Verifying Differentially Private Median Estimation

Hyukjun Kwon, Chenglin Fan
Department of Computer Science and Engineering
Seoul National University
{todd4, fanchenglin}@snu.ac.kr

June 13, 2025

Abstract

Differential Privacy (DP) is a robust privacy guarantee that is widely employed in private data analysis today, finding broad application in domains such as statistical query release and machine learning. However, DP achieves privacy by introducing noise into data or query answers, which malicious actors could exploit during analysis. To address this concern, we propose the first verifiable differentially private median estimation scheme based on zk-SNARKs. Our scheme combines the exponential mechanism and a utility function for median estimation into an arithmetic circuit, leveraging a scaled version of the inverse cumulative distribution function (CDF) method for precise sampling from the distribution derived from the utility function. This approach not only ensures privacy but also provides a mechanism to verify that the algorithm achieves DP guarantees without revealing sensitive information in the process.

1 Introduction

Modern digital services collect personal fine-grained information more than ever—from our GPS location status and health data to shopping and public transportation usage history. These kinds of information collections help tailor convenient services to our needs, such as personalized healthcare and automatic product recommendations at online store. However, they often work in exchange for our privacy Zang and Bolot [2011]. Our personal information remains at risk while we use modern, convenience-driven services. Companies have attempted to protect privacy by anonymizing user data before analysis. However, even anonymized datasets can reveal personal information. For instance, in the 2006 Netflix Prize competition, Netflix released a dataset of 100 million movie ratings from 500,000 anonymized users. Narayanan and Shmatikov [2008] showed that such anonymization was insufficient: using statistical de-anonymization techniques, they re-identified users based on approximate knowledge of a few ratings and timestamps.

To counter such vulnerabilities, the framework of Differential Privacy (DP) was introduced [Dwork et al., 2006b, Dwork and Roth, 2014]. DP provides formal privacy guarantees by adding carefully calibrated random noise to data or query outputs, ensuring that the inclusion or exclusion of a single individual's data has only a limited effect on the overall result. This protects against a wide range of inference attacks and has been adopted in various domains, including optimization [Gupta et al., 2010], reinforcement learning [Vietri et al., 2020], and deep learning [Abadi et al., 2016]. However, DP introduces a new challenge: since the output is intentionally randomized, it becomes difficult to determine whether a privacy-preserving computation was executed correctly. A malicious party could distort outputs and attribute discrepancies to DP noise, undermining trust in the system. This

opens up an integrity gap—while DP protects privacy, it does not inherently guarantee correctness or transparency of the computation.

To bridge this gap, researchers have explored the concept of *Verifiable Differential Privacy*, which enables third parties to verify whether a computation satisfies DP without compromising privacy. These methods often leverage cryptographic tools such as zero-knowledge proofs (ZKPs) [Goldwasser et al., 1985], particularly zk-SNARKs [Bitansky et al., 2012] (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge), to prove compliance with DP guarantees without revealing sensitive information. ZKP is a cryptographic method in which one party (the Prover) can prove to another party (the Verifier) that they know a piece of information or that a statement is true without revealing any information about the piece of information itself.

Additionally, Local Differential Privacy (LDP) [Kasiviswanathan et al., 2011, Evfimievski et al., 2003] offers an alternative model where users perturb their data locally before sharing it, eliminating the need for a trusted data curator. While LDP inherently provides stronger individual privacy, it often comes at the cost of degraded utility. Verifiability in LDP has also been explored by combining local randomizers with cryptographic proofs, enabling users to prove that they applied a valid LDP mechanism [Kato et al., 2021, Garrido et al., 2022, Bontekoe et al., 2024]. These verifiable LDP schemes are naturally suited to categorical queries via randomized response [Warner, 1965, Ambainis et al., 2003], but they incur substantial computational overheads and often remain impractical for large-scale deployment.

In summary, while verifiable LDP protocols support categorical queries, their high computational cost and limited utility reduce practical applicability. On the other hand, in the central DP setting, verifiability has largely been limited to numerical queries (e.g., via the Laplace mechanism), and no general verifiable mechanism has been proposed for categorical queries that require the exponential mechanism.

This work aims to close that gap by designing a verifiable exponential mechanism for categorical data in the central DP setting, combining cryptographic soundness with practical efficiency.

Our Contributions. We extend verifiable differential privacy in the central setting to categorical queries by introducing the first verifiable Exponential Mechanism (EM) for median estimation using zk-SNARKs. Our contributions are as follows:

1. **Circuit Design for Verifiable Exponential Mechanism:** We construct an arithmetic circuit for the exponential mechanism tailored to median estimation. Our design includes modular subcircuits that collectively implement a scaled inverse CDF sampling algorithm over the distribution defined by the utility function and EM.

2. **Public Verifiability:** We integrate zk-SNARKs with the Poseidon hash function to enable any public or private Verifier to efficiently confirm that the reported median was sampled correctly, without malicious tampering.

3. **Fully Non-Interactive Protocol:** We ensure non-interactivity by leveraging a public bulletin board and non-interactive randomness supplied by the Verifier, removing the need for any online interaction between Prover and Verifier.

4. **Formal Guarantees:** We provide formal proofs establishing the security, differential privacy, and strong utility guarantees of our proposed scheme.

