations' complexities. There is a struggle to understand how DP offers probabilistic but not absolute guarantees. Explaining this to custodians of highly sensitive data is difficult because privacy is expected but not always technically feasible.

However, this is an important discussion, there is a necessary understanding to be achieved here because the interplay between privacy and utility governs the results of a differentially private synthetic data generator. The

only people who can take the responsibility for managing the balance between privacy and utility are the data custodians.

4 Generating Synthetic Data Using SQLSYNTHGEN

SQLSYNTHGEN (SSG) is a software package developed to meet the requirements outlined in Section 3.3. When connected to an existing relational database, SSG builds a new empty database with the same schema. It copies over the non-sensitive data, such as look-up tables, and generates structurally correct synthetic data with random values. Optionally, SSG can refine these synthetic values using aggregates and statistical properties. SSG can apply differential privacy to obfuscate the true values of these properties in a measurable way. The new database is then populated with these synthetic values.

4.1 Technical Overview

The default output dataset from SSG is structurally correct and has no disclosure risk. These are datasets that sit on the far left end of the spectrum in Figure 1. No information about the real dataset has been disclosed, beyond the structure in which they are stored. This can already be useful e.g. for building software testing modules and pipe-lining scripts, and can be safely released if vocabularies and schema can be shared. ***This meets REQ-1: Produce relational datasets for any given schema.***

SSG can be further configured to generate synthetic data that (in reference to Figure 1), can be as sophisticated as multivariate plausible data. This is achieved by allowing the user to define SQL statements that extract aggregate statistics and statistical properties from the real data. These extracted values are then used to shape the distributions and marginals of the synthetic data. ***This meets REQ-2: To generate synthetic data using statistical properties computed from real patients.***

As part of its process, SSG generates a human-readable audit trail that details the entire data generation process. This includes what information was extracted from real data, the methods used for extraction, the computed results, and how these values were injected into the synthetic data generation. The audit trail is a human readable file whose contents are incorporated directly into the SDG process. ***Ensure that information disclosed about real patients are easily understandable by humans.***

SSG pipeline design enables the selective production of synthetic datasets with varying levels of fidelity. Users control the shaping of synthetic data by specifying which information is extracted from real data, how it is computed, and how it is utilised. SSG's configuration supports agile development, allowing for incremental fidelity improvements as needed, while maintaining transparency, auditability, and control over privacy risks at every stage. Additionally, users have the option to apply differential privacy to protect the marginals extracted from the source data.

In order to support this design, SSG's process for generating synthetic relational datasets can be broken into three separate steps, as shown in Figure 2. They are as follows:

1. SSG **builds** a new database to store synthetic data. This new database will be populated by synthetic data generated in the next steps. Look-up tables which do not have any privacy concerns are copied over entirely, to maintain foreign key constraints.

2. By default, SSG **generates** random but structurally correct data.

3. As an option, SSG can **refine** random values for higher accuracy by using extracted statistics from real data, with or without DP. For example, mean of height by age and gender can be extracted from real patients and the correlation be used to generate higher fidelity data.

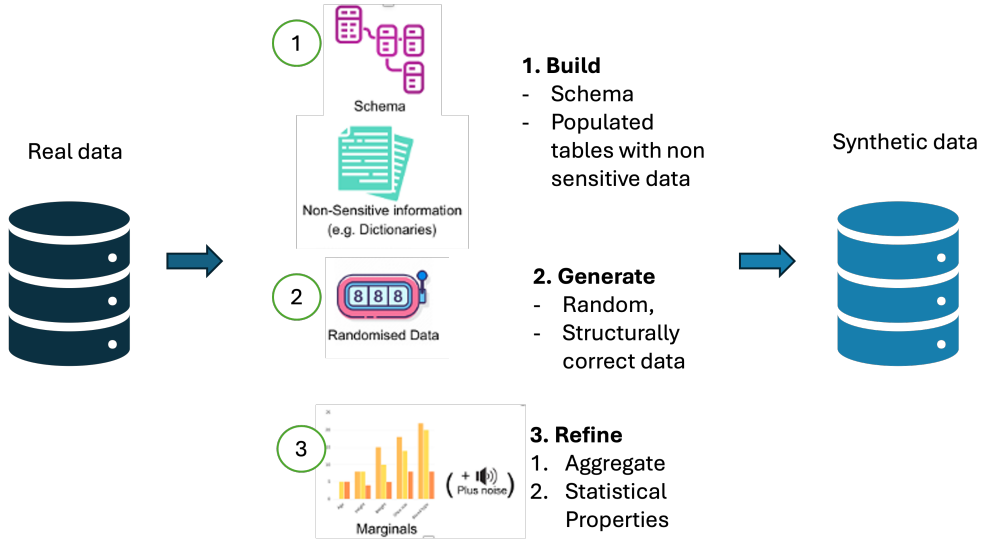

Figure 2: The processes of SQLSynthGen in order

For more information and tutorials about SQLSynthGen, please refer to our repository at `https://github.com/alan-turing-institute/sqlsynthgen`. Our repository [12] contains installation instructions, comprehensive documentation and trouble shooting guides to help get started with the software. The repository also contains a simple tutorial using a Kaggle dataset [7] as well as an advanced example based on the Observational Medical Outcomes Partnership (OMOP)[25], which provides a standardised data model for observational healthcare data.

In the following sections, we demonstrate the use of SSG in creating synthetic data based on a publicly available AirBnB Kaggle dataset [7].

4.2 Building a Replica of a Real Dataset

In this example, let us consider that our dataset is contained in a database called ‘airbnb‘ in a local PostgreSQL instance. We want to port the schema to a new ‘airbnb_synthetic‘ database, and populate the ‘airbnb_synthetic‘ database with synthetic rows that mirror some of the statistical properties of the ‘airbnb‘ dataset.

Build schema tables: We connect to the real dataset by setting connection credentials in environment variables. We run a series of commands `sqlsynthgen make-tables`, `sqlsynthgen create-tables` and `sqlsynthgen make-generators` to auto-generate two Python files.

The first file, ‘orm.py‘, outlines the structure of the PostgreSQL ‘airbnb‘ dataset by mapping each table in ’airbnb’ to a corresponding Python class. Each column in these tables is represented as a class field. This mapping is generated using SQLAlchemy[4], which is a SQL toolkit and Object-Relational Mapping (ORM) library for Python. By using SQLAlchemy in SSG for mapping, users do not need to perform any additional configuration to describe the schema of the real dataset. The ‘orm.py‘ file serves as a foundation for building a new ‘airbnb_synthetic‘ PostgreSQL database, complete with the necessary tables, columns and data types. Listing 2 shows a snippet from ‘orm.py‘ that demonstrates how the ‘users‘ table from the ‘airbnb‘ dataset is mapped as a Python class.

```python
class User(Base):
    __tablename__ = "users"

    id = Column(String, primary_key=True)
    date_account_created = Column(Date)
    ...
```

Listing 2: Section of PostgreSQL table ‘user‘ represented as a Python class

Copy over lookup tables: A lookup table, or a vocabulary, is a table used to store a predefined set of values that are referenced by other tables. They contain a finite and static set of values such as codes, names of categories or descriptions. Look-up tables are a good practice adopted help normalise databases by removing redundancy and enabling efficient data management. They work by using foreign key constraints to ensure values in related tables are consistent and valid. These foreign key constraints need to be satisfied when generating synthetic data in relational datasets. On their own, vocabularies provide only limited utility, since the more interesting aspects of the data are usually found in the non-vocabulary tables.

The fidelity of the synthetic dataset can be improved by ensuring the vocabulary tables have perfect fidelity from the beginning, since they do not raise privacy concerns (although some vocabularies are copyright-protected). In this section, we demonstrate how SSG addresses vocabulary tables by copying them in their entirety, thereby eliminating the need for synthesis.

First we specify vocabulary tables in a `config.yaml`; the listing 3 below denotes ‘countries‘ as a vocabulary table. All values in denoted vocabulary tables are copied to an auto-generated `.yaml` file. Listing 4 shows a snippet of data from the ‘countries‘ table which has been copied to a auto-generated `countries.yaml` file.

```yaml
tables:
    countries:
        vocabulary_table: true
```

Listing 3: A yaml section to demarcate table ’countries’ as a vocabulary table

```yaml
- country_destination: AU
  destination_km2: 7741220
  destination_language: eng
    :
- country_destination: CA
  destination_km2: 9984670
  destination_language: eng
  distance_km: 2828.1333
    :
```

Listing 4: Example of data rows copied from ‘countries‘ vocabulary table

The primary reason for copying vocabularies this way is to maximise transparency for auditing purposes. Data holders can audit each value extracted from the real dataset, before creating any synthetic data. Note that we

have to be careful in making sure that the tables marked as vocabulary tables truly do not hold privacy sensitive data, otherwise catastrophic privacy leaks are possible, where the original data is exposed raw and in full.

The downside of this approach is clear when scaling up to address vocabulary tables which are very large. Therefore our generator pipeline is modular to ensure that vocabularies need only be copied once when creating more rows to add into a synthetic dataset.

Generate Random Values that are Structurally Correct: The second auto-generated file, 'ssg.py', contains Python code that generates random values matching the data types defined by the Python classes. This human-readable Python code serves as part of the audit trail, demonstrating how values for populating each table column are generated. For complex schemas with multiple tables and columns, the generator code for each column is easily identifiable and can be customised independently of rest of the generator.

Listing 5 demonstrates the auto-generated Python code for generating 'id' and 'date_account_created' values for the 'User' table. 'id' is assigned generic, password-like values, and 'date_account_created' is assigned a random date value.

```
class usersGenerator:
    num_rows_per_pass = 1

    def __init__(self, src_db_conn, dst_db_conn):
        pass
        self.id = generic.person.password()
        self.date_account_created = generic.datetime.date()
        ...
```

Listing 5: A Python class for generating synthetic id and date_account_created values for Postgres table 'User'

Refine values using aggregate statistics: The default behaviour of SSG is to generate syntactically correct, random values. This section shows how we incorporate aggregate and statistical properties of real data in order to generate synthetic data that retain those properties.

We demonstrate an example to generate normally distributed synthetic values to populate a 'users.age' column, with reference to the mean and standard deviation values of the real data. The user begins by defining SQL statements in the 'age_stats' section of a 'config.yaml' file. This is demonstrated in listing 6. SSG uses the credentials provided to authenticate to the database and execute SQL statements to compute the required values. Computed values are recorded in an auto-generated `src-stats.yaml` file, demonstrated in listing 7. These can be can be referenced by the Python data generators. Listing 8 shows the Python provider function that generates a distribution of values to meet the statistical properties computed and recorded in 'config.yaml' and 'src-stats.yaml'.

```yaml
src-stats:
    - name: age_stats
    query: >
    SELECT AVG(age)::float AS mean, STDDEV(age)::float AS std_dev
    FROM users
    WHERE age <= 100
tables:
    users:
        row_generators:
          - name: airbnb_generators.user_age_provider
            kwargs:
              query_results: SRC_STATS["age_stats"]
            columns_assigned: age
```

Listing 6: A section of the config.yaml file that shows an SQL statement to compute mean and average of column ‘users.age‘. Results are stored as ‘age_stats‘.

```yaml
age_stats:
- mean: 36.54434029695572
  std_dev: 11.708339792587486
```

Listing 7: Example of mean and standard deviation values computed from ‘users.age‘ column

```python
  import random
  def user_age_provider(query_results):
      mean: float = query_results[0]["mean"]
      std_dev: float = query_results[0]["std_dev"]
      return random.gauss(mean, std_dev)
```

Listing 8: A provider function

The primary reason for extracting information using SQL statements and documenting it in ‘config.yaml‘ is to maximise transparency for auditing purposes. Similar to vocabularies, users can audit information that is disclosed about real data by reviewing the human-readable ‘config.yaml‘ and ‘src-stats.yaml‘ files. Multiple properties, such as marginals, percentiles, and skewness, can be used simultaneously to enhance the fidelity of synthetic data. These computations can be resource-intensive with large datasets. To address this, the SSG generator process is modularised: properties are computed and stored once, allowing subsequent generators to reference these values, which will be reliable provided the real dataset has not changed significantly.

Introduce differential privacy into aggregate statistics: Differential privacy is arguably the most popular technique for providing privacy guarantees on SDGs. Let us imagine two datasets:

- A synthetic dataset B generated with information of person X.

- A synthetic dataset A generated without information of person X.

If both datasets were generated using a differentially-private mechanism, performing a query on dataset A should provide the same, or almost the same, result as performing the same query on dataset B [19]. Differentially private mechanisms hide the presence or absence of person X —or one any individual— in the dataset, which implies strong protection of their privacy [21]. To accomplish this, these mechanisms inject random noise to the synthetic data. The amount of noise is a function of the privacy parameter epsilon ε that measures how similar the datasets A and B are required to be. ε needs to be chosen carefully to provide the required privacy guarantee.

One of the most common fundamental techniques for generating synthetic data in a differentially private involved 3 steps: 1) select, or choose, some queries over the original data, 2) measure, or execute, those queries using a differentially private mechanism, and 3) generate synthetic data using these measurements [20].

SQLSYNTHGEN enables the select and measure steps by supporting differentially private SQL queries in 'src-stats.yaml' (Listing 9).

```yaml
src-stats:
  - name: age_stats
    dp-query: >
      SELECT AVG(age) AS mean, STDDEV(age) AS std_dev
      FROM query_result
    epsilon: 0.5
    delta: 0.000001
    snsql-metadata:
      max_ids: 1
      id:
        type: string
        private_id: true
      age:
        type: float
        lower: 0
        upper: 100
```

Listing 9: A differentially-private SQL query.

Internally, SQLSYNTHGEN uses SMARTNOISE SQL [1] to execute differentially private queries. As seen in Listing 9, SMARTNOISE SQL needs additional information besides the SQL query for applying a differentially private mechanism, including the privacy parameter epsilon ε. Regarding the final generate step, the query results are made available to provider functions —demonstrated in Listing 8— so SQLSYNTHGEN users can use these measures for data generation.

5 Discussion

The proliferation of research on synthetic data over the past five years underscores its significance in addressing data scarcity and sensitivity issues in machine learning. With 25,600 papers published from 2023 to mid-2024 alone, these studies span diverse domains, including computer vision, natural language processing, and healthcare [11], primarily focusing on the generation, evaluation, and application of synthetic data, particularly using GANs [3]. Originally research-driven, these methods are now being translated into practical applications, revealing new challenges and considerations [18].

Our development of a Synthetic Data Generator (SDG) for sharing sensitive hospital information has highlighted these key challenges:

There is a lack of generators developed for relational data: The development of synthetic generators commonly explore image, text data, or tabular data. Our experience is that synthetic data generators overlook the relational data format, possibly because of the foreign key constraints satisfaction criteria. This is a problem because hospital datasets are often stored in relational formats.

There is a lack of explainability in privacy preserving mechanisms: Explainability in synthetic data generators is a crucial issue for custodians of sensitive data, especially in hospitals. The lack of explainability undermines discussions between hospital data stakeholders, including both staff and patients. One discussion impacted by the lack of explainability is that of maintaining a balance between privacy guarantees and the utility of the synthetic data. While ensuring that synthetic data generators do not leak sensitive information is essential, explaining the privacy preservation mechanisms involved can be complex. Furthermore, the processes used by generators based on GANs and deep neural networks are opaque, making it difficult to assure stakeholders of the synthetic data's reliability and safety. Finally, both generators and metrics (e.g., fidelity, diversity) used to evaluate the quality of synthetic data are not easily interpretable.