5. **Practical Implementation:** We implement our scheme using Groth16 zk-SNARKs and Poseidon hash function, and demonstrate its practicality by showing that, for datasets of size up to 7,000, proof generation completes within a few minutes and verification takes less than a second.

1.1 Related Works

Differential Privacy (DP) was first introduced by Dwork et al. [2006b,a] as a robust framework for protecting individual privacy in statistical data analysis. To implement DP for numerical queries, they proposed the *Laplace Mechanism* Dwork et al. [2006b], which adds noise drawn from the Laplace distribution to query results. The magnitude of the noise is carefully calibrated to the sensitivity of the query to ensure a formal privacy guarantee while retaining utility.

To address categorical queries, McSherry and Talwar [2007] proposed the *Exponential Mechanism*. Instead of perturbing the output directly, this mechanism introduces a utility function over possible outputs and samples a response from a distribution biased toward higher-utility outcomes. This enables privacy-preserving data analysis in scenarios where outputs are not numeric.

Verifiable Differential Privacy (VDP) extends DP by enabling third parties to verify whether a computation satisfies differential privacy. This concept was first formalized by Narayan et al. [2015], though their approach did not support categorical queries and lacked experimental validation. To provide cryptographic guarantees for DP enforcement, Biswas and Cormode [2023] proposed using *zero-knowledge proofs* (ZKPs), particularly zk-SNARKs, to verify differentially private computations. Their interactive protocol focuses on counting queries by combining binomial noise with homomorphic commitments. This allows privacy-preserving proofs of compliance without revealing any sensitive data. However, their method is limited to simple counting queries. To broaden applicability, Wei et al. [2025] introduced a zk-SNARK-based proof system for verifying the *Laplace Mechanism*, making it possible to verify approximate DP for general numerical queries. Their method still does not support categorical queries or mechanisms like the Exponential Mechanism.

Verifiable Local Differential Privacy (VLDP): LDP was studied in early works such as Kasiviswanathan et al. [2011] and Evfimievski et al. [2003]. Under LDP, each user perturbs their own data before sharing it with the aggregator. While the randomized response [Warner, 1965], [Ambainis et al., 2003] is naturally suitable for categorical queries, verifiable LDP protocols incur a significant overhead compared to central DP protocols: each participant must produce a zero-knowledge proof that guarantees correct execution of the local DP mechanism, which results in a substantial computational and temporal cost. Moreover, since the noise is injected independently by each user rather than centrally, local DP inherently provides weaker utility guarantees than its central-model counterpart. More recently, Kato et al. [2021] introduced verifiable randomization methods for binary Randomized Response (RR), allowing each user to generate a zero-knowledge proof that their RR perturbation was executed correctly. Garrido et al. [2022] proposed a verifiable DP polling scheme, where each participant engages in the protocol by applying verifiable local DP to his data and submitting the data with a zkSNARK proof that the LDP perturbation was done correctly. Most recently, Bontekoe et al. [2024] presented a fully verifiable DP scheme in the local and shuffle model for k-ary randomized response (kRR).

1.2 Technique Overview of Our Approach

This paper addresses the challenge of verifying that a differentially private median estimation is both *accurate* and *private*. To achieve this, the authors combine the exponential mechanism (used for DP in categorical queries) with zk-SNARKs, a form of zero-knowledge proof. The probabilistic sampling of the exponential mechanism is encoded into a static arithmetic circuit, which enforces correct execution by design. Precomputed lookup tables approximate exponential values to support sampling within finite fields, while inverse CDF sampling ensures that the output adheres to the desired DP distribution. By generating a succinct proof (via zk-SNARKs) that the circuit was executed faithfully, any third party can verify that the reported median satisfies differential privacy

guarantees—without accessing the raw data or compromising privacy.

2 Preliminaries

In this section, we introduce the notation and background concepts used throughout this paper. We begin by fixing our basic notation and formally defining differential privacy. We then define the median query and establish the notation of relevant rank function and utility function. Next, we give a concise overview of zero-knowledge proofs. After that, we review cryptographic hash functions, which is an essential building block for designing the scheme that ensures input authenticity. Finally, we describe inverse CDF sampling, which serves as our primary mechanism for generating differentially private outputs.

2.1 Notations

Let $\mathcal{X}$ be the set of possible input categories. $\mathcal{D} \in \mathbb{N}^{|\mathcal{X}|}$ denotes a database of m records. We define "Adjacent Databases" as two databases that differ in exactly one record and denote $\mathcal{D} \sim \mathcal{D}'$. $\mathcal{R}$ denotes the range of possible outputs of our mechanism or query. $\mathcal{M}$ denotes a mechanism or query that maps a database to an element of $\mathcal{R}$. We denote the utility function of the exponential mechanism as $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \to \mathbb{R}$. We assume $\mathbb{F}_p$ is a finite field of order prime p.

2.2 Differential Privacy

Differential Privacy (in short, DP) guarantees that the distribution of the outputs of a mechanism regarding any two adjacent databases cannot differ more than a certain amount. First introduced by Dwork et al., DP has become a standard for data privacy preservation. For standard analysis of DP, parameters ε and δ are used.

Definition 2.1 ((ε, δ)-Differential Privacy [Dwork et al., 2006a])**.** *A mechanism $\mathcal{M}$ satisfies (ε, δ)-DP if for all $\mathcal{D} \sim \mathcal{D}'$ and all $S \subseteq \mathcal{R}$:*

$$\Pr[\mathcal{M}(\mathcal{D}) \in S] \leq \exp(\varepsilon) \cdot \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta$$

Smaller ε means stronger privacy. If $\delta = 0$, we call it Pure Differential Privacy. On the other hand, if $\delta > 0$, we call it Approximate Differential Privacy. We achieve DP by adding some statistical noise to the returned value of the query, directly or indirectly. In this process, the sensitivity of the query determines how much noise should be added to guarantee (ε, δ)-DP.

Definition 2.2 (ℓ_1-Sensitivity)**.** *The ℓ_1-sensitivity of a query function f is defined by:*

$$\Delta f = \max_{\mathcal{D} \sim \mathcal{D}'} \|f(\mathcal{D}) - f(\mathcal{D}')\|_1$$

2.3 Exponential Mechanism

Exponential Mechanism (EM) is a DP mechanism that is specialized in categorical queries. EM rates each element r in the range of the query by a utility function $u(\mathcal{D}, r)$, and outputs an element $r' \in \mathcal{R}$ with probability proportional to the exponential of the utility score of r'. The followings are the formal definition of the sensitivity of Utility Function and resulting Exponential Mechanism.

Definition 2.3 (Sensitivity of Utility Function)**.** *The sensitivity of a utility function u is defined by:*

$$\Delta u = \max_{r \in \mathcal{R}} \max_{\mathcal{D} \sim \mathcal{D}'} \left| u(\mathcal{D}, r) - u(\mathcal{D}', r) \right|_1$$

Definition 2.4 (Exponential Mechanism McSherry and Talwar [2007])**.** *The Exponential Mechanism $\mathcal{M}$ selects and outputs an element $r \in \mathcal{R}$ with probability as follows:*

$$\Pr(\mathcal{M}(\mathcal{D}, u, \mathcal{R}) = r') = \frac{\exp\left(\frac{u(\mathcal{D},r')}{2\Delta u}\right)}{\sum_{r\in\mathcal{R}} \exp\left(\frac{u(\mathcal{D},r)}{2\Delta u}\right)}$$

Theorem 2.1. *The Exponential Mechanism of Definition 2.4 satisfies $(\varepsilon, 0)$-DP.*

2.4 Median Query

The definition of Median Query is as follows.

Definition 2.5 (Median Query)**.**

$$\mathit{Median}(\mathcal{D}) = \begin{cases} x_{\frac{m+1}{2}} & \text{if } m \text{ is odd} \\ \frac{1}{2}(x_{\frac{m}{2}} + x_{\frac{m}{2}+1}) & \text{if } m \text{ is even} \end{cases}$$

Let us consider constructing a mechanism that when given a database, outputs a differentially private median. In this paper, we use the exponential mechanism for DP median estimation. First, we define the rank function $\text{rank}_{\mathcal{D}}(r)$ and the utility function $u(\mathcal{D}, r)$ as follows.

Definition 2.6 (Definition of rank and utility function for DP median estimation)**.**

$$\mathit{rank}_D(r) = |\{x \in D \mid x < r\}|, \qquad u(\mathcal{D}, r) = -\left|\mathit{rank}_{\mathcal{D}}(r) - \frac{n-1}{2}\right|$$

Lemma 2.1 (Sensitivity of $u(\mathcal{D}, r)$)**.** *The sensitivity of $u(\mathcal{D}, r)$ is 1. In other words, $\Delta u = 1$.*

Due to Lemma 2.1 and Theorem 2.1, we propose the following DP median estimation mechanism $\mathcal{M}_{median}$ based on exponential mechanism.

Definition 2.7 (Differentially Private Median Estimation)**.** *The DP median estimation $\mathcal{M}_{median}(\mathcal{D}, u, \mathcal{R})$ outputs an element $r' \in \mathcal{R}$ with probability proportional to $\exp\left(\frac{\varepsilon \cdot u(\mathcal{D},r')}{2}\right)$ where the utility function u is defined as Definition 2.6.*

2.5 Zero-Knowledge Proofs

Zero-Knowledge Proofs (ZKP) is a technique to prove that a statement is true without leaking any other information about the statement other than the fact that the statement is true. Formally, a zero knowledge proof system (P, V) must satisfy the following properties.

Definition 2.8 (Security Definitions of ZKPGoldwasser et al. [1985])**.** *The security definition that a ZKP must satisfy is as follows.*

- *Completeness: If the statement is true, an honest P convinces V with high probability.*
- *Soundness: Any cheating P fails to convince V with high probability.*
- *Zero-Knowledge: V learns nothing beyond the fact that the statement is true.*

2.6 The Definition of zk-SNARKs

zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) form a subclass of zero-knowledge proofs distinguished by their non-interactivity and succinctness. The first practical instantiation, Pinocchio, was presented at IEEE Symposium on Security and Privacy, and since then a variety of zk-SNARK constructions have been proposed and deployed. In our implementation, we adopt Groth16 protocol as our zk-SNARK of choice. Groth16 is known for extremely succinct proofs that consists of only 3 group elements, fast verification and proof generation.