We specifically addressed this explainability challenge in a series of workshops with patient and public involvement, and using SSG as an exemplar. There were two key messages from our stakeholders. Firstly, they were reassured to understand the distinction in the source of the data. Anonymised data is processed from the original data whereas synthetic data is generated *de novo*. Secondly, they valued using a language that talked about sharing information (with synthetic data) in contrast to sharing data (with anonymisation). There was recognition that information is *already* shared and tools like SSG are trustworthy because they are transparent about what information is used to generate the synthetic data.

Despite its design to address these challenges, our SQLSYNTHGEN tool has several limitations:

Lack of Autonomous Model Discovery: Unlike GANs-based [3] or Bayesian-based [8] generators, SSG cannot autonomously discover underlying models or relationships. Users must predetermine the models, limiting the tool's adaptability and the transferability of algorithms trained on its outputs to real-world data.

Need to Ensure Security: The design of SSG includes copying vocabulary tables in their entirety and executing SQL statements on real data based on user configurations, makes it a powerful tool. However, these features introduce risks of user errors. Accidental copying tables with sensitive data could lead to severe data breeches. Executing SQL statements without proper access controls could damage real patient information.

Lack of Evaluation: SSG allows users to selectively disclose information used to shape synthetic data outputs but it lacks an integrated evaluation mechanism. Since each piece of information is independently disclosed, there is an opportunity here to iteratively fine-tune the balance between fidelity and privacy by combining SSG with an evaluation tool such as TAPAS [15].

6 Conclusion

The number of research papers on synthetic data has surged significantly, indicating its growing importance in addressing data scarcity and sensitivity issues in machine learning. There is a notable gap in the development of synthetic data generators specifically for relational data structures. Most exciting developments on generators

focus on time-series, graph, audio, imaging or tabular data structures, often neglecting the complexities associated with relational databases, such as foreign key constraints. This limitation is significant because many practical applications, particularly in healthcare, rely heavily on relational data formats.

Aside from the oversight in provision for relational data, the lack of explainability in privacy-preserving mechanisms is a critical challenge. For synthetic data to be trusted and widely adopted, especially in sensitive domains like healthcare, stakeholders need to understand how privacy is preserved. The opacity of deep learning models and GANs currently used in generating synthetic data makes it difficult to provide this assurance, which can hinder stakeholder discussions and acceptance.

The direction for future work on the application of synthetic data generation in sensitive data context is clear:

1. **Development of Relational Data Generators:** There is a clear need for synthetic data generators that can handle relational data formats effectively, addressing issues like foreign key constraints.
2. **Improving Explainability:** Enhancing the explainability of synthetic data generation processes will be crucial for gaining stakeholder trust and facilitating broader adoption by custodians of sensitive data.
3. **Integrated Evaluation Frameworks:** Combining synthetic data generators with comprehensive evaluation or attack frameworks can help explainability as well as ensuring an optimal balance between fidelity and privacy.

By addressing these challenges and focusing on these future directions, the practical application of synthetic data can be significantly enhanced, making it a more viable solution for real-world problems, particularly in sensitive domains such as healthcare.

7 Acknowledgements

1. This project was funded by Ecosystem Leadership Award under the EPSRC OobfJ22\100020.
2. This project was supported by the National Institute for Health and Care Research (NIHR)University College London Hospitals (UCLH) Biomedical Research Centre (BRC).
3. The authors thank Olajumoke Olatunji for her help in improving the documentation of SSG and helping us present it.

References

[1] Joshua Allen, Sarah Bird, and Kathleen Walker. Opendp platform for differential privacy, May 2020.

[2] Alhanoof Althnian, Duaa AlSaeed, Heyam Al-Baity, Amani Samha, Alanoud Bin Dris, Najla Alzakari, Afnan Abou Elwafa, and Heba Kurdi. Impact of dataset size on classification performance: An empirical evaluation in the medical domain. Applied Sciences, 11(2), 2021.

[3] Márcio Antunes and Ernesto Oliveira. Survey on synthetic data generation, evaluation methods and gans. Mathematics, 10(15):2733, 2022.

[4] M. Bayer. Sqlalchemy. `https://www.sqlalchemy.org/`.

[5] Emily E. Berkson, Jared D. VanCor, Steven Esposito, Gary Chern, and Mark D. Pritt. Synthetic data generation to mitigate the low/no-shot problem in machine learning. In 2019 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), pages 1–8. IEEE, 2019.

[6] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. IEEE Trans. Pattern Anal. Mach. Intell., 44(11):7327–7347, 2022.

[7] Airbnb New User Bookings. Airbnb new user bookings. Kaggle, https://www.kaggle.com/competitions/airbnb-recruiting-new-user-bookings, 2015. [Accessed: November, 25, 2015].

[8] Kuntai Cai, Xiaokui Xiao, and Graham Cormode. Privlava: Synthesizing relational data with foreign keys under differential privacy. Proceedings of the ACM on Management of Data, 1:1–25, 06 2023.

[9] FK Dankar and K El Emam. Practicing differential privacy in health care: A review. Transactions on Data Privacy, 6(1):35–67, 2013.

[10] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci., 9(3-4):211–407, 2014.

[11] Joao Fonseca and Fernando Bação. Tabular and latent space synthetic data generation: a literature review. Journal of Big Data, 10, 07 2023.

[12] C. Gavidia-Calderon, M. Hauru, I. Stenson, and M. Yong. sqlsynthgen. https://github.com/alan-turing-institute/sqlsynthgen, 2024.

[13] Matt Hancock. Data saves lives: reshaping health and social care with data, 2022. Accessed: 2024-06-04.

[14] S. Harris, T. Bonnici, T. Keen, W. Lilaonitkul, M. J. White, and N. Swanepoel. Clinical deployment environments: Five pillars of translational machine learning for health. Frontiers in Digital Health, 4:939292, Aug 2022.

[15] Florimond Houssiau, James Jordon, Samuel N. Cohen, Owen Daniel, Andrew Elliott, James Geddes, Callum Mole, Camila Rangel-Smith, and Lukasz Szpruch. Tapas: a toolbox for adversarial privacy auditing of synthetic data, 2022.

[16] LA Jones, JR Nelder, JM Fryer, PH Alsop, MR Geary, M Prince, and RN Cardinal. Public opinion on sharing data from health services for clinical and research purposes without explicit consent: an anonymous online survey in the uk. BMJ Open, 12(4):e057579, Apr 2022.

[17] James Jordon, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N. Cohen, and Adrian Weller. Synthetic data - what, why and how? CoRR, abs/2205.03257, 2022.

[18] James Jordon, Lukasz Szpruch, Francois Houssiau, and Matteo Bottarelli. Synthetic data - what, why and how?, 2022.

[19] A Kopp. Microsoft smartnoise: Differential privacy machine learning case studies. Technical report, Microsoft, 2021. Accessed: 2024-06-08.

[20] Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. AIM: an adaptive and iterative mechanism for differentially private synthetic data. Proc. VLDB Endow., 15(11):2599–2612, 2022.

[21] J. P. Near and C. Abuah. Programming Differential Privacy, volume 1. 2021.

[22] Beata Nowok, Gillian M. Raab, and Chris Dibben. synthpop: Bespoke creation of synthetic data in r. Journal of Statistical Software, 74(11):1–26, 2016.

[23] Cian O'Donovan, Sonya Coleman, Dermot Kerr, Christian Cole, Simon Li, David Sarmiento Perez, and Hari Sood. Trusted research environment users. November 2023.

[24] Office for National Statistics. Ons methodology working paper series number 16: Synthetic data pilot. Technical report, Office for National Statistics, 2021.

[25] Observational Medical Outcomes Partnership (OMOP). Omop common data model. https://www.ohdsi.org/data-standardization/the-common-data-model/, 2024.

[26] A. Patra, R. Batra, A. Chandrasekaran, and C. Kim. A multi-fidelity information-fusion approach to machine learn and predict polymer bandgap. Computational Materials Science, 172:109280, 2020.

[27] C. Santoni, D. Zhang, Z. Zhang, and D. Samaras. Toward ultra-efficient high fidelity predictions of wind turbine wakes: Augmenting the accuracy of engineering models via les-trained machine learning. arXiv preprint arXiv:2404.07938, 2024.

[28] G. Schomerus et al. The stigma of alcohol-related liver disease and its impact on healthcare. Journal of Hepatology, 77(2):516–524, Aug 2022.

[29] Synth Team. Synth: The open source declarative data generator, 2021.

[30] A. Tucker, Z. Wang, Y. Rotalinti, and P. Myles. Generating high-fidelity synthetic patient data for assessing machine learning healthcare software. npj Digital Medicine, 3, 2020.

[31] West JD Vasan K. The hidden influence of communities in collaborative funding of clinical science. R Soc Open Sci., 8(8), 2021.

[32] Jason Walonoski, Mark Kramer, James Nichols, Marc Galdzicki, Amelia Quina, Christopher Moesel, Darrell Hall, Tim Duffield, Matthew Gratch, Pascal Coorevits, David Sundwall, Emily Grant, Colin Jones, and Liza Tong. Synthea: Synthetic patient population simulator, 2020. Accessed: 2024-06-08.

[33] Rachel M. Werner and David A. Asch. The unintended consequences of publicly reporting quality information. Journal of the American Medical Association, 293(10):1239–1244, 2005.

Differential Privacy for Time Series: A Survey

Yulian Mao[1,2], Qingqing Ye[2 *], Qi Wang[1,3], Haibo Hu[2]
[1]Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China
[2]Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University, Hong Kong SAR, China
[3]Shenzhen Key Laboratory of Safety and Security for Next Generation of Industrial Internet, Shenzhen, China
yulian.mao@connect.polyu.hk, qqing.ye@polyu.edu.hk, wangqi@sustech.edu.cn, haibo.hu@polyu.edu.hk

Abstract

Time series are extensively used in finance, healthcare, IoT, and smart cities. However, in many applications, time series often contain personal information, so releasing them publicly can pose privacy risks. Differential privacy has recently emerged as the state-of-the-art approach for safeguarding data privacy. Unfortunately, adapting differential privacy to time series presents unique challenges compared to other data types due to their large volume, temporal correlations, and dynamic nature. To address users' demands for time series analysis while simultaneously protecting privacy, a significant body of research works have been proposed. The aim of this survey is to summarize these works and provide a holistic view of the DP mechanisms under differential privacy. Furthermore, we will discuss the challenges associated with time series release, especially in one of its most prevalent applications — location based services. Finally, we will explore open challenges and shed light on directions for future research.

1 Introduction

Time series are being generated on a large scale across a wide range of application domains, such as IoT, finance, healthcare monitoring, operational event logs, and smart home sensors. For example, smart home devices such as thermostats and humidity sensors generate continuous time series on environmental conditions, tracking temperature fluctuations and moisture levels to optimize home climate control based on user activities. Additionally, trajectories represent a unique type of time series that contain both spatial and temporal information, such as GPS data tracking the movement of vehicles or individuals. To facilitate the analysis of time series and support various downstream tasks, numerous methods have been proposed, ranging from traditional statistical techniques, such as ARIMA [1] and exponential smoothing [2], to advanced machine learning models, such as long short-term memory (LSTM) networks [3]. However, a key issue in these time series applications is privacy. Since many data sources such as smart home sensors and location trajectories contain individuals' private information, the direct release or analysis of such time series can lead to significant privacy violations. Consequently, developing privacy-preserving mechanisms for time series analysis is essential.

Differential privacy (DP) [4] is a paradigm of privacy-preserving mechanisms that provides a theoretical privacy guarantee and has been further extended to the local setting to accommodate more general scenarios [5, 6].

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

* Dr. Qingqing Ye is the corresponding author.

Numerous DP mechanisms have been developed to support various queries [7, 8, 9], including estimating statistics such as frequency [7, 10] and mean [11], ensuring that the privacy of individuals in the dataset is protected even when aggregate information is released. Recently, research has shifted towards more complex applications, such as graph data mining [12] and machine learning problems [13]. Techniques such as differentially private stochastic gradient descent (DP-SGD) [14, 15] have been developed to train machine learning models with differential privacy guarantees, enabling the use of private datasets for tasks like classification and prediction without compromising individuals' privacy. It is worth noting that differential privacy is also being explored in the context of time series [16, 17, 19]. This involves developing new mechanisms capable of handling the features of time series, ensuring that privacy is maintained.

Nevertheless, time series data present more challenges compared to other types of data due to their large volume, temporal correlation, and dynamic nature. The large volume, in particular, poses significant issues for privacy models, as protecting every element of a time series would degrade utility. To address this challenge, three privacy levels have been proposed [19]: event-level privacy, which protects a single element in the time series; w-event level privacy, which provides a privacy guarantee for w consecutive elements; and user-level privacy, which protects all elements associated with an individual. To enhance utility, sampling and filtering-based mechanisms [18], as well as privacy budget allocation strategies [19], have been suggested. Additionally, the correlations between elements can lead to privacy breaches [20], necessitating countermeasures to confine these correlations [21, 22]. Given numerous research efforts in this field, a taxonomy is necessary to summarize the existing works and identify areas for future research.

However, to the best of our knowledge, there is no up-to-date and comprehensive survey specifically for time series under differential privacy. Dwork and Roth [23] coauthored a comprehensive survey on differential privacy, which seems outdated now in terms of state-of-the-art techniques. More recently, there are two surveys from Zhao et al. [24, 25] focusing on the concepts and applications of differential privacy, but they do not extensively cover time series. Miranda-Pascual et al. [26] conducted a survey on trajectory data publication, which mainly talks about downstream tasks with little emphasis on privacy preservation. The most relevant survey on time series under differential privacy is by Katsomallos et al. [27]. However, since it was published in 2019, the paper does not reflect current technical trends, such as LDP, which now accounts for a crucial portion of privacy-preserving time series research.

In this survey, we aim to provide a comprehensive review of research works on time series under differential privacy. The main contents and paper organization are summarized as follows.

- **Section 2: Fundamental concepts of time series and differential privacy.** We first introduce the basic concepts of time series and differential privacy, including the definitions of differential privacy and the composition theorems. Additionally, we elucidate the three privacy levels specifically defined within the context of time series.

- **Section 3: Count queries and corresponding advanced queries.** We begin with an introduction to count queries, and then present two core techniques for their realization: the binary tree-based mechanism [17] and the matrix mechanism [28]. Following this, we discuss advanced queries, such as frequency and histogram estimations, which are based on count queries. Finally, we explore downstream applications derived from count queries.

- **Section 4: Sum/mean queries and downstream applications.** We list sum and mean queries together due to their inherent correlation. Following the introduction of sum and mean queries, we present the developed mechanisms for these queries. Subsequently, we review the literature on downstream applications.

- **Section 5: Time series release.** We classify the literature into two categories: methods based on value perturbation and methods based on synthesis. For value perturbation-based methods, we first review privacy budget allocation strategies and then present the optimization strategies to improve utility. We

then introduce the synthesis-based methods, including those based on statistics and generative models. Additionally, we discuss the privacy models of the time series mechanisms and review a line of work that perturbs the temporal order rather than the values to accommodate value-critical scenarios.