Major zk-SNARK protocols, including Groth16, accept as input an arithmetic circuit or an equivalent Rank-1 Constraint System (R1CS). The Prover translates the statement to be proven into such a circuit (or R1CS) and supplies a witness, which is an assignment to the circuit's wires that satisfies every constraint exactly when the statement is true. The proof system then compiles this representation into algebraic objects (e.g., multivariate polynomials). Verification consists of evaluating these polynomials at randomly chosen points and checking their consistency with the public inputs and the provided proof. The Schwartz–Zippel lemma underpins the soundness guarantee by bounding the probability that an incorrect witness nevertheless satisfies all evaluations to a negligible amount.

Lemma 2.2 (Schwartz–Zippel Lemma)**.** *Let $P(x_1, x_2, ..., x_n) \in \mathbb{F}[x_1, x_2, ..., x_n]$ be a non-zero polynomial of degree d, where $\mathbb{F}$ is a field.*

Let $S \subseteq \mathbb{F}$ be a finite subset. Then:

$$\Pr_{r_1,...,r_n \in S}[P(r_1, \ldots, r_n) = 0] \leq \frac{d}{|S|}$$

where $r_1, ..., r_n$ are chosen independently and uniformly at random from S.

2.7 Hash Function

Hash Function is an efficiently computable function that maps an input of arbitrary length to an output of fixed length. In this paper, we assume that every hash function we use is a cryptographic hash. Also, later we prove the security of our scheme in Random Oracle Model defined as follows.

Definition 2.9 (Cryptographic Hash Function)**.** *Cryptographic Hash Function is a hash function $H : \mathcal{X} \to \mathcal{Y}$ that satisfies the following properties.*

- *Collision Resistance: It is infeasible to find $x \neq x' \in \mathcal{X}$, such that $H(x) = H(x')$.*
- *Preimage Resistance: Given $y \in \mathcal{Y}$, it is infeasible to find $x \in \mathcal{X}$ such that $H(x) = y$.*
- *Second-preimage Resistance: Given $x \in \mathcal{X}$, it is infeasible to find $x' \in \mathcal{X}$ such that $H(x) = H(x')$.*

Definition 2.10 (Random Oracle Model)**.** *For each new input $x \in \mathcal{X}$, hash function $H : \mathcal{X} \to \mathcal{Y}$ outputs a uniformly random output from $\mathcal{Y}$. And for any repeated input $x' \in \mathcal{X}$, the hash function outputs the same value from previous sampling.*

Assumption of cryptographic hash function and the Random Oracle Model is crucial in guaranteeing the soundness and zero-knowledge of our scheme.

2.8 Inverse CDF Sampling

Our scheme performs Inverse CDF Sampling from the distribution derived from the exponential mechanism. The definition of Inverse CDF Sampling method is as follows.

Definition 2.11 (Inverse CDF Sampling)**.** *Assume that the CDF of a distribution D is F_D. To sample from the distribution D, draw $U \sim Uniform(0,1)$ and output $Y = F_D^{-1}(U)$.*

3 Security Model

With the foundational tools in place, we define the roles and security assumptions in our verifiable differential privacy framework. The setting involves three roles—**Data Providers**, a **Data Analyst (Prover)**, and a **Verifier**—each with distinct objectives. All parties have private randomness, and communication between each Data Provider and the Analyst occurs over authenticated, confidential channels.

Data Providers represent the statistical population. Before protocol execution, each provider independently samples private randomness r_i and posts a commitment $\mathsf{hash}(x_i, r_i)$ to a public bulletin board, where x_i is the provider's private input. When the protocol begins, each provider securely transmits (x_i, r_i) to the Data Analyst via an encrypted and authenticated channel. We adopt an honest-but-non-colluding model: every Data Provider follows the protocol exactly as specified and does not collaborate with any other party. Modeling collusion, strategic deviation, or dishonest behavior among providers is outside the scope of this work.

The **Data Analyst** functions as the principal adversary in our verifiable differential privacy framework. The Analyst may act maliciously, with the intent to submit incorrect statistical outputs or generate counterfeit proofs that could mislead the Verifier. As such, the central goal of our protocol design is to ensure that no Data Analyst can produce a falsified output or proof that deceives the Verifier into accepting an invalid result.

The **Verifier** plays a critical role in ensuring trust in the published output. Its purpose is to validate both the statistical result and the proof generated by the Data Analyst. We model the Verifier as an active adversary capable of inspecting all available protocol transcripts in an effort to learn more than what the differentially private output reveals. To mitigate this risk, our construction ensures that the protocol satisfies a strong zero-knowledge property, thereby guaranteeing that the Verifier learns nothing beyond the validity of the published output.

We now present our main technical contribution: the design and implementation of a verifiable exponential mechanism for median estimation. This includes a high-level overview of the protocol, the architecture of the arithmetic circuit, and the core design choices that enable correctness and efficiency.

4 Verifiable Exponential Mechanism

In this section, we describe our construction of a Verifiable Exponential Mechanism for median queries. We begin by outlining the high-level framework of the scheme and detailing the overall architecture of the principal circuit. We then proceed to introduce each submodule which serves as a foundational building block of the complete circuit. The precise designs of the submodules are shown in the figures in the next section.

4.1 Conceptual Framework

In the following, let m and n be positive integers. Let $\mathbb{F}_p$ denote the finite field of prime order p over which our arithmetic circuit is defined. Let $hash : \{0,1\}^* \to \mathbb{F}_p$ be a cryptographic hash function.

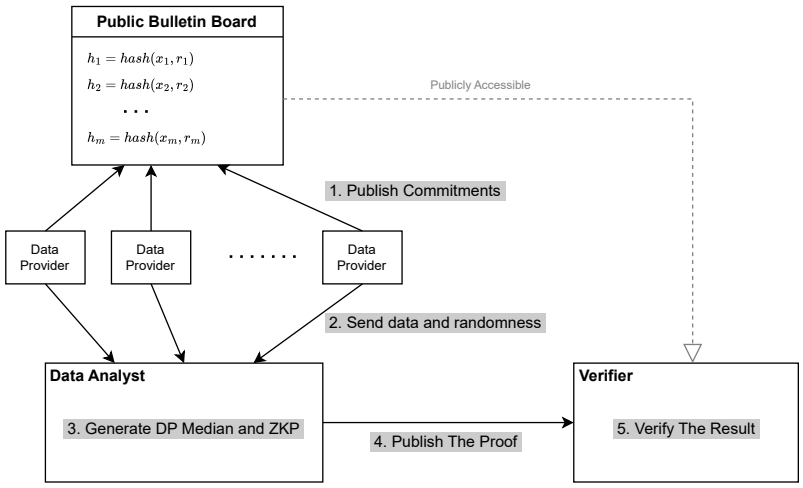

Figure 1: Framework

The set $\{0,1\}^*$ denotes an input space of arbitrary length. The overall architecture of the scheme is illustrated in Figure 1.

The protocol involves m Data Providers and a single Data Analyst; the number of Verifiers remains unspecified due to the scheme's public verifiability. For each $i \in [1, m]$, the i-th Data Provider holds a private integer input $x_i \in [0, n)$ and independent randomness value $r_i \in \mathbb{F}_p$. The objective is to compute a differentially private estimation of the median of the set $\{x_1, .., x_m\}$. The whole scheme consists of three phases.

Commitment Phase. Each Data Provider commits to its private input x_i by publishing $h_i = hash(x_i, r_i)$ on a public bulletin board.

Execution Phase. Each Data Provider transmits (x_i, r_i) to the Data Analyst over a secure channel. The Data Analyst aggregates these inputs and applies the verifiable exponential mechanism for median estimation. After the proof generation, the Data Analyst publishes both the perturbed statistics and a corresponding zero-knowledge proof generated with the underlying zk-SNARK protocol.

Verification Phase. Any Verifier can validate the published median with the proof by (i) confirming that the commitments output by the circuit match those posted on the public bulletin board, (ii) checking the range of the query matches the public inputs of the circuit, and (iii) checking the proof's validity. Since all materials required for verification are publicly accessible, the scheme achieves public verifiability.

4.2 The Main Circuit

The architecture of the main circuit is illustrated in Figure 2. It is composed of five submodules—`utility`, `expLookup`, `inverseCDF`, `inputBinder`, and `mod`. The public and private inputs of the main circuit are specified as follows.

- $range[0 : n-1]$: The range of the median query (Public)
- $input[0 : m-1]$: The inputs from the Data Providers (Private)
- $rand[0 : m-1]$: The randomness values from the Data Providers (Private)

The outputs of the main circuit are specified as follows. All of the output are public.

- $commitment[0 : m-1]$: The commitments to the inputs of Data Providers
- $DP\ median$: The final median estimation value

Implementation Challenges and Design Decisions. Implementing the exponential mechanism inside a static arithmetic circuit presents two main obstacles. First, the circuit must perform verifiable sampling according to a distribution derived from the utility function, yet it cannot execute dynamic branching or loops. Second, standard stochastic methods like rejection sampling rely on data-dependent control flow, which is incompatible with the fixed-depth nature of static circuits.

To overcome these issues, we employ a scaled inverse cumulative distribution function (CDF) sampling technique. By precomputing a lookup table of scaled exponential values and encoding the sampling process as a small, fixed-depth sequence of comparisons and multiplications, we avoid any dynamic branching. This design ensures that (i) the sampling faithfully follows the distribution, and (ii) all operations remain within the static circuit model.

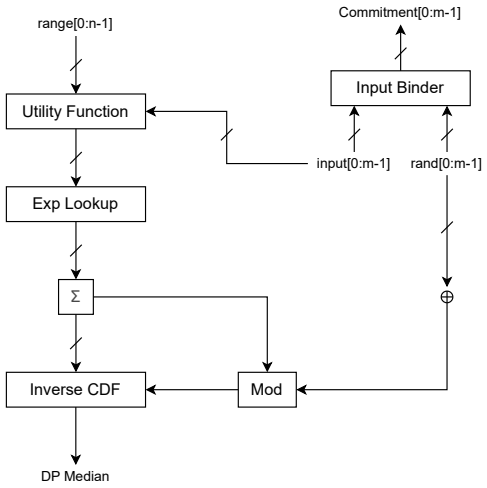

Figure 2: Main Circuit

The arithmetic circuits for zk-SNARKs are defined over a finite field, infinite accuracy on real-valued exponentials is impossible. Therefore, we propose a conceptual method to divide the lookup table into two parts. The first part is a physical lookup table t. This is the area that works like a normal lookup table that approximates relatively "large" exponentials that can be approximated into an integer. The other part is filled with all 0 (method ***set0***) or all $k = \left\lceil \frac{1}{\exp(\varepsilon/2)-1} \right\rceil$ (method ***setk***). We call the whole infinite-sized lookup table as the conceptual lookup table T. Further details about the lookup table are provided in the next subsection.