- **Section 6: Location based services and trajectory release.** Given that location based services are common applications under differential privacy, we dedicate a section to discussing the relevant literature. To improve the utility, geo-indistinguishability [29] was proposed to constrain the perturbation domain. Moreover, due to the apparent temporal correlation, the relationships between locations need to be considered. Finally, we present mechanisms designed for trajectory release based on perturbation and synthesis.
- **Section 7: Open challenges.** We present a few future research directions for DP-based time series in terms of privacy model, potential correlation-based attacks, complex data type, and learning based problems.

2 Preliminaries

In this section, we first introduce the basic concepts of differential privacy and differential privacy for time series. As for the latter, we mainly focus on different privacy levels of DP mechanisms used to analyze time series.

2.1 Differential Privacy

Differential privacy is a rigorous and practical formalization that provides a quantitative measure of privacy leakage for an individual when participating in a database [4]. In the nearly 20 years since its inception, differential privacy has become widely adopted as a privacy-preserving framework. Many companies, such as Microsoft [30], Google [31], and Apple [32], utilize differential privacy to collect users' data while providing privacy guarantees. Additionally, the US Census Bureau adopted differential privacy for the 2020 decennial census [33].

Based on utilization scenarios, differential privacy can be broadly categorized into centralized differential privacy (CDP) and local differential privacy (LDP) [6]. Centralized differential privacy requires a trusted third party to act as the data curator, collecting data from users and releasing the processed results to the public. The trusted third party is assumed to safeguard private information and not disclose it. However, in many situations, such a trusted third party may not exist. Consequently, local differential privacy has been proposed, allowing data to be sanitized locally before being uploaded. These two different scenarios are depicted in Fig 1, and their formal definitions are provided below.

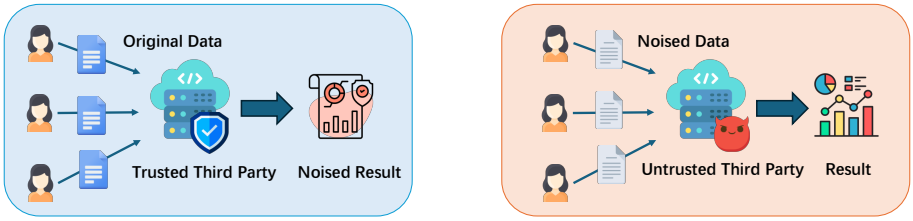

Figure 1: An illustration for centralized differential privacy and local differential privacy. In the context of centralized differential privacy, a trusted third party collects original data from users and adds noise to the processed result. In contrast, under local differential privacy, users add noise to their data locally before uploading the noised data to the untrusted third party.

2.1.1 Centralized Differential Privacy (CDP)

Before delving into the formal definition of centralized differential privacy, it is important to first elucidate some underlying concepts. We start with the definition of neighboring datasets. [Neighboring Datasets [23, 24, 34]] Two datasets D and D' are *neighboring* if they only differ by only one record. In *Unbounded CDP*, D can be obtained from D' by adding or removing one record, whereas in *Bounded DP*, D can be obtained from D' by replacing one record.

[(ϵ, δ)-Centralized Differential Privacy ((ϵ, δ)-CDP) [23, 24, 34]] A randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-centralized differential privacy if and only if for any two neighboring datasets D, D', and any possible output $R \subseteq \mathrm{Range}(\mathcal{M})$, there is

$$\Pr[\mathcal{M}(D) = R] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(D') = R] + \delta.$$

When $\delta = 0$, $\mathcal{M}$ satisfies ϵ-centralized differential privacy.

2.1.2 Local Differential Privacy (LDP)

As we aforementioned, local differential privacy is adopted in a local mode. Compared with CDP, LDP requires more added noise to ensure privacy but does not need a trusted third party. Therefore, neighboring datasets in LDP can be any input from users. [(ϵ, δ)-local differential privacy ((ϵ, δ)-LDP) [24, 34]] A randomized mechanism $\mathcal{A}$ satisfies (ϵ, δ)-local differential privacy if and only if for any inputs v, v', and any possible output $r \subseteq \mathrm{Range}(\mathcal{A})$, there is

$$\Pr[\mathcal{A}(v) = r] \leq e^{\epsilon} \cdot \Pr[\mathcal{A}(v') = r] + \delta.$$

When $\delta = 0$, $\mathcal{A}$ satisfies ϵ-local differential privacy which is also called pure-LDP [7].

2.1.3 Composition Theorems

Under differential privacy (both CDP and LDP), there are two useful composition theorems [34]: sequential composition and parallel composition.

[Sequential Composition [34]] Given a dataset x, and two mechanisms M_1, M_2 satisfy (ϵ_1, δ_1)-DP and (ϵ_2, δ_2)-DP, respectively, the mechanism $M = (M_1(x), M_2(x))$ satisfies $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-DP.

[Parallel Composition [34]] Given a mechanism M satisfy (ϵ, δ)-DP, and the k disjoint separations of the dataset x (i.e., $x_1 \cup x_2 \cup \cdots \cup x_k = x$), the release $M(x_1), M(x_2), \cdots, M(x_k)$ satisfies (ϵ, δ)-DP.

2.1.4 Pufferfish Privacy

When providing privacy guarantees for correlated time series, differential privacy faces the challenge of excessive noise addition. Specifically, group differential privacy [23] necessitates adding $O(T)$ noise for a correlated time series with length T, leading to significant utility degradation. To address correlated data, pufferfish privacy, a generalized version of differential privacy, was proposed [35]. In addition to the privacy budget ϵ, pufferfish privacy requires three additional parameters [36]: a set of secrets $\mathcal{S}$ representing users' private data, a set of secret pairs $\mathcal{Q} \subseteq \mathcal{S} \times \mathcal{S}$ that must remain indistinguishable, and a class of data distribution Θ indicating the correlation. [ϵ-Pufferfish Privacy [36]] Give the parameters $\mathcal{S}$, $\mathcal{Q}$, and Θ, a randomized mechanism M satisfies ϵ-pufferfish privacy if $\forall \theta \in \Theta$ with $X \sim \theta$, $\forall (s_i, s_j) \in \mathcal{Q}$, $\forall w \in \mathrm{Range}(M)$, there is

$$\Pr(M(X) = w | s_i, \theta) \leq e^{\epsilon} \cdot \Pr(M(X) = w | s_j, \theta),$$

when $\Pr[\mathrm{s_i}|\theta] \neq 0$ and $\Pr[\mathrm{s_j}|\theta] \neq 0$.

2.1.5 Differential Privacy Mechanisms

Given that noise addition from a distribution is a fundamental implementation of differential privacy, this technique is extensively used for statistical estimation. Numerous applications rely on count queries, including frequency estimation, histogram estimation, and top-k item mining. Korolova et al. [37] introduced a mechanism for releasing search log histograms by adding Laplace noise. Xiao et al. [38] proposed a wavelet-based mechanism to release range count queries. In the context of LDP, Wang et al. [7] reviewed existing mechanisms for frequency estimation and introduced two optimized approaches. Li et al. [11] developed a mechanism for estimating numerical data distributions. Wang et al. [9] proposed a prefix extension mechanism to identify the top-k frequent items within a large domain. Beyond count queries, many applications focus on sum or mean queries. Wang et al. [39] devised a mechanism with optimized perturbation probability for numerical data under LDP. Xue et al. [40] introduced a mean estimation mechanism that supports personalized privacy budgets for each user. Zhou et al. [41] developed a mechanism to estimate the mean of sparse vectors. To incorporate data knowledge, Wei et al. [42] proposed an optimized mean estimation mechanism based on data distribution estimated in the initial phase under LDP. Beyond statistical estimations, data release for downstream tasks is another critical application. Ye et al. [43] proposed a mechanism to release edge information for clustering coefficient estimation. Ma et al. [44] developed a mechanism to construct decision trees under LDP, enhancing utility through the adoption of public data. Additionally, since the introduction of Differentially Private Stochastic Gradient Descent (DP-SGD) [14], numerous privacy-preserving learning-based mechanisms have been proposed under differential privacy.

Across the various scenarios, time series represent a unique research field due to their sequential nature and temporal dependencies. This adds complexity to ensuring differential privacy while maintaining data utility. In the following sections, we will introduce the concepts of time series under differential privacy.

2.2 Differential Privacy for Time Series

In this subsection, we will introduce the concept of time series and the specific definitions of differential privacy for time series, including various privacy levels. Additionally, we will provide a concise roadmap of this survey.

2.2.1 Time Series

In general, a time series is regarded an ordered sequence of values with finite length [45], while data streams are continuously generated sequences with infinite length [46]. For ease of presentation, both are referred to as “time series" in this paper, encompassing both finite and infinite settings.

[Time Series [45, 46, 47]] A time series S is an ordered sequences of values, i.e., $S = \{S_{t_1}, S_{t_2}, S_{t_3}, \cdots\}$. For simplicity, the timestamp is usually omitted, and a time series is denoted as $S = \{S_1, S_2, S_3, \cdots\}$. If any element $S_i \in \mathbb{R}$, the time series is called a univariate infinite time series. Otherwise, the time series is a multivariate infinite time series if $S_i \in \mathbb{R}^d$, namely, each element is with d dimensions.

Note that if a time series has a finite length, it is called a finite time series. Otherwise, it is referred to as an infinite time series.

2.2.2 Privacy Levels

In the context of time series, three major privacy levels have been proposed based on the privacy guarantees. Event-level privacy only protects a single element within a time series, w-event level privacy provides a privacy guarantee for a sequence of w consecutive elements, and user-level privacy protects the entire time series. The corresponding definitions are provided below, with illustrations depicted in Fig. 2.

[Event-Level Adjacent Time Series [19]] For two time series S and S', they are event-level adjacent if

1) There exists a timestamp i, $S_i \neq S'_i$;

2) For any other timestamp j, $S_j = S'_j$.

[Event-Level Privacy [19]] A randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-event-level differential privacy if and only if for any two event-level adjacent time series S, S', and any possible output $R \subseteq \text{Range}(\mathcal{M})$, there is

$$\Pr[\mathcal{M}(S) = R] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(S') = R] + \delta.$$

When $\delta = 0$, $\mathcal{M}$ satisfies ϵ-event-level differential privacy.

[w-Event Level Adjacent Time Series [19]] For two time series S and S', they are w-event level adjacent if

1) There exists w consecutive timestamp $\{k, k+1, \cdots, k+w-1\}$ and for any $i \in \{k, k+1, \cdots, k+w-1\}$, $S_i \neq S'_i$;

2) For any other timestamp j, $S_j = S'_j$.

[w-Event Level Privacy] A randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-w-event level differential privacy if and only if for any two w-event level adjacent time series S, S', and any possible output $R \subseteq \text{Range}(\mathcal{M})$, there is

$$\Pr[\mathcal{M}(S) = R] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(S') = R] + \delta.$$

When $\delta = 0$, $\mathcal{M}$ satisfies ϵ-w-event level differential privacy.

[User-Level Adjacent Time Series] For two time series S and S', they are user-level adjacent if for all timestamps $t_{u_i} = \{t_1, t_2, \cdots, t_k\}$ from any user u_i, there is $S_j \neq S'_j, \forall j \in t_{u_i}$. Note that t_{u_i} can be infinite for infinite time series.

[User-Level Privacy] A randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-user-level differential privacy if and only if for any two user-level adjacent time series S, S', and any possible output $R \subseteq \text{Range}(\mathcal{M})$, there is

$$\Pr[\mathcal{M}(S) = R] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(S') = R] + \delta.$$

When $\delta = 0$, $\mathcal{M}$ satisfies ϵ-user-level differential privacy.

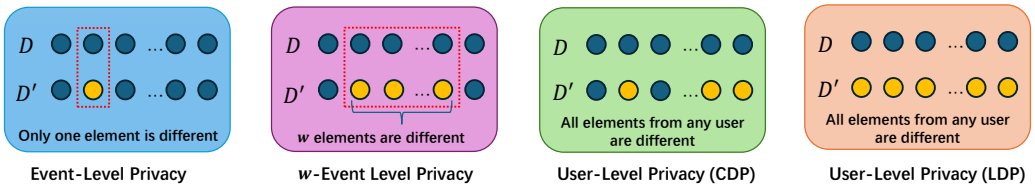

Figure 2: An illustration for privacy levels. In terms of event-level privacy, there is only one element different in the neighboring datasets. While for the w-event level privacy, there are w consecutive elements that differ in the neighboring datasets. For user-level privacy, all the elements from any user can be different in the neighboring datasets. Note that under LDP, the two user-level adjacent time series are from different users, which means the elements could be entirely different.

Obviously, event-level privacy provides the lowest level of privacy but requires the least amount of noise. Conversely, user-level privacy guarantees the strongest privacy but necessitates the largest amount of noise, which can significantly degrade utility.

2.3 Roadmap of This Survey

Since count queries and sum/mean queries are fundamental statistical operations, many advanced queries and downstream applications are derived from them. This survey begins with a review on count queries, followed by a discussion of sum/mean queries. Each subsection on these queries starts with an introduction to the concepts, followed by a review of their downstream applications. Subsequently, we introduce data release mechanisms designed to publicize data for downstream tasks. Given the popularity of location based services in time series applications, we dedicate a separate section to trajectories, reviewing the literature on location perturbation, temporal correlation issues, and trajectory release. To suggest future directions, we propose open challenges related to privacy models, temporal correlation-based attacks, complex data types, and learning-based problems. The roadmap for this survey is illustrated in Fig. 3.

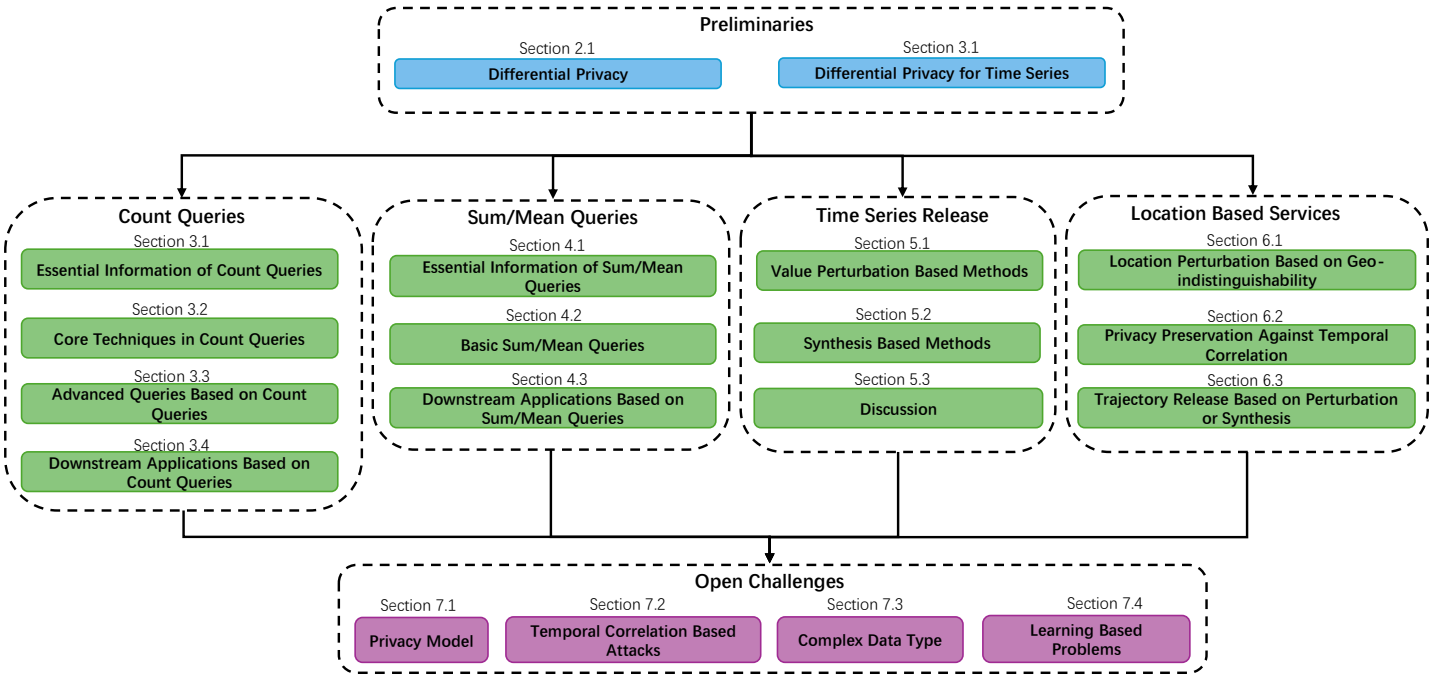

Figure 3: Roadmap of this survey.