4.3 Implementation Details

4.3.1 `inputBinder` Module

The `inputBinder` Module gets inputs and associated randomnesses $\{(x_i, r_i)\}_{i=1}^{m}$ from the m Data Providers and outputs m commitments $commitment[i] = h_i$ that binds each provider's data. During verification, the Verifier confirms that each output of this module is consistent with the commitment that is posted on the public bulletin board. Without this submodule, a malicious Data Analyst

could substitute arbitrary values for some x_i and still pass the verification procedure. `inputBinder` module prevents such attacks by implementing each commitment as

$$h_i = hash(x_i, r_i)$$

using a collision-resistant hash function. The circuit-level structure of `inputBinder` is illustrated in Figure 3.

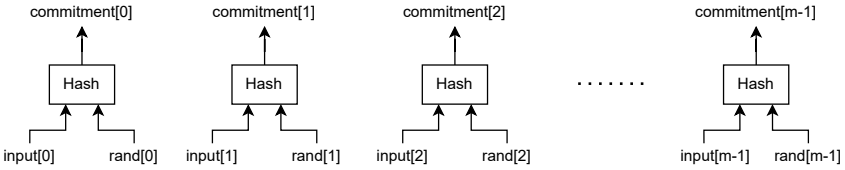

Figure 3: `inputBinder` Module

4.3.2 `utility` Module

The `utility` module computes, for every candidate output $y \in \mathcal{R} = [0, n]$, the calibrated utility score for each Data Provider's input x_i. The module implements the utility function given in Definition 2.6 and does some calibration to make the highest utility score be always 0. We implement the `utility` module with $m \cdot n$ comparators and n `abs` submodule that calculates the absolute value of the input. After that, each absolute utility score goes into the `sub_min` submodule, which gets n absolute utility scores as the inputs and calculates the minimum of them, and finally subtract the minimum value from each inputs to output the final calibrated utility score. The overall architecture is illustrated in Figure 4 and Figure 5.

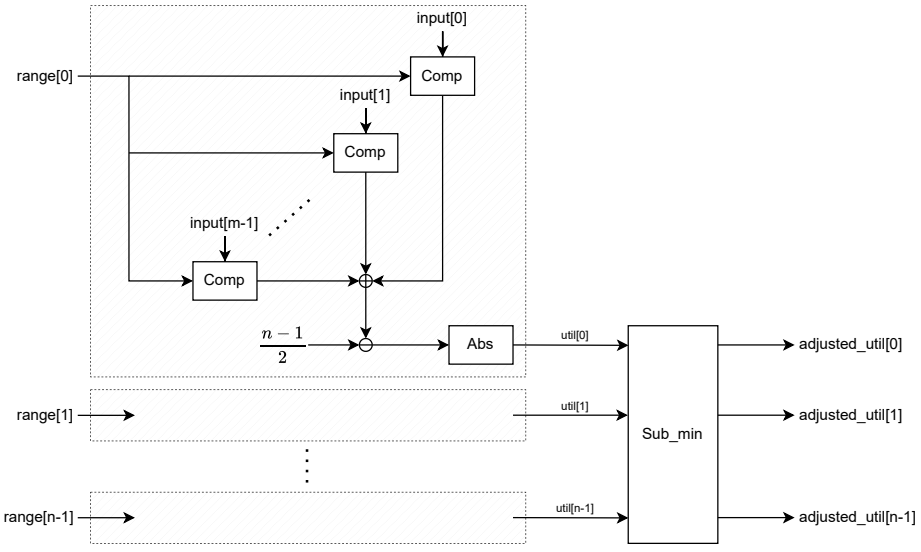

Figure 4: `utility` Module

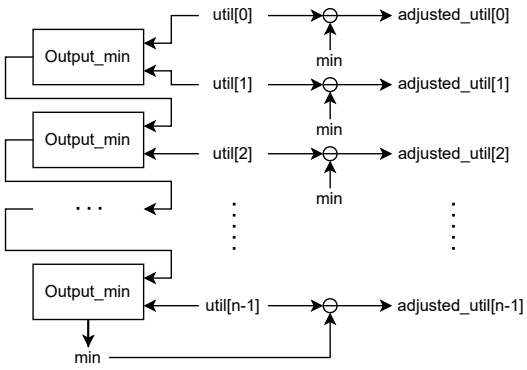

Figure 5: `sub_min` Module

4.3.3 `expLookup` Module

Since our circuit is defined on a finite field $\mathbb{F}_p$, infinite accuracy on real-valued exponentials is impossible. Therefore, we propose a conceptual method to divide the lookup table into two parts. The first part is a physical lookup table t. This is the area that works like a normal lookup table that approximates relatively "large" exponentials that can be approximated into an integer modulo p. The other part is filled with all 0 (method ***set0***) or all $k = \left\lceil \frac{1}{\exp(\varepsilon/2)-1} \right\rceil$ (method ***setk***). We call the whole infinite-sized lookup table as the conceptual lookup table T.