3 Differential Privacy for Count Queries

In this section, we introduce the concept of count queries within the framework of differential privacy, a topic that has garnered substantial research attention.

3.1 Essential Information of Count Queries

For a time series S comprising categorical values with length t (t can be ∞), a count query can be formally expressed as follow:

$$\mathrm{F}_{\mathrm{cnt}}(v, S) = \sum_{i=1}^{t} \mathbf{1}_v(x_i),$$

where $\mathbf{1}_v(x_i)$ denotes the indicator function, defined below:

$$\mathbf{1}_v(x_i) := \begin{cases} 1 & \text{if } x_i = v, \\ 0 & \text{if } x_i \neq v. \end{cases}$$

Count queries are fundamental in the context of differential privacy, as they form the basis for other tasks, such as frequency and histogram estimations. Compared with other scenarios, applying count queries to time series presents unique challenges. Time series arrives as a continuous stream, necessitating the injection of a larger amount of noise to provide the desired privacy guarantees due to continuous release of query results.

3.2 Core Techniques in Count Queries

For continuously releasing a finite time series, a naive approach is to release each element with the entire privacy budget ϵ [17]. However, such a method would introduce substantial additive noise, leading to low utility with an error bound of $O(\frac{\sqrt{T}}{\epsilon})$. Dwork et al. [16] proposed the first work to handle binary time series under event-level CDP with a logarithm error bound. However, this mechanism is limited to time series with finite length T. Subsequently, Chan et al. [17] improved the mechanism to support the release of infinite binary time series. Both of these works employ a tree-based method to enhance utility. The binary tree mechanism [17], illustrated in Fig. 4, ensures that each release influences at most one node at each level for finite time series. Consequently, each node only needs to add noise corresponding to the privacy budget $\frac{\epsilon}{(\log T+1)}$. To guarantee a logarithm error bound for infinite time series [17, 48], more binary trees will be construed. Since the update of one element only influences one tree, each tree will be allocated an entire privacy budget. More details can be referred to Fig. 4.

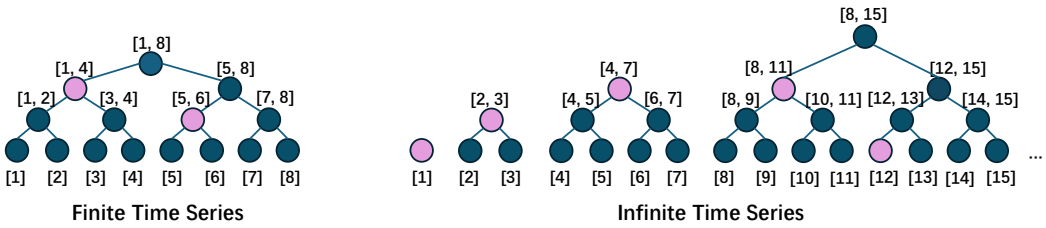

Figure 4: Here is an illustration of binary tree mechanism for time series count query under event-level CDP. For finite time series $[1, 8]$ (i.e., $T = 8$), the count query result for the range $[1, 6]$ is specified as $\hat{\mathrm{F}}_{\mathrm{cnt}}(v, [1, 6]) = \mathrm{F}_{\mathrm{cnt}}(v, [1, 4]) + \mathrm{F}_{\mathrm{cnt}}(v, [5, 6]) + 2Lap(\frac{\log T+1}{\epsilon})$, where $Lap(\cdot)$ is drawn from the Laplace distribution with zero mean. For an infinite time series, multiple binary trees will be constructed. Since a change in any single element only influences one tree, each tree will be allocated an entire privacy budget. The overall privacy budget consumption of the mechanism is ϵ, which can be calculated using parallel composition [34]. For example, $\hat{\mathrm{F}}_{\mathrm{cnt}}(v, [1, 12]) = \hat{\mathrm{F}}_{\mathrm{cnt}}(v, [1, 1]) + \hat{\mathrm{F}}_{\mathrm{cnt}}(v, [2, 3]) + \hat{\mathrm{F}}_{\mathrm{cnt}}(v, [4, 7]) + \hat{\mathrm{F}}_{\mathrm{cnt}}(v, [8, 11]) + \hat{\mathrm{F}}_{\mathrm{cnt}}(v, [12, 12])$.

In addition to the tree-based structure, another approach to handle time series under differential privacy is based on the matrix mechanism [28, 49]. Without privacy concerns, a count query $\mathcal{M}$ for binary time series x with length n can be specified as

$$\mathcal{M}(x) = Mx = \begin{bmatrix} 1 & 0 & 0 & \cdots \\ 1 & 1 & 0 & \cdots \\ 1 & 1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} x = \begin{bmatrix} x_1 \\ x_1 + x_2 \\ \vdots \\ \sum_1^n x_i \end{bmatrix}.$$

Based on this formulation, the matrix mechanism is employed to release data streams under (ϵ, δ)-DP for further error reduction. Given a workload matrix M, the strategy matrix R and reconstruction matrix L are first

constructed, denoted as $M = LR$. The strategy matrix R is utilized to pre-process the input x. After adding a Gaussian noise vector z to the processed term Rx, a post-processing step L is applied. In summary, for an input time series $x \in \mathbb{R}^n$, the matrix mechanism is denoted as

$$\mathcal{M}_{L,R}(x) = L(Rx + z),$$

where $z \sim N(0, \|R\|_{1\to 2}^2 C_{\epsilon,\delta}^2 \mathbf{I})$ to ensure (ϵ, δ)-DP, $\|R\|_{1\to 2}^2$ is the maximum of the 2-norm of the columns of the strategy matrix R. The additive mean squared error $\mathrm{err}_{\ell_2^2}$ of the matrix mechanism is

$$\mathrm{err}_{\ell_2^2}(\mathcal{M}_{L,R}, \mathcal{M}, n) = \max_{x\in\mathbb{R}^n} \mathbb{E}\left[\frac{1}{n}\|\mathcal{M}_{L,R}(x) - \mathcal{M}(x)\|_2^2\right] = \frac{1}{n} trace(L^T L)\|R\|_{1\to 2}^2 C_{\epsilon,\delta}^2.$$

Hence, the matrix mechanism can be regarded as an optimization problem. For more comprehensive information, we recommend reading the work by Henzinger et al. [49].

3.3 Advanced Queries Based on Count Queries

Query	Ref.	Type	Privacy Level	Method Primitive	Error Bound
Binary Counting	[16]	finite	event-level CDP	binary tree mechanism	$O(\frac{1}{\epsilon} \cdot (\log^{1.5} t))$
	[17]	infinite	event-level CDP	binary tree mechanism	$O(\frac{1}{\epsilon} \cdot (\log^{1.5} t))$
	[48]	infinite	user-level CDP	binary tree mechanism	$O(\frac{\kappa(D_t)}{\epsilon} \cdot \log^{1.5} t \cdot \log^{1+\theta}(\kappa(D_t)))$
	[49]	infinite	event-level (ϵ, δ)-CDP	matrix mechanism	$O(\frac{4}{\epsilon^2}(\frac{4}{9} + \ln(\frac{1}{\delta}\sqrt{\frac{2}{\pi}}))(1 + \frac{\ln(4n/5))}{\pi})^2)$
Frequency Estimation	[50]	finite	event-level zCDP	multi-branch tree	$O(\tau \log T \sqrt{2(s+1)(t-1)\log(6T/\beta)})$
	[48]	infinite	user-level CDP	binary tree mechanism	$O(\frac{\kappa(D_t)}{\epsilon\theta} \cdot \log^{1+\theta}(\kappa(D_t))) \cdot \log(tR/\beta)$
Distinct Elements Counting	[51]	finite	user-level CDP	bipartite maximum matching	$O(\frac{\ell_*}{\epsilon} \log(\frac{\ell_{\max}}{\beta})$

Table 5: A brief summary of count-based queries, where t indicates the current timestamp, T is the length of the time series, D represents the dataset, $\kappa(D_t)$ denotes the maximum number of elements contributed by any user, τ is the privacy level of zero-concentrated differential privacy (zCDP) [52], θ is any small constant, ℓ is the sensitivity (ℓ_* represents the bounded sensitivity), and β is the confidence parameter. [1]

Based on basic binary count queries, numerous other types of count queries have been proposed, including frequency estimation, frequency moment estimation, and distinct element counting. A brief summary of the literature is presented in Table 5. Cardoso et al. [50] introduced differentially private histograms in the continual observation model with an unknown domain. To facilitate practical implementation, the authors propose a mechanism that continually returns a noisy histogram by aggregating counts at each round and adding noise to them. Dong et al. [48] proposed a mechanism to estimate frequency under user-level CDP. Since a user may contribute multiple elements to the time series, the mechanism first estimates each user's contribution and then applies a truncation process to retain only a limited number of elements per user, marking subsequent items as invalid. Furthermore, the proposed approach reduces the domain of elements to further enhance the utility of frequency estimates. The work by [51] proposes a method to estimate the number of distinct elements in a time series, and obtain the bounds on the true number of unique elements. The paper models the dataset as a bipartite graph and reduces the unique counting process to a max-flow problem, allowing the utilization of standard algorithms for bipartite maximum matching to solve unique counting problem. Furthermore, Kalemaj et al. [53] proposed a mechanism that achieves a logarithmic error bound for releasing distinct elements with insertions and deletions in a finite time series, under item-level differential privacy, which considers neighboring datasets

[1] The error bound of [49] in the table is the L_2 norm error.

differing by more than one element deletion. Epasto et al. [54] presented the first work to release differentially private ℓ_p frequency moments, denoted as $\sum_i f_i^p$. Notably, when $p = 1$, the frequency moment reduces to distinct counting. Practically, Zhang et al. [55] propose DP-SQLP, the first differentially private stream aggregation processing system, which has been implemented for Google Shopping and is planned for future application to Google Trends.

3.4 Downstream Applications Based on Count Queries

The application of differential privacy in count queries primarily involves releasing histograms and monitoring anomalies. Recent research [56] has demonstrated that employing subsampling and filtering techniques can reduce the sensitivity of real-time series, thereby enhancing the utility of differentially private mechanisms applied to such data. To improve utility, these mechanisms [18, 57, 58, 59] typically employ sampling to reduce the number of elements needing protection and filtering to mitigate the impact of added noise. Additionally, some mechanisms leverage pufferfish privacy [60, 61], which introduces less noise while achieving similar privacy guarantees. The details of these mechanisms are as follows. Fan et al. [18] introduced a mechanism for collecting count query results under the FAST framework, which employs filtering and adaptive sampling techniques to satisfy CDP. In a separate work, Fan et al. [57] applied the FAST framework for anomaly detection, specifically for detecting epidemic outbreaks. Li et al. [58] proposed a mechanism that only releases the histogram when it significantly differs from previous values, with the threshold adjusted according to feedback from the control system. Wang et al. [59] proposed a framework called SecWeb, following the idea of FAST under w-event level differential privacy. In addition to adaptive sampling and filtering, SecWeb incorporates dynamic grouping and injects Laplace noise based on the groups rather than individual elements. Wang et al. [8] proposed the first mechanism to achieve almost local differential privacy (LDP) under the w-event privacy model. Their approach involves employing multiple agents to collect data from users and release sanitized data to an untrusted third party. Liang et al. [60] introduced a mechanism for releasing web browsing histograms under the pufferfish privacy framework, which is beneficial for perturbing correlated data. Their proposed mechanism includes a model to quantify privacy leakage arising from temporal correlations and presents three strategies to enhance the model's efficiency: bounding the number of secret pairs, limiting the session length, and avoiding repetitive computations. Ding et al. [61] proposed a mechanism within the framework of pufferfish privacy to make the time and occurrence of an element indistinguishable.

Based on count queries, mechanisms proposed under the local differential privacy (LDP) framework are commonly used to estimate statistics. To conserve the privacy budget under LDP, a memoization technique [62] was proposed, which stores sanitized versions of all values for further release. [63] improved upon memoization by incorporating hashing. However, memoization may leak privacy in the presence of a knowledgeable adversary who can potentially derive changes without prior knowledge. Xue et al. [64] introduced a difference tree-based mechanism that applies fresh perturbation at each timestamp under user-level privacy, enabling the aggregation of statistics without violating changing points. Additionally, [65] proposed a method to reduce the item domain, thereby enhancing utility. Beyond memoization based methods, He et al. [66] proposed a privacy budget allocation strategy to enhance the utility of frequency release under w-event level condensed local differential privacy (CLDP) [67]. In their approach, the allocated privacy budget depends on the predicted elements, determined via a proportional-integral-derivative (PID) controller. In other applications, Feng et al. [68] proposed a mechanism to estimate the distribution of infinite time series while satisfying user-level differential privacy. Their approach achieves reasonable utility by bounding privacy leakage and optimizing the allocation of the privacy budget. Li et al. [69] introduced the first work on collecting the top-k items from a time series while satisfying event-level local differential privacy and adhering to a bounded memory space constraint. Their proposed mechanisms are based on the HeavyGuardian data structure, which maintains the frequently occurring elements while evicting the infrequent ones. Additionally, Gu et al. [70] introduced a mechanism under pattern-level privacy, which is similar to w-event level privacy but does not require successions. To privately release critical patterns (i.e., subsequences

of elements in a time series), their mechanism perturbs the existence of each element, thereby providing a privacy guarantee.

4 Differential Privacy for Sum/Mean Queries

This section provides a comprehensive review of the literature concerning sum and mean queries within the differential privacy framework. It explores the concept of sum/mean queries, the basic queries, and the downstream applications.

4.1 Essential Information of Sum/Mean Queries

While preserving the occurrence of an element in a time series is important, maintaining the accuracy of the element's value is equally crucial. To ensure precise results for queries such as sum or mean, small deviations in the perturbation process are necessary. Since the mean is directly correlated to the sum, we discuss sum and mean queries together.

The sum query can be denoted as

$$\mathrm{F}_{sum}(T, S) = \sum_{i=1}^{T} S_i,$$

where T represents the timestamp for sum release and S is the corresponding time Series. The range query on sum is

$$\mathrm{F}_{rsum}((T_1, T_2), S) = \sum_{i=T_1}^{T_2} S_i,$$

where (T_1, T_2) represents the query range, and S is the corresponding time Series.

As for the mean query, it can be denoted as

$$\mathrm{F}_{mean}(T, S) = \frac{1}{T} \sum_{i=1}^{T} S_i,$$

where T represents the timestamp for mean release, and S is the corresponding time Series. Another common mean query is to release the mean at a timestamp from users' time series,

$$\mathrm{F}_{rtm}(t, S) = \frac{1}{n} \sum_{i=1}^{n} S_t^{u_i},$$

where $S_t^{u_i}$ represents the value at timestamp t from the user u_i, and n is the number of users.

4.2 Basic Sum/Mean Queries

There have proposed a line of work to release sum/mean queries under differential privacy, with a brief summary provided in Table 6. The pioneering work by Bolot et al. [71] was the first to study the continual decaying sums problem. They explored three variants: the window sum (range sum query), which releases the sum of W consecutive elements; the exponential decay sum, which releases the sum of elements weighted by an exponential function; and the polynomial sum, which releases the sum of elements weighted by a polynomial function. Henzinger et al. [72] also investigated the continual decaying sum problem. Their work introduced the use of the Gaussian mechanism for adding noise and derived tighter error bounds. In contrast, Dong et al. [48]

[2]The error bound of [72] in the table is the L_2 norm error.

Query	Ref.	Type	Privacy Level	Method Primitive	Error Bound
Sum	[48]	infinite	user-level CDP	binary tree mechanism	$O(\frac{\varphi(D_t)}{\epsilon\theta} \cdot \log^{1.5}(tR) \cdot \log^{1+\theta}(\varphi(D_t)) \cdot \log(1/\beta))$
Window Sum	[71]	finite	event-level CDP	binary tree mechanism	$O(\frac{1}{\epsilon} \log W \frac{1}{\beta})$
	[72]	finite	event-level (ϵ, δ)-CDP	matrix mechanism	$O(2\sigma^2_{\epsilon,\delta}\Delta^2(1 + \frac{\log W}{\pi} + \frac{2}{W})^2)$
Exponential Decay Sum	[71]	finite	event-level CDP	binary tree mechanism	$O(\frac{1}{\epsilon} \log \frac{\alpha}{1-\alpha} \frac{1}{\beta})$
	[72]	finite	event-level (ϵ, δ)-CDP	matrix mechanism	$O(\sigma^2_{\epsilon,\delta}\Delta^2(1 + \frac{1}{\pi} S_{T,2\alpha})^2)$
Polynomial Decay Sum	[71]	finite	event-level CDP	binary tree mechanism	$\Omega(1 - \frac{\epsilon^{c-1}}{\log^{c-1}(1/\beta)})$
	[72]	finite	event-level (ϵ, δ)-CDP	matrix mechanism	$O(\sigma^2_{\epsilon,\delta}\Delta^2(1 + \frac{H_{T,2c}-1}{4})^2)$

Table 6: The table provides a brief summary of sum-based queries, where $\varphi(D_t)$ denotes the maximum contribution from any user at time t, θ is a small constant, W is the window length, β is the confidence parameter, α indicates the exponential decay parameter, c represents the polynomial decay parameter, $H_{T,2c}$ is the generalized Harmonic sum, $S_{T,2\alpha}$ is a defined series sum with $\alpha > 1$.[2]

addressed sum queries by reducing them to count queries, as their approach could only handle the latter. For each timestamp with value x_i, they expand it into R steps, where the first $x_i(w.l.o.g., x_i < R)$ steps are filled with 1, and the remaining steps are filled with a special symbol $\perp$. However, this method requires prior knowledge of the maximum value R that can occur in the time series. [73] proposed a method for answering sum queries with a threshold under a multi-branch tree structure. The threshold is optimized based on the expected squared error between the true result and the estimated one. Their proposed mechanism cannot handle infinite time series, so they claim that most queries focus on a limited range, allowing for truncation of the infinite time series. Additionally, their work introduced the first mechanism to release time series under LDP with a threshold for value truncation. Instead of directly perturbing the values, the mechanisms proposed by Ye et al. [74, 75] perturb the temporal order. This approach makes them naturally adaptable for sum/mean queries while preserving the original values. These methods demonstrate superior performance for calculating moving averages.

4.3 Downstream Applications Based on Sum/Mean Queries

Due to utility considerations, existing mechanisms for downstream applications based on sum and mean queries primarily operate under event-level privacy or w-event level privacy. Perrier et al. [76] introduced a differentially private mechanism for publishing statistics of real-valued time series under event-level privacy, such as moving averages derived from energy data collected through smart meters. Their approach addresses scenarios where the bound on observations is either overly conservative or unknown, which is crucial for real-time monitoring applications. The proposed mechanism optimizes utility by scaling the added noise to the threshold value instead of a potentially larger bound, thereby improving accuracy. To enable real-time computation of the mean at any timestamp from users' time series under w-event LDP, Wang et al. [77] proposed sampling strategy to select important elements and a privacy budget allocation strategy according to the importance of the elements. However, the sampling process may inadvertently reveal some private information due to the intentional selection. Kurt et al. [78] proposed an online anomaly detection method for networks based on the cumulative sum algorithm, satisfying event-level (ϵ, δ)-differential privacy. Their approach adds noise to the statistic at each timestamp from each network node, operating under event-level privacy, and then derives the mean from the data of all nodes. This allows for detecting anomalies by monitoring changes in the released means.

5 Differential Privacy for Time Series Release

Time series release aims to publicly share private time series while preserving privacy. Value perturbation methods add noise to the data values, often using sampling and adaptive budget allocation for utility. While temporal perturbation methods dispatch elements across timestamps to obfuscate event timings, avoiding value distortion but risking empty releases or collisions.

5.1 Value Perturbation Based Methods

Time series release aims to directly publicize the time series for downstream tasks. Since time series release focuses on preserving the accuracy of values, privacy budget allocation is critical for controlling added noise and minimizing distortion. Therefore, a sampling-based method is often employed to select crucial elements according to the tasks, reducing the number of points requiring privacy budget allocation and enhancing utility. Additionally, to further improve the utility of the released data, a post-processing step can be adopted to correct the noisy data using prior knowledge. The outline of time series release is summarized in Fig. 5.

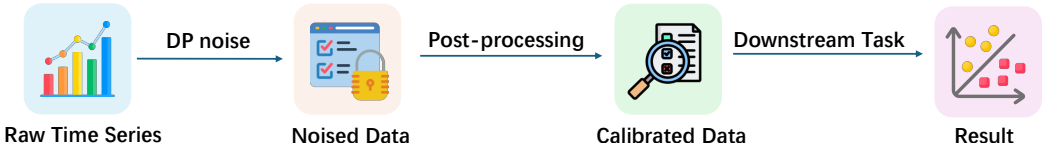

Figure 5: The outline of time series release under differential privacy.

5.1.1 Privacy Budget Allocation Strategies

User-level privacy protects the elements from any user but requires a larger privacy budget for reasonable utility, particularly challenging for time series release. Conversely, event-level privacy safeguards individual elements, yet may not suffice for comprehensive privacy guarantees. To balance the privacy levels, Kellaris et al. [19] first proposed w-event level privacy under CDP. By leveraging w-event level privacy, sanitized time series can offer better privacy guarantees than event-level privacy and higher utility than user-level privacy. To optimize the advantages of w-event level privacy, Kellaris et al. [19] proposed two privacy budget allocation strategies. Building upon the ideas of the budget distribution and absorption strategies in [19], Ren et al. [79] proposed corresponding strategies under LDP framework. To mitigate the utility degradation caused by dividing the privacy budget, the authors divide the users instead, with each user reporting only one timestamp within a w-length window. Several sampling-based methods have been developed to reduce the number of elements requiring a privacy budget, with adaptive budget allocation based on element importance. Wang et al. [80] introduced RescueDP, a scheme for real-time publishing of spatio-temporal crowd-sourced data with w-event level CDP, integrating adaptive sampling, privacy budget allocation, dynamic grouping, perturbation, and filtering techniques. The adaptive sampling component adjusts sampling rates based on data changes, ensuring efficient resource utilization. The privacy budget allocation mechanism dynamically distributes the privacy budget for sampling points across successive timestamps. Zhang et al. [81] proposed Re-DPoctor, a real-time health data releasing scheme ensuring w-event level CDP, enhancing utility with a partition algorithm safeguarding health data patterns and improving privacy through adaptive sampling and budget allocation. He et al. [66] introduced a new privacy concept using condensed local differential privacy (CLDP) [67] for w-event level privacy, aiming to enhance utility. They save privacy budget from empty releases and reallocate it to released elements, and utilize a PID controller-based method for adaptive budget allocation. However, this approach may inadvertently disclose private information through omitted empty points.

5.1.2 Optimization Strategies

Leveraging the correlations present in time series, the pre-processing or post-processing methods can be applied to improve the utility of the perturbed data. Ren et al. [82] addressed privacy challenges in high-dimensional crowdsourced data by proposing LoPub, an LDP data publication algorithm under event-level privacy. They use expectation maximization (EM) and Lasso regression to efficiently estimate multivariate joint distributions, identifying attribute correlations to reduce data dimensionality for distribution learning speed and utility improvements. Wang et al. [83] proposed the methods, LoCop and DR_LoCop, for releasing high-dimensional crowdsourced data under event-level LDP. The methods comprise four integrated components: a transformation component that ensures LDP by hashing and randomizing the data, an estimation component that infers probability distributions from the resulting Bloom filter strings, a computation component that derives marginal distributions and captures dependencies among dimensions, and a sampling component that generates a new synthetic dataset based on the computed distributions and dependencies. Fioretto and Hentenryck [84] introduced OptStream, a method for releasing time series under w-event level, which extends to handling hierarchical streams like energy profiles, making it applicable beyond its original energy domain. OptStream involves four key steps: sampling points for private measurement, perturbing them for privacy, reconstructing non-sampled points, and post-processing with convex optimization to improve accuracy by redistributing added noise. Zhang et al. [85] introduced a method for releasing differentially private sequential data using first-order autoregressive processes under user-level privacy. Their approach estimates unreleased data from previously released data by leveraging learned correlations, without requiring prior knowledge. The estimated data is combined with the observed data and perturbed with calibrated noise at each timestamp, facilitating real-time data release. Li et al. [86] proposed a framework for locally private stream data release that employs shuffling and subsampling techniques. Their approach maintains utility in the context of continual data collection by sampling a subset of users at each timestamp. An optimal sample size is determined to reduce redundant data and enhance utility, and the framework incorporates pre-processing within the shuffler to mitigate bias arising from distributed sampling. Besides pre- or post-processing methods, Bao et al. [87] proposed a mechanism based on the assumption of data fluctuation. Since time series may not change significantly over time, this mechanism formalizes the correlation between elements, allowing a later element to be represented by previous elements. To protect privacy, noise is added to the correlation.

5.2 Synthesis Based Methods

Directly releasing users' data poses significant risks of privacy breaches. An alternative approach is to train a synthesis model under strict privacy conditions and then release the data generated by this model for downstream tasks. To synthesize time series data under DP, one method involves first estimating the relevant statistics and then generating the synthetic data based on these estimations. Additionally, the advent of Generative Adversarial Networks (GANs) under DP [88, 89] has facilitated the use of deep learning algorithms to generate data, thereby enhancing both privacy protection and data accuracy.

5.2.1 Synthesis Based on Statistics

Synthesis mechanisms based on statistical features first capture the statistical characteristics from datasets under DP. Subsequently, new data is generated according to these estimated statistics. He et al. [90] introduced an efficient polynomial-time algorithm for generating online differentially private synthetic data under event-level privacy from a continuous time series within the hypercube $[0, 1]^d$. The algorithm achieves near-optimal accuracy bounds in 1-Wasserstein distance and extends previous work to include Lipschitz queries. By utilizing an online hierarchical partitioning approach and a novel Inhomogeneous Sparse Counting Algorithm, the method maintains strong privacy guarantees while ensuring high utility for infinite time horizons. To achieve a higher privacy level, Bun et al. [91] focused on generating differentially private synthetic data through statistical estimation under user-level CDP. They proposed algorithms that maintain the accuracy of fixed time window and cumulative

time queries, ensuring minimal error while preserving privacy. Their approach involves a two-stage process that combines noisy estimates with post-processing techniques to ensure consistency and accuracy in synthetic data generation.

5.2.2 Synthesis Based on Generative Models

In addition to statistics-based mechanisms, another method for synthesizing time series is through Generative Adversarial Networks (GANs). Unlike other data types, time series requires consideration of temporal correlations. Frigerio et al. [92] presented a framework for releasing high-quality open data while ensuring user privacy through DP, addressing both continuous and discrete data. By leveraging deep learning and generative models with long short-term memory networks, the framework maintains data utility and correlations, introducing innovations such as clipping decay to optimize performance. Wang et al. [93] introduced PART-GAN, a privacy-preserving generative model designed for time series augmentation and sharing. PART-GAN combines Conditional and Temporal Generative Adversarial Networks (CT-GAN) with differential privacy mechanisms, enabling the generation of unlimited synthetic data that addresses issues of incomplete and irregularly sampled time series. Torfi et al. [94] proposed a mechanism to generate high-quality synthetic health record data while ensuring privacy using Rényi Differential Privacy (RDP). Their framework combines convolutional autoencoders and convolutional generative adversarial networks (CGAN) to effectively handle both discrete and continuous data, preserving temporal and feature correlations. For specific applications, Lamp et al. [95] introduced GlucoSynth, a novel privacy-preserving GAN framework designed to generate high-quality synthetic glucose traces while maintaining strong differential privacy guarantees. By focusing on preserving the relationships among glucose events (motifs) and temporal dynamics, GlucoSynth addresses the unique challenges of synthesizing glucose data.

5.3 Discussion

The aforementioned release mechanisms modify the values of the original time series, which can degrade utility in value-critical scenarios. For elements in a time series where occurrence indicates more sensitive information, temporal perturbation can be employed to avoid distorting the original values. Ye et al. [74] first proposed a method to achieve temporal perturbation in the local setting, maintaining the privacy guarantee while enhancing utility. As illustrated in Fig. 6, unlike value perturbation that directly modifies the original values by

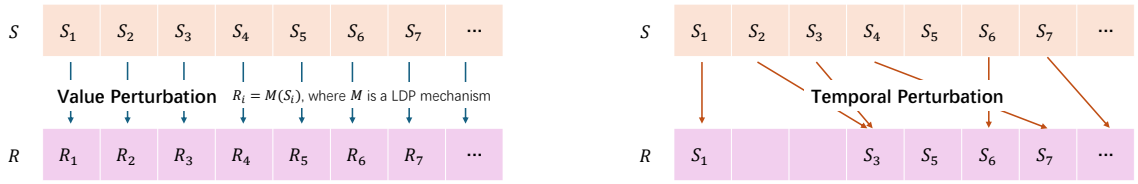

Figure 6: Value perturbation-based methods will perturb the original values by adding DP noise. In contrast, temporal perturbation-based methods will dispatch the values to corresponding timestamps for release.

adding LDP noise, temporal perturbation dispatches elements across different timestamps. This obfuscates the precise occurrence times, preventing adversaries from determining the exact event timings. However, temporal perturbation can lead to issues such as delayed releases, empty releases (where no elements are dispatched to certain timestamps), and element substitutions, resulting in missing data. To address these issues, Ye et al. [75] proposed a bi-directional perturbation mechanism that eliminates collisions during the dispatching process, ensuring that elements are only delayed. Furthermore, Mao et al. [96] extended the definition to a metric-based version tailored for anomaly detection, aiming to reduce collisions involving anomalous elements. However, these proposed mechanisms [74, 75, 96] primarily address event-level privacy concerns, leaving room for enhancements to achieve higher levels of privacy.