Let the size of the physical lookup table t be l, which means that the first l entries are the same as the physical lookup table. Then, each entry of the conceptual lookup table is filled as follows.

$$T[i] = \begin{cases} \left\lfloor \exp(\frac{\varepsilon}{2}) \cdot T[i+1] \right\rfloor & \text{if } i < l-1 \\ k & \text{if } i = l-1 \\ 0 & \text{if } i \geq l \text{ and } \textbf{\textit{set0}} \\ k & \text{if } i \geq l \text{ and } \textbf{\textit{setk}} \end{cases}$$

By the definition of $T[i]$, we can easily see the ratio of each adjacent entries does not exceed $\exp(\varepsilon/2)$ in the physical lookup table, and this works as a crucial property to achieve differential privacy.

The `expLookup` Module is implemented with a comparator, several simple arithmetic gates, and a $l:1$ Multiplexer as in Figure 6 where the right circuit demonstrates ***set0*** method, and the left circuit demonstrates ***setk*** method.

4.3.4 `inverseCDF` Module

The `inverseCDF` modules consumes the cumulative exponential weights $\{c_i\}$ computed over the candidate outputs, a non-manipulable randomness seed ρ, and the range $\mathcal{R}$ of the query. It returns the differentially private median by performing a scaled inverse-CDF sampling procedure. Define u_i as the exponential of the calibrated utility score for range$[i]$. The procedure can then be described logically as follows.

1. Each c_i are calculated as follows.

$$c_i = \begin{cases} u_0 & \text{if } i = 0 \\ c_{i-1} + u_i & \text{if } i > 0 \end{cases}$$

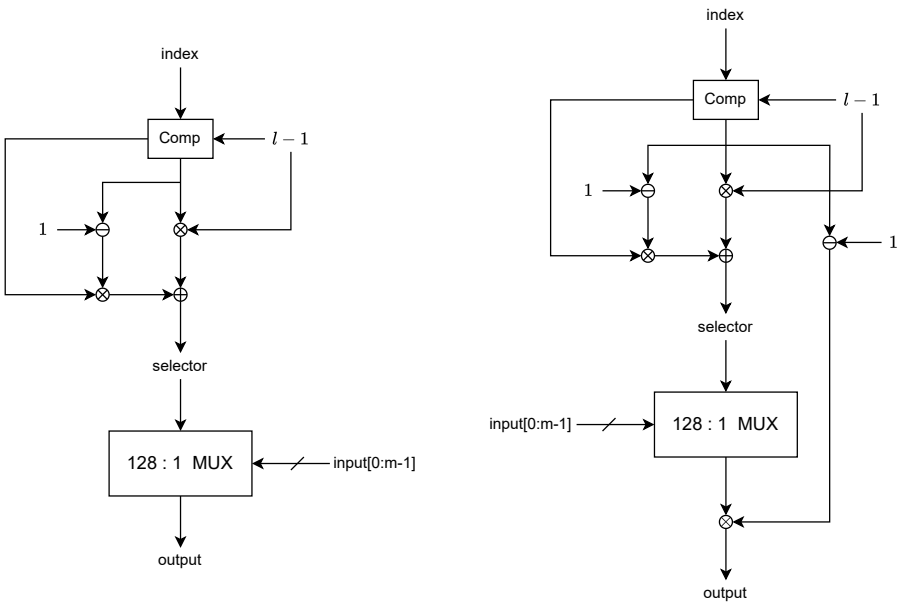

Figure 6: `expLookup` Module initiated with $l = 128$

2. Use the randomness ρ (interpreted as a uniform integer in $[0, c_{n-1})$) to select the unique index j satisfying

$$c_j - 1 \;\leq\; \rho \;<\; c_j.$$

3. Output $range[j]$ as the DP median.

The overall circuit-level architecture is illustrated in Figure 7.

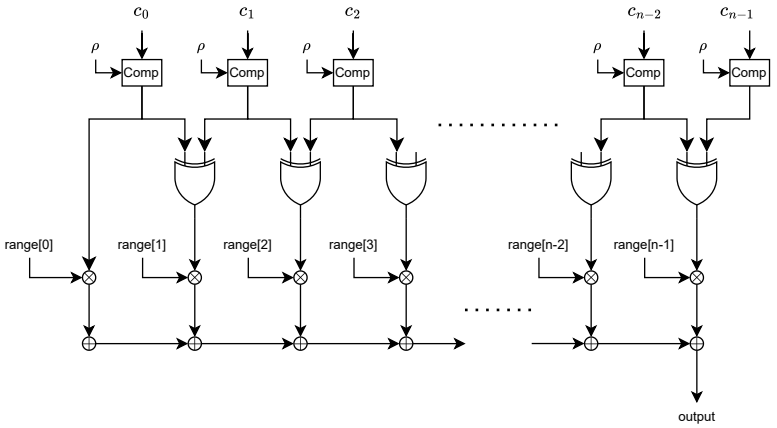

Figure 7: `inverseCDF` Module: this module performs inverse CDF sampling from the distribution defined by the exponential mechanism.

4.3.5 `mod` Module

The `mod` module receives the sum of randomnesses from the Data Providers and c_{n-1} that is defined in the `inverseCDF` module. The module outputs the result of modulo reduction of the sum of randomnesses. The output is guaranteed to be uniform over $[0, c_{n-1})$ as long as at least one of the Data Providers have sampled their internal randomness honestly.