6 Differential Privacy for Location Based Services

Location Based Service (LBS) is a crucial application of mobile computing that provides personalized services based on users' geographical locations. These services, ranging from navigation assistance to location-based recommendations, require continuous collection and analysis of users' location data, often in the form of trajectories.

A trajectory is a specific type of time series that comprises spatial-temporal data. It can be regarded as a sequence of time-ordered points, denoted as $T = \{p_1, p_2, \cdots, p_T\}$, where p_i represents a location and T is the length of the trajectory. Compared with other time series, the correlations within trajectory data are more pronounced due to the constraints imposed by spatial variation. Regarding privacy levels for trajectory data, location privacy corresponds to event-level privacy, providing protection for individual locations, whereas trajectory privacy safeguards the entire trajectory.

In this section, we introduce mechanisms for handling trajectory data under differential privacy, organized according to utility improvement in location perturbation, privacy preservation against temporal correlation, and trajectory release. These mechanisms ensure that the privacy of individual locations and movements is preserved while maintaining the utility of the data for analysis and service provision.

6.1 Location Perturbation Based on Geo-indistinguishability

For meaningful outputs in LBS, the perturbed location should not deviate excessively from the actual one. As illustrated in Fig. 7, constraining the perturbation domain is essential for improving utility; otherwise, a large perturbation domain yields less useful results. For example, perturbing Paris to London is impractical [29]. Therefore, a metric-based privacy notion, ϵ-geo-indistinguishability, is proposed. Specifically, a user's level of privacy is defined as $\ell = \epsilon r$, where r is the radius of the perturbation domain, corresponding to r_i in Fig. 7. Here is the formal definition of geo-indistinguishability. [Geo-indistinguishability [29]] Given any two locations x and x' ($d(x, x') \leq r$), a randomized mechanism M satisfies ϵ-geo-indistinguishability iff

$$\Pr[M(x) \in Z] \leq e^{\epsilon d(x,x')} \Pr[M(x') \in Z],$$

where $Z \subseteq \mathcal{Z}$ is the possible output domain, where $d(x, x')$ represents a distance measure between x and x'.

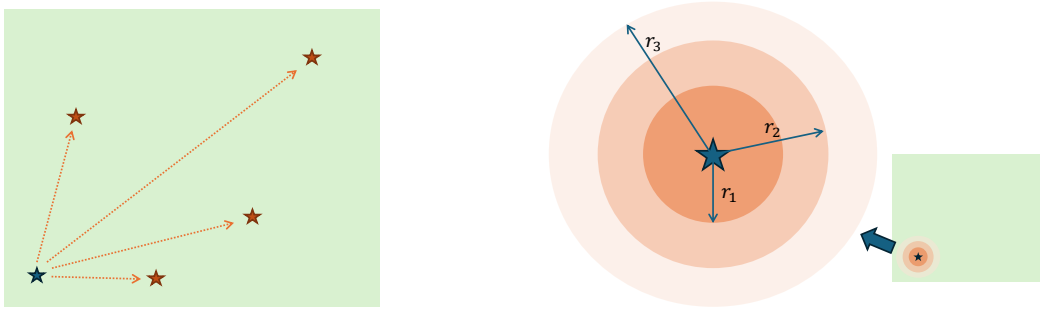

Figure 7: A traditional DP mechanism (illustrated in the left-hand figure) involves a perturbation domain (green area) for a location that is typically large, resulting in low utility for the perturbed location. To improve utility while providing useful services, a metric-based privacy notion, geo-indistinguishability, is introduced to control the perturbation (illustrated in the right-hand figure). A smaller distance r_i leads to higher utility but offers less privacy. Therefore, it is important to balance the trade off between utility and privacy when designing mechanisms under geo-indistinguishability.

Since the inception of geo-indistinguishability, numerous enhancements have been made to improve the privacy notion from various perspectives. To enhance the calculation efficiency, Bordenabe et al. [97] proposed

a method to optimize the trade-off between geo-indistinguishability and service quality in location privacy, employing linear programming to minimize service quality loss while ensuring optimal privacy guarantees. By reducing the number of constraints from cubic to quadratic, their approach significantly improves computational efficiency. Building on the concept of geo-indistinguishability, Weggenmann and Kerschbaum [98] introduced the notion of directional privacy, a relaxation of pure differential privacy that performs effectively in the local model. To enhance practical utility, Zhao et al. [99] proposed geo-ellipse-indistinguishability to protect individual location data in directional distribution analysis. This method incorporates the covariance matrix to account for dispersion and orientation of community locations, using elliptical noise instead of circular noise. The proposed mechanisms, based on gamma and multivariate normal distributions, ensure higher probabilities of randomized locations aligning with community location trends while maintaining statistical quality. Liang and Yi [100] further advanced the concept of geo-indistinguishability by introducing concentrated geo-privacy, an update from the CDP version. This approach supports advanced composition mechanisms for high-dimensional data and achieves a lower noise scale, thereby enhancing overall privacy protection while maintaining utility. Zhao and Chen [101] proposed vector-indistinguishability (vector-ind) to enhance location privacy by preserving distance and direction dependencies between successive locations. They introduced four mechanisms using Laplace and Uniform distributions to achieve vector-ind, maintaining data utility while ensuring CDP.

Several mechanisms have been proposed to address various privacy issues across numerous scenarios based on geo-indistinguishability. Yu et al. [102] proposed PIVE, a dynamic differential location privacy framework that integrates geo-indistinguishability and expected inference error to protect against inference attacks. PIVE operates in two phases: First, it identifies a protection location set based on user-defined error thresholds and prior knowledge; And then, it generates pseudo-locations within this set, ensuring differential privacy. This approach enables adaptive, personalized privacy settings tailored to individual user needs and location-based service requirements, thereby enhancing both privacy preservation and utility. Cao et al. [103] extended differential privacy to define ϵ-spatiotemporal event privacy and proposed a framework to quantify its protection level in existing location privacy-preserving mechanisms. They demonstrated their framework by adapting the Planar Laplace Mechanism for geo-indistinguishability to ensure spatiotemporal event privacy while maintaining linear computational complexity. Niu et al. [104] introduced Eclipse, a mechanism that combines geo-indistinguishability, k-anonymity, and expected inference error to protect location privacy against long-term observation attacks. Eclipse obfuscates user locations within an anonymity set, minimizing privacy leakage while maintaining service usability and correctness. Qiu et al. [105] tackled the Vehicle-based spatial crowdsourcing Location Privacy (VLP) problem, aiming to minimize travel cost distortion while preserving location privacy over road networks. They redefined geo-indistinguishability based on path distance and approximated the VLP problem as a linear programming formulation through discretization. To improve time efficiency, they proposed a two-layer optimization algorithm and analyzed the trade-off between privacy and quality of service. Haydari et al. [106] proposed a differential privacy-based map-matching algorithm (DPMM) for protecting user privacy in mobility data. DPMM generates privatized link-level location trajectories by incorporating road characteristics such as capacity and functional role. The algorithm adaptively selects the noise level based on link density, effectively balancing privacy preservation and trajectory accuracy.

6.2 Privacy Preservation Against Temporal Correlation

Due to the intrinsic features of location data, temporal correlations pose significant privacy issues when handling locations. Specifically, an adversary can infer information about a location based on its preceding or succeeding elements. For instance, Shao et al. [20] proposed iTracker, a framework designed to recover multiple trajectories from differentially private data using a structured sparsity model. iTracker leverages interdependencies among locations to enhance recovery accuracy, effectively challenging existing Laplace perturbation-based location protection mechanisms. To address privacy risks posed by temporal correlations in location data, numerous mechanisms have been proposed. Xiao and Xiong [107] introduced a solution that preserves location privacy with

rigorous differential privacy guarantees by proposing δ-location set, which accounts for temporal correlations in location data. They also introduced the sensitivity hull to capture geometric sensitivity in multidimensional space and presented the planar isotropic mechanism (PIM), an efficient location perturbation method that achieves optimal utility while meeting differential privacy requirements. Cao et al. [21] first investigated the privacy loss of CDP mechanisms under temporal correlations and introduced the concept of Temporal Privacy Leakage (TPL). They proposed an efficient algorithm to calculate TPL and designed methods to convert traditional DP mechanisms to ones that mitigate TPL, ensuring privacy over continuous data releases by bounding the leakage within a defined parameter α. Xiao et al. [22] proposed LocLok, a system that protects user locations with differential privacy by modeling temporal correlations using a hidden Markov model and applying PIM for optimal noise addition. LocLok generates possible locations via the Markov model, perturbs them with PIM, and infers true locations within a set of all possible locations, ensuring robust local privacy even when adversaries have access to historical location data. Liu et al. [108] protected location privacy by analyzing the impact of temporal-spatial correlations and proposing new privacy definitions, introducing Bayesian-based geo-indistinguishability to better evaluate and enhance privacy levels. Their method optimally allocates noise among spatially and temporally correlated locations, effectively protecting sensitive locations within a trajectory while achieving differential privacy. Ma et al. [109] proposed RPTR under w-event level CDP to protect real-time vehicle trajectory data. They employed dynamic sampling and ensemble Kalman filters, utilizing a position transfer probability matrix to infer correlations and ensure accurate predictions while balancing data availability and privacy. Additionally, they introduced a regional privacy weight mechanism to enhance protection in high-density areas, thereby ensuring higher prediction accuracy and adaptability across different scenarios. Cao et al. [110] propose a post-processing framework to enhance the utility of differentially private streaming data releases by leveraging temporal correlations. They modeled the problem as a maximum a posterior estimation, transformed it into a nonlinear constrained programming problem, and used a transition matrix to incorporate both probabilistic and deterministic constraints. Ahuja et al. [111] proposed a method to release histogram information from trajectories under user-level CDP. To enhance utility, they introduced a method based on variational autoencoders to refine the histograms by utilizing the correlations of histograms.

6.3 Trajectory Release Based on Perturbation or Synthesis

As location-based services (LBS) become increasingly integral to everyday applications, ensuring the privacy of users while maintaining the utility of location data remains a critical challenge. Various mechanisms have been proposed to address this issue, each focusing on different aspects of location privacy and data utility. Wang et al. [112] introduced L-SRR, the first LDP framework for location-based services, enhancing utility while ensuring strict privacy. The proposed staircase randomized response mechanism perturbs user locations using optimized probabilities, significantly improving utility for applications such as traffic density estimation and k-nearest neighbor queries. Cunningham [113] introduced a locally differentially private mechanism for trajectory data sharing that integrates public knowledge to enhance utility while ensuring privacy. This mechanism perturbs hierarchically-structured n-grams of trajectory data to capture spatio-temporal relationships, leveraging public data without compromising privacy. Zhang et al. [114] proposed a trajectory perturbation mechanism under user-level LDP that enhances privacy by using adjacent direction information to connect neighboring points. They introduce a two-stage pivot sampling process utilizing bi-directional clues from pivots, and an anchor-based method to restrict the spatial region of trajectories.

Synthetic trajectory generation has emerged as another promising solution, allowing for the publication of useful data without compromising individual privacy. Gursoy et al. [115] presented DP-Star, a framework for publishing trajectory data that ensures differential privacy while maintaining high utility. DP-Star normalizes raw trajectories using representative points, constructs a density-aware grid to preserve spatial densities, and employs a private Markov mobility model to maintain correlations and intra-trajectory mobility. This results in synthetic trajectory datasets that are both privacy-preserving and useful for various data mining tasks. Moreover,

Gursoy et al. [116] presented AdaTrace, a scalable location trace synthesizer that achieves statistical privacy, deterministic attack resilience, and strong utility preservation. AdaTrace generates differentially private synthetic traces through a four-phase process: feature extraction, noise injection, and utility-aware synthesis. The synthetic traces preserve utility-critical information and are robust against Bayesian inference, partial sniffing, and outlier leakage attacks, ensuring privacy without significant utility loss. Du et al. [117] introduced LDPTrace, a locally differentially private framework for synthesizing realistic trajectories with minimal computational cost and strong privacy guarantees. LDPTrace captures key movement patterns from users' trajectories, ensuring robust statistical privacy and resilience against attacks. Extensive evaluations demonstrate that LDPTrace generates authentic trajectories without external knowledge, outperforming existing methods in terms of utility and privacy protection. Hu et al. [118] introduced RetraSyn under w-event level LDP, aimed at real-time trajectory synthesis while ensuring data privacy. RetraSyn leverages mobility patterns from trajectory streams and incorporates a global mobility model, dynamic update mechanisms, and Markov-based synthesis to generate realistic trajectories. This framework effectively captures complex spatial-temporal contexts and employs adaptive privacy budget allocation strategies, ensuring authenticity and practicality in diverse real-world scenarios. Sun et al. [119] proposed SPRT, a method for synthesizing private and realistic vehicle trajectories by incorporating geographic structures into differential privacy mechanisms. SPRT constructs a geography-aware grid to capture accurate mobility patterns and defines a moveable constraint based on real-world conditions, enhancing both summary-level statistics and individual-level mobility patterns.

7 Open Challenges

Although many mechanisms have been proposed to handle time series under differential privacy, several issues still need to be addressed. In this section, the challenges will be introduced according to privacy model, potential attacks, data type, and learning based problems.

7.1 Privacy Model

The privacy model is a crucial factor in differential privacy. As aforementioned, there are three privacy levels when handling time series [19]. Event-level privacy guarantees the privacy of a single element in a time series, making it easier to implement since the sensitivity of an individual element in neighboring datasets is simpler to measure. In contrast, user-level privacy provides a higher privacy guarantee and is more practical in real-world applications. However, bounding the sensitivity of a single user's participation is challenging, making the allocation of the privacy budget more complex. Additionally, utility issues become more pronounced when dealing with infinite time series.

Several research works have explored handling infinite time series under user-level privacy with specific conditions. Dong et al. [48] introduced mechanisms for basic queries such as count and sum. These mechanisms are based on event-level privacy approaches and are adapted to user-level privacy by bounding the maximum changes of a single user in a time series. For LDP, Xue et al. [64] proposed a mechanism for count queries, but it requires that the time series does not fluctuate significantly. Feng et al. [68] proposed a strategy that randomly allocates the privacy budget according to a converging sum series.

Therefore, improving the utility of infinite time series under user-level differential privacy is an intriguing future direction. Beyond basic queries, efforts can be made to accommodate specific queries or applications. Key challenges include accurately measuring data sensitivity and effectively allocating the privacy budget. Overcoming these challenges can enhance the practical utility of differential privacy mechanisms for managing infinite time series under user-level privacy.

7.2 Temporal Correlation Based Attacks

Compared to other data types, the correlation in time series is more pronounced. Due to the inherent sequential nature of time series, each element is often directly influenced by its predecessors. Various works have employed the Markov model to capture and represent these correlations under DP [21, 22, 115]. In the context of the Markov model, an element is directly influenced only by its immediate neighboring element. Consequently, the influence of previous elements is implicitly carried forward through the chain of direct dependencies between neighboring elements. Since time series are always modeled explicitly, this simplicity can result in the model overlooking long-term information that extend beyond immediate neighbors. To capture such information, more complex models like the long short-term memory network [3] are needed. Moreover, for specific types of time series such as trajectories, public knowledge can introduce additional privacy issues. For example, certain perturbations may be impossible due to physical world limitations. In summary, compared with single data points, elements in time series are at a greater risk of privacy leakage. This suggests a potential direction for research: attacking existing privacy mechanisms by exploiting these correlations and to design new mechanisms that account for the inherent dependencies in time series.

7.3 Complex Data Type

Most current works can only handle simple time series with high utility, such as one value at each timestamp. However, real-world data are more complex, and mechanisms should be designed to handle this complexity. Here are two examples:

First, sensor data are often multi-dimensional and correlated across each dimension. This complexity requires mechanisms capable of managing and analyzing data with multiple interacting variables. Traditional methods that handle single-dimensional elements at each timestamp are insufficient for capturing the nuances of multi-dimensional sensor data. For example, environmental sensors may collect temperature, humidity, and air pressure simultaneously. Analyzing these factors independently can miss critical interactions and patterns, such as how temperature changes might influence humidity levels during users' activities. Therefore, advanced methods must be developed to process and interpret multi-dimensional sensor data effectively.

Second, the element at each timestamp can be intricate. For instance, social networks change over time, making real-time analysis of such dynamic networks complicated. Managing the evolution of relationships and interactions within the network adds a layer of complexity beyond time series analysis. Additionally, privacy concerns in such contexts extend beyond the temporal dimension to include graph privacy, encompassing node-level privacy and edge-level privacy. Mechanisms must account for these additional privacy requirements to ensure data protection while enabling real-time analysis.

7.4 Learning Based Problems

Current DP mechanisms are primarily designed for basic queries, such as count, mean, and frequency. However, time series without privacy concerns are often used for more complex downstream tasks, such as classification and clustering. These tasks require a deeper understanding and manipulation of the data, going beyond simple statistical queries. To accommodate more practical downstream tasks, it is essential to develop DP mechanisms that can support these sophisticated operations effectively, ensuring both the utility and privacy of the data.

A reasonable solution is to generate synthetic time series from the real dataset. Synthetic approach allows the fundamental features of the time series to be captured while protecting the privacy of the underlying data. Since the features of time series are complex and extend beyond basic statistics, traditional statistical methods are insufficient for capturing these intricate patterns. For example, critical patterns such as seasonal trends, cyclic behaviors, and sudden anomalies are vital in time series analysis but are not adequately addressed by basic statistical methods.

Therefore, learning-based methods are preferable for generating synthetic time series. These methods can model and replicate the complex dependencies and structures inherent in time series. For instance, the method proposed by Lamp et al. [95] for synthesizing glucose traces exemplifies how deep learning can be applied to generate realistic and privacy-preserving synthetic data. Predictably, more and more application-specific mechanisms will be proposed.

8 Conclusion

In this paper, we present a comprehensive survey on handling time series under differential privacy. We begin by introducing the basic concepts of time series and differential privacy, along with relevant definitions. Our survey starts with an exploration of two basic queries: count queries and sum/mean queries. For each query type, we first explain the concept of the basic query, then review the core techniques or developments related to the queries, and finally discuss the advanced queries derived from the basic ones. At the end of each query section, we review the downstream tasks based on these queries. Subsequently, we introduce mechanisms for time series release, categorizing them into value perturbation based methods and synthetic generation based methods. Additionally, we dedicate a separate section to location-based services (LBS), as they are common application scenarios for time series. We review relevant papers for LBS according to two popular privacy issues and the demands of trajectory release. Finally, we illustrate four open challenges and suggest future directions.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No: 62372122, 62072390, and 92270123), the Research Grants Council, Hong Kong SAR (Grant No: 15203120, 15208923 and 15210023), Shenzhen Key Laboratory of Safety and Security for Next Generation of Industrial Internet
(Grant No. ZDSYS20210623092007023) and Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation (Grant No. 2020B121201001)

References

[1] Hyndman, R. & Athanasopoulos, G. Forecasting: principles and practice. (OTexts,2018)

[2] Gardner Jr, E. Exponential smoothing: The state of the art—Part II. International Journal Of Forecasting. **22**, 637-666 (2006)

[3] Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W. & Woo, W. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. Advances In Neural Information Processing Systems. **28** (2015)

[4] C. Dwork, "Differential privacy," in International colloquium on automata, languages, and programming. Springer, 2006, pp. 1–12.

[5] Ye, Q. & Hu, H. Local differential privacy: Tools, challenges, and opportunities. International Conference On Web Information Systems Engineering. pp. 13-23 (2020)

[6] M. Yang, T. Guo, T. Zhu, I. Tjuawinata, J. Zhao, and K.-Y. Lam, "Local differential privacy and its applications: A comprehensive survey," Computer Standards & Interfaces, p. 103827, 2023.

[7] Wang, T., Blocki, J., Li, N. & Jha, S. Locally differentially private protocols for frequency estimation. 26th USENIX Security Symposium (USENIX Security 17). pp. 729-745 (2017)

[8] Wang, Z., Pang, X., Chen, Y., Shao, H., Wang, Q., Wu, L., Chen, H. & Qi, H. Privacy-preserving crowd-sourced statistical data publishing with an untrusted server. IEEE Transactions On Mobile Computing. **18**, 1356-1367 (2018)

[9] Wang, T., Li, N. & Jha, S. Locally differentially private heavy hitter identification. IEEE Transactions On Dependable And Secure Computing. **18**, 982-993 (2019)

[10] Fu, Y., Ye, Q., Du, R. & Hu, H. Collecting Multi-type and Correlation-Constrained Streaming Sensor Data with Local Differential Privacy. ACM Transactions On Sensor Networks. (2023)

[11] Li, Z., Wang, T., Lopuhaä-Zwakenberg, M., Li, N. & Škoric, B. Estimating numerical distributions under local differential privacy. Proceedings Of The 2020 ACM SIGMOD International Conference On Management Of Data. pp. 621-635 (2020)

[12] Ye, Q., Hu, H., Au, M., Meng, X. & Xiao, X. Towards locally differentially private generic graph metric estimation. 2020 IEEE 36th International Conference On Data Engineering (ICDE). pp. 1922-1925 (2020)

[13] Zhang, J., Zhang, Z., Xiao, X., Yang, Y. & Winslett, M. Functional mechanism: Regression analysis under differential privacy. ArXiv Preprint ArXiv:1208.0219. (2012)

[14] Abadi, M., Chu, A., Goodfellow, I., McMahan, H., Mironov, I., Talwar, K. & Zhang, L. Deep learning with differential privacy. Proceedings Of The 2016 ACM SIGSAC Conference On Computer And Communications Security. pp. 308-318 (2016)

[15] Fu, J., Ye, Q., Hu, H., Chen, Z., Wang, L., Wang, K. & Xun, R. DPSUR: Accelerating Differentially Private Stochastic Gradient Descent Using Selective Update and Release. ArXiv Preprint ArXiv:2311.14056. (2023)

[16] Dwork, C., Naor, M., Pitassi, T. & Rothblum, G. Differential privacy under continual observation. Proceedings Of The Forty-second ACM Symposium On Theory Of Computing. pp. 715-724 (2010)

[17] Chan, T., Shi, E. & Song, D. Private and continual release of statistics. ACM Transactions On Information And System Security (TISSEC). **14**, 1-24 (2011)

[18] Fan, L. & Xiong, L. An adaptive approach to real-time aggregate monitoring with differential privacy. IEEE Transactions On Knowledge And Data Engineering. **26**, 2094-2106 (2013)

[19] Kellaris, G., Papadopoulos, S., Xiao, X. & Papadias, D. Differentially private event sequences over infinite streams. Proc. VLDB Endow.. **7**, 1155-1166 (2014,8), https://doi.org/10.14778/2732977.2732989

[20] Shao, M., Li, J., Yan, Q., Chen, F., Huang, H. & Chen, X. Structured sparsity model based trajectory tracking using private location data release. IEEE Transactions On Dependable And Secure Computing. **18**, 2983-2995 (2020)

[21] Cao, Y., Yoshikawa, M., Xiao, Y. & Xiong, L. Quantifying differential privacy under temporal correlations. 2017 IEEE 33rd International Conference On Data Engineering (ICDE). pp. 821-832 (2017)

[22] Xiao, Y., Xiong, L., Zhang, S. & Cao, Y. Loclok: Location cloaking with differential privacy via hidden markov model. Proceedings Of The VLDB Endowment. **10**, 1901-1904 (2017)

[23] Dwork, C., Roth, A. & Others The algorithmic foundations of differential privacy. Foundations And Trends® In Theoretical Computer Science. **9**, 211-407 (2014)

[24] Zhao, Y. & Chen, J. A survey on differential privacy for unstructured data content. ACM Computing Surveys (CSUR). **54**, 1-28 (2022)

[25] Zhao, Y., Du, J. & Chen, J. Scenario-based Adaptations of Differential Privacy: A Technical Survey. ACM Computing Surveys. **56**, 1-39 (2024)

[26] Miranda-Pascual, À., Guerra-Balboa, P., Parra-Arnau, J., Forné, J. & Strufe, T. SoK: Differentially private publication of trajectory data. Proceedings On Privacy Enhancing Technologies. (2023)

[27] Katsomallos, M., Tzompanaki, K. & Kotzinos, D. Privacy, space and time: A survey on privacy-preserving continuous data publishing. Journal Of Spatial Information Science. **2019**, 57-103 (2019)

[28] Li, C., Miklau, G., Hay, M., McGregor, A. & Rastogi, V. The matrix mechanism: optimizing linear counting queries under differential privacy. The VLDB Journal. **24** pp. 757-781 (2015)

[29] Andrés, M., Bordenabe, N., Chatzikokolakis, K. & Palamidessi, C. Geo-indistinguishability: Differential privacy for location-based systems. Proceedings Of The 2013 ACM SIGSAC Conference On Computer & Communications Security. pp. 901-914 (2013)

[30] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," Advances in Neural Information Processing Systems, vol. 30, 2017.

[31] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in Proceedings of the 2014 ACM SIGSAC conference on computer and communications security, 2014, pp. 1054–1067.

[32] A. G. Thakurta, A. H. Vyrros, U. S. Vaishampayan, G. Kapoor, J. Freudiger, V. R. Sridhar, and D. Davidson, "Learning new words," Mar. 14 2017, uS Patent 9,594,741.

[33] C. Dwork, "Differential Privacy and the US Census," in Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI symposium on principles of database systems, 2019, pp. 1–1.

[34] Li, N., Lyu, M., Su, D. & Yang, W. Differential privacy: From theory to practice. (Springer,2017)

[35] Kifer, D. & Machanavajjhala, A. Pufferfish: A framework for mathematical privacy definitions. ACM Transactions On Database Systems (TODS). **39**, 1-36 (2014)

[36] Song, S., Wang, Y. & Chaudhuri, K. Pufferfish privacy mechanisms for correlated data. Proceedings Of The 2017 ACM International Conference On Management Of Data. pp. 1291-1306 (2017)
[37] Korolova, A., Kenthapadi, K., Mishra, N. & Ntoulas, A. Releasing search queries and clicks privately. Proceedings Of The 18th International Conference On World Wide Web. pp. 171-180 (2009)
[38] Xiao, X., Wang, G. & Gehrke, J. Differential privacy via wavelet transforms. IEEE Transactions On Knowledge And Data Engineering. **23**, 1200-1214 (2010)
[39] Wang, N., Xiao, X., Yang, Y., Zhao, J., Hui, S., Shin, H., Shin, J. & Yu, G. Collecting and analyzing multidimensional data with local differential privacy. 2019 IEEE 35th International Conference On Data Engineering (ICDE). pp. 638-649 (2019)
[40] Xue, Q., Zhu, Y. & Wang, J. Mean estimation over numeric data with personalized local differential privacy. Frontiers Of Computer Science. **16** pp. 1-10 (2022)
[41] Zhou, M., Wang, T., Chan, T., Fanti, G. & Shi, E. Locally differentially private sparse vector aggregation. 2022 IEEE Symposium On Security And Privacy (SP). pp. 422-439 (2022)
[42] Wei, F., Bao, E., Xiao, X., Yang, Y. & Ding, B. AAA: an Adaptive Mechanism for Locally Differential Private Mean Estimation. ArXiv Preprint ArXiv:2404.01625. (2024)
[43] Ye, Q., Hu, H., Au, M., Meng, X. & Xiao, X. LF-GDPR: A framework for estimating graph metrics with local differential privacy. IEEE Transactions On Knowledge And Data Engineering. **34**, 4905-4920 (2020)
[44] Ma, Y., Zhang, H., Cai, Y. & Yang, H. Decision tree for locally private estimation with public data. Advances In Neural Information Processing Systems. **36** (2024)
[45] Esling, P. & Agon, C. Time-series data mining. ACM Computing Surveys (CSUR). **45**, 1-34 (2012)
[46] Silva, J., Faria, E., Barros, R., Hruschka, E., Carvalho, A. & Gama, J. Data stream clustering: A survey. ACM Computing Surveys (CSUR). **46**, 1-31 (2013)
[47] Ruiz, A., Flynn, M., Large, J., Middlehurst, M. & Bagnall, A. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining And Knowledge Discovery. **35**, 401-449 (2021)
[48] Dong, W., Luo, Q. & Yi, K. Continual Observation under User-level Differential Privacy. 2023 IEEE Symposium On Security And Privacy (SP). pp. 2190-2207 (2023)
[49] Henzinger, M., Upadhyay, J. & Upadhyay, S. Almost tight error bounds on differentially private continual counting. Proceedings Of The 2023 Annual ACM-SIAM Symposium On Discrete Algorithms (SODA). pp. 5003-5039 (2023)
[50] Cardoso, A. & Rogers, R. Differentially private histograms under continual observation: Streaming selection into the unknown. International Conference On Artificial Intelligence And Statistics. pp. 2397-2419 (2022)
[51] Knop, A. & Steinke, T. Counting Distinct Elements Under Person-Level Differential Privacy. 37th Conference On Neural Information Processing Systems (NeurIPS 2023). (2023)
[52] Bun, M. & Steinke, T. Concentrated differential privacy: Simplifications, extensions, and lower bounds. Theory Of Cryptography Conference. pp. 635-658 (2016)
[53] Kalemaj, I., Jain, P., Raskhodnikova, S., Sivakumar, S. & Smith, A. Counting Distinct Elements in the Turnstile Model with Differential Privacy under Continual Observation. Advances In Neural Information Processing Systems, NeurIPS 2023. (2023)
[54] Epasto, A., Mao, J., Medina, A., Mirrokni, V., Vassilvitskii, S. & Zhong, P. Differentially private continual releases of streaming frequency moment estimations. ArXiv Preprint ArXiv:2301.05605. (2023)
[55] Zhang, B., Doroshenko, V., Kairouz, P., Steinke, T., Thakurta, A., Ma, Z., Apte, H. & Spacek, J. Differentially Private Stream Processing at Scale. ArXiv Preprint ArXiv:2303.18086. (2023)
[56] Koga, T., Meehan, C. & Chaudhuri, K. Privacy amplification by subsampling in time domain. International Conference On Artificial Intelligence And Statistics. pp. 4055-4069 (2022)
[57] Fan, L. & Xiong, L. Differentially private anomaly detection with a case study on epidemic outbreak detection. 2013 IEEE 13th International Conference On Data Mining Workshops. pp. 833-840 (2013)
[58] Li, H., Xiong, L., Jiang, X. & Liu, J. Differentially private histogram publication for dynamic datasets: an adaptive sampling approach. Proceedings Of The 24th ACM International On Conference On Information And Knowledge Management. pp. 1001-1010 (2015)
[59] Wang, Q., Lu, X., Zhang, Y., Wang, Z., Qin, Z. & Ren, K. Secweb: Privacy-preserving web browsing monitoring with w-event differential privacy. Security And Privacy In Communication Networks: 12th International Conference, SecureComm 2016, Guangzhou, China, October 10-12, 2016, Proceedings 12. pp. 454-474 (2017)

[60] Liang, W., Chen, H., Liu, R., Wu, Y. & Li, C. A pufferfish privacy mechanism for monitoring web browsing behavior under temporal correlations. Computers & Security. **92** pp. 101754 (2020)

[61] Ding, J., Ghosh, A., Sarkar, R. & Gao, J. Publishing Asynchronous Event Times with Pufferfish Privacy. 2022 18th International Conference On Distributed Computing In Sensor Systems (DCOSS). pp. 53-60 (2022)

[62] Erlingsson, Ú., Pihur, V. & Korolova, A. Rappor: Randomized aggregatable privacy-preserving ordinal response. Proceedings Of The 2014 ACM SIGSAC Conference On Computer And Communications Security. pp. 1054-1067 (2014)

[63] Ding, B., Kulkarni, J. & Yekhanin, S. Collecting telemetry data privately. Advances In Neural Information Processing Systems. **30** (2017)

[64] Xue, Q., Ye, Q., Hu, H., Zhu, Y. & Wang, J. DDRM: A continual frequency estimation mechanism with local differential privacy. IEEE Transactions On Knowledge And Data Engineering. (2022)

[65] Arcolezi, H., Pinzón, C., Palamidessi, C. & Gambs, S. Frequency estimation of evolving data under local differential privacy. ArXiv Preprint ArXiv:2210.00262. (2022)

[66] He, Y., Wang, F., Deng, X., Ni, J., Feng, J. & Liu, S. Ordinal data stream collection with condensed local differential privacy. 2022 IEEE 24th Int Conf On High Performance Computing & Communications; 8th Int Conf On Data Science & Systems; 20th Int Conf On Smart City; 8th Int Conf On Dependability In Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys). pp. 562-569 (2022)

[67] Gursoy, M., Tamersoy, A., Truex, S., Wei, W. & Liu, L. Secure and utility-aware data collection with condensed local differential privacy. IEEE Transactions On Dependable And Secure Computing. **18**, 2365-2378 (2019)

[68] Feng, S., Mohammady, M., Wang, H., Li, X., Qin, Z. & Hong, Y. DPI: Ensuring Strict Differential Privacy for Infinite Data Streaming. ArXiv Preprint ArXiv:2312.04738. (2023)

[69] Li, X., Liu, W., Lou, J., Hong, Y., Zhang, L., Qin, Z. & Ren, K. Local differentially private heavy hitter detection in data streams with bounded memory. Proceedings Of The ACM On Management Of Data. **2**, 1-27 (2024)

[70] Gu, H., Plagemann, T., Benndorf, M., Goebel, V. & Koldehofe, B. Differential Privacy for Protecting Private Patterns in Data Streams. 2023 IEEE 39th International Conference On Data Engineering Workshops (ICDEW). pp. 118-124 (2023)

[71] Bolot, J., Fawaz, N., Muthukrishnan, S., Nikolov, A. & Taft, N. Private decayed predicate sums on streams. Proceedings Of The 16th International Conference On Database Theory. pp. 284-295 (2013)

[72] Henzinger, M., Upadhyay, J. & Upadhyay, S. A unifying framework for differentially private sums under continual observation. Proceedings Of The 2024 Annual ACM-SIAM Symposium On Discrete Algorithms (SODA). pp. 995-1018 (2024)

[73] Wang, T., Chen, J., Zhang, Z., Su, D., Cheng, Y., Li, Z., Li, N. & Jha, S. Continuous release of data streams under both centralized and local differential privacy. Proceedings Of The 2021 ACM SIGSAC Conference On Computer And Communications Security. pp. 1237-1253 (2021)

[74] Ye, Q., Hu, H., Li, N., Meng, X., Zheng, H. & Yan, H. Beyond value perturbation: Local differential privacy in the temporal setting. IEEE INFOCOM 2021-IEEE Conference On Computer Communications. pp. 1-10 (2021)

[75] Ye, Q., Hu, H., Huang, K., Au, M. & Xue, Q. Stateful switch: Optimized time series release with local differential privacy. IEEE INFOCOM 2023-IEEE Conference On Computer Communications. pp. 1-10 (2023)

[76] Perrier, V., Asghar, H. & Kaafar, D. Private continual release of real-valued data streams. ArXiv Preprint ArXiv:1811.03197. (2018)

[77] Wang, Z., Liu, W., Pang, X., Ren, J., Liu, Z. & Chen, Y. Towards pattern-aware privacy-preserving real-time data collection. IEEE INFOCOM 2020-IEEE Conference On Computer Communications. pp. 109-118 (2020)

[78] Kurt, M., Yılmaz, Y., Wang, X. & Mosterman, P. Online privacy-preserving data-driven network anomaly detection. IEEE Journal On Selected Areas In Communications. **40**, 982-998 (2022)

[79] Ren, X., Shi, L., Yu, W., Yang, S., Zhao, C. & Xu, Z. LDP-IDS: Local differential privacy for infinite data streams. Proceedings Of The 2022 International Conference On Management Of Data. pp. 1064-1077 (2022)

[80] Wang, Q., Zhang, Y., Lu, X., Wang, Z., Qin, Z. & Ren, K. RescueDP: Real-time spatio-temporal crowd-sourced data publishing with differential privacy. IEEE INFOCOM 2016-The 35th Annual IEEE International Conference On Computer Communications. pp. 1-9 (2016)

[81] Zhang, J., Liang, X., Zhang, Z., He, S. & Shi, Z. Re-DPoctor: Real-time health data releasing with w-day differential privacy. GLOBECOM 2017-2017 IEEE Global Communications Conference. pp. 1-6 (2017)

[82] Ren, X., Yu, C., Yu, W., Yang, S., Yang, X., McCann, J. & Philip, S. LoPub : high-dimensional crowdsourced data

publication with local differential privacy. IEEE Transactions On Information Forensics And Security. **13**, 2151-2166 (2018)

[83] Wang, T., Yang, X., Ren, X., Yu, W. & Yang, S. Locally private high-dimensional crowdsourced data release based on copula functions. IEEE Transactions On Services Computing. **15**, 778-792 (2019)

[84] Fioretto, F. & Van Hentenryck, P. Optstream: Releasing time series privately. Journal Of Artificial Intelligence Research. **65** pp. 423-456 (2019)

[85] Zhang, X., Khalili, M. & Liu, M. Differentially private real-time release of sequential data. ACM Transactions On Privacy And Security. **26**, 1-29 (2022)

[86] Li, X., Cao, Y. & Yoshikawa, M. Locally Private Streaming Data Release with Shuffling and Subsampling. 2023 IEEE 39th International Conference On Data Engineering Workshops (ICDEW). pp. 125-131 (2023)

[87] Bao, E., Yang, Y., Xiao, X. & Ding, B. CGM: an enhanced mechanism for streaming data collection with local differential privacy. Proceedings Of The VLDB Endowment. **14**, 2258-2270 (2021)

[88] Xie, L., Lin, K., Wang, S., Wang, F. & Zhou, J. Differentially private generative adversarial network. ArXiv Preprint ArXiv:1802.06739. (2018)

[89] Jordon, J., Yoon, J. & Van Der Schaar, M. PATE-GAN: Generating synthetic data with differential privacy guarantees. International Conference On Learning Representations. (2018)

[90] He, Y., Vershynin, R. & Zhu, Y. Online Differentially Private Synthetic Data Generation. ArXiv Preprint ArXiv:2402.08012. (2024)

[91] Bun, M., Gaboardi, M., Neunhoeffer, M. & Zhang, W. Continual Release of Differentially Private Synthetic Data from Longitudinal Data Collections. Proceedings Of The ACM On Management Of Data. **2**, 1-26 (2024)

[92] Frigerio, L., Oliveira, A., Gomez, L. & Duverger, P. Differentially private generative adversarial networks for time series, continuous, and discrete open data. ICT Systems Security And Privacy Protection: 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings 34. pp. 151-164 (2019)

[93] Wang, S., Rudolph, C., Nepal, S., Grobler, M. & Chen, S. PART-GAN: Privacy-preserving time-series sharing. Artificial Neural Networks And Machine Learning–ICANN 2020: 29th International Conference On Artificial Neural Networks, Bratislava, Slovakia, September 15–18, 2020, Proceedings, Part I 29. pp. 578-593 (2020)

[94] Torfi, A., Fox, E. & Reddy, C. Differentially private synthetic medical data generation using convolutional GANs. Information Sciences. **586** pp. 485-500 (2022)

[95] Lamp, J., Derdzinski, M., Hannemann, C., Linden, J., Feng, L., Wang, T. & Evans, D. GlucoSynth: Generating Differentially-Private Synthetic Glucose Traces. Advances In Neural Information Processing Systems. **36** (2024)

[96] Mao, Y., Ye, Q., Wang, Q. & Hu, H. Utility-Aware Time Series Data Release with Anomalies under TLDP. IEEE Transactions On Mobile Computing. (2023)

[97] Bordenabe, N., Chatzikokolakis, K. & Palamidessi, C. Optimal geo-indistinguishable mechanisms for location privacy. Proceedings Of The 2014 ACM SIGSAC Conference On Computer And Communications Security. pp. 251-262 (2014)

[98] Weggenmann, B. & Kerschbaum, F. Differential privacy for directional data. Proceedings Of The 2021 ACM SIGSAC Conference On Computer And Communications Security. pp. 1205-1222 (2021)

[99] Zhao, Y., Yuan, D., Du, J. & Chen, J. Geo-ellipse-indistinguishability: community-aware location privacy protection for directional distribution. IEEE Transactions On Knowledge And Data Engineering. (2022)

[100] Liang, Y. & Yi, K. Concentrated geo-privacy. Proceedings Of The 2023 ACM SIGSAC Conference On Computer And Communications Security. pp. 1934-1948 (2023)

[101] Zhao, Y. & Chen, J. Vector-indistinguishability: location dependency based privacy protection for successive location data. IEEE Transactions On Computers. (2023)

[102] Yu, L., Liu, L. & Pu, C. Dynamic Differential Location Privacy with Personalized Error Bounds.. NDSS. (2017)

[103] Cao, Y., Xiao, Y., Xiong, L. & Bai, L. PriSTE: from location privacy to spatiotemporal event privacy. 2019 IEEE 35th International Conference On Data Engineering (ICDE). pp. 1606-1609 (2019)

[104] Niu, B., Chen, Y., Wang, Z., Li, F., Wang, B. & Li, H. Eclipse: Preserving differential location privacy against long-term observation attacks. IEEE Transactions On Mobile Computing. **21**, 125-138 (2020)

[105] Qiu, C., Squicciarini, A., Pang, C., Wang, N. & Wu, B. Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability. IEEE Transactions On Mobile Computing. **21**, 2436-2450 (2020)

[106] Haydari, A., Chuah, C., Zhang, M., Macfarlane, J. & Peisert, S. Differentially private map matching for mobility trajectories. Proceedings Of The 38th Annual Computer Security Applications Conference. pp. 293-303 (2022)

[107] Xiao, Y. & Xiong, L. Protecting locations with differential privacy under temporal correlations. Proceedings Of The 22nd ACM SIGSAC Conference On Computer And Communications Security. pp. 1298-1309 (2015)

[108] Liu, B., Zhu, T., Zhou, W., Wang, K., Zhou, H. & Ding, M. Protecting privacy-sensitive locations in trajectories with correlated positions. 2019 IEEE Global Communications Conference (GLOBECOM). pp. 1-6 (2019)

[109] Ma, Z., Zhang, T., Liu, X., Li, X. & Ren, K. Real-time privacy-preserving data release over vehicle trajectory. IEEE Transactions On Vehicular Technology. **68**, 8091-8102 (2019)

[110] Cao, X., Cao, Y., Pappachan, P., Nakamura, A. & Yoshikawa, M. Differentially Private Streaming Data Release Under Temporal Correlations via Post-processing. IFIP Annual Conference On Data And Applications Security And Privacy. pp. 184-200 (2023)

[111] Ahuja, R., Zeighami, S., Ghinita, G. & Shahabi, C. A Neural Approach to Spatio-Temporal Data Release with User-Level Differential Privacy. Proceedings Of The ACM On Management Of Data. **1**, 1-25 (2023)

[112] Wang, H., Hong, H., Xiong, L., Qin, Z. & Hong, Y. L-srr: Local differential privacy for location-based services with staircase randomized response. Proceedings Of The 2022 ACM SIGSAC Conference On Computer And Communications Security. pp. 2809-2823 (2022)

[113] Cunningham, T., Cormode, G., Ferhatosmanoglu, H. & Srivastava, D. Real-world trajectory sharing with local differential privacy. ArXiv Preprint ArXiv:2108.02084. (2021)

[114] Zhang, Y., Ye, Q., Chen, R., Hu, H. & Han, Q. Trajectory data collection with local differential privacy. ArXiv Preprint ArXiv:2307.09339. (2023)

[115] Gursoy, M., Liu, L., Truex, S. & Yu, L. Differentially private and utility preserving publication of trajectory data. IEEE Transactions On Mobile Computing. **18**, 2315-2329 (2018)

[116] Gursoy, M., Liu, L., Truex, S., Yu, L. & Wei, W. Utility-aware synthesis of differentially private and attack-resilient location traces. Proceedings Of The 2018 ACM SIGSAC Conference On Computer And Communications Security. pp. 196-211 (2018)

[117] Du, Y., Hu, Y., Zhang, Z., Fang, Z., Chen, L., Zheng, B. & Gao, Y. Ldptrace: Locally differentially private trajectory synthesis. Proceedings Of The VLDB Endowment. **16**, 1897-1909 (2023)

[118] Hu, Y., Du, Y., Zhang, Z., Fang, Z., Chen, L., Zheng, K. & Gao, Y. Real-Time Trajectory Synthesis with Local Differential Privacy. ArXiv Preprint ArXiv:2404.11450. (2024)

[119] Sun, X., Ye, Q., Hu, H., Wang, Y., Huang, K., Wo, T. & Xu, J. Synthesizing realistic trajectory data with differential privacy. IEEE Transactions On Intelligent Transportation Systems. (2023)