5 Security and Utility Analysis

We rigorously analyze the security and privacy properties of our proposed mechanism, and provide formal proofs of completeness, soundness, and zero-knowledge. We also evaluate the differential privacy guarantees and utility of the mechanism. The proofs for the theorems and lemmas in this section is provided in the Appendix.

5.1 Security and Privacy Analysis

Security Definitions

From a security standpoint, our scheme satisfies completeness, soundness, and zero-knowledge. We propose formal definition of these properties as below.

Definition 5.1 (Perfect Completeness)**.**
Any Prover that followed the scheme honestly outputs a proof that always passes the verification.

Definition 5.2 (Soundness)**.**
No malicious Prover can output a proof that convince the Verifier except with negligible probability.

Definition 5.3 (Zero-knowledge)**.**
Upon successful verification, the verifier learns nothing beyond the fact that the differentially private median was sampled faithfully according to the protocol.

Theorem 5.1. *Our scheme satisfies perfect completeness, soundness, and zero-knowledge.*

Proof. (*Perfect Completeness*) The completeness of our scheme follows from the design of the circuit and the completeness of underlying zk-SNARKs protocol.

(*Soundness*) Under the assumption that the underlying zk-SNARKs protocol provides soundness itself, a malicious Prover $\mathcal{A}$ could generate a false proof that could convince the Verifier by doing at least one of the following steps:

1. Tamper with the inputs from the Data Providers.
2. Tamper with the range of the query.

The Former is infeasible under the assumption of second-preimage resistance of the cryptographic hash function. The latter is impossible since the range of the query is the public input, therefore the Verifier could easily check if the Prover is cheating or not.

(*Zero-knowledge*) Under the assumption that the underlying zk-SNARKs protocol is zero knowledge, it suffices to show that each public inputs and output of the circuit does not leak any information other than DP median. It is trivial that the only source that the information of inputs can leak through is the commitments of the inputs. In the Random Oracle Model, the distribution of the commitments is uniform due to the randomness received from each Data Providers. Therefore, no information other than negligible amount can be leaked.

For a formal proof, assume that the range of the query $\mathcal{R}$, the sampled differentially private median Med and the public commitments $h_1, ..., h_m$ from the Data Providers are publicly known. Then We can construct a simulator Sim as follows:

1. Sim receives the published commitments $h_1, ..., h_m$.
2. Sim sample m pairs of (x, r) from $x \in \mathcal{R}, r \in \mathbb{F}_p$ that satisfies the following conditions.
   - r is uniform over $\mathbb{F}_p$
   - When all of the commitments $h_1, ..., h_m$ are distinct, there are no overlapping pairs $(x_i, r_i) = (x_j, r_j)$ for $i \neq j$.
   - The DP median sampled with the randomness $\rho = (\sum_{i=1}^{m} r_i \ mod p) \ mod N$ is Med.
3. Sim programs the oracle $hash$ so that $\forall i, hash(x_i, r_i) = h_i$.
4. Using the previously sampled pairs $(x_1, r_1), ..., (x_m, r_m)$, Sim generates a zkSNARK proof π and differentially private median Med from the previously sampled (x, r) pairs.
5. Sim outputs the transcript $(\pi, Med, h_1, ..., h_m, range[0], ..., range[n-1])$.

We are proving zero-knowledge in Random Oracle Model, where the simulator can produce the 'dummy' witnesses $(x_1, r_1), ..., (x_m, r_m)$ that satisfies $\forall i, hash(x_i, r_i) = h_i$ by programming the oracle. Therefore, the final transcript passes the verification. Furthermore, since we assume that the underlying zk-SNARK scheme is zero-knowledge, the transcript from the simulator is indistinguishable from an authentic proof which is generated by the Data Analyst who receives private data from the Data Providers. □

Privacy Guarantee

Our scheme should also satisfy differential privacy. In this subsection we prove that each of the method to fill in the lookup table satisfies differential privacy. T is the conceptual lookup table. We define $\mathrm{OPT}_u(\mathcal{D}) = \max_{r \in \mathcal{R}} u(\mathcal{D}, r)$ to be the maximum possible utility score from a database $\mathcal{D}$. Let $Util_{\mathcal{D}}$ be the table of the exponentiated and calibrated utility scores of each element in the range of the query under a database $\mathcal{D}$. i.e. $Util_{\mathcal{D}}$ is the table of un-normalized probability that each element is picked as a DP median. We assume $\sum_{r \in \mathcal{R}} Util_{\mathcal{D}}[r] = N$. Note that each entry in $Util_{\mathcal{D}}$ is calculated as follows.

$$Util_{\mathcal{D}}[r] = T[\mathrm{OPT}_u(\mathcal{D}) - u(\mathcal{D}, r)]$$

Theorem 5.2. ***set0*** *satisfies* (ε, δ)*-DP for* $\delta = \exp(\frac{\varepsilon}{2}) \cdot \frac{k}{N}$ *and* $k = \left\lceil \frac{1}{\exp(\varepsilon/2) - 1} \right\rceil$.

Proof. Let l be the size of the 'physical' lookup table, which is an integer such that $T[i] = 0$ for $i \geq l$ and $T[i] > 0$ for $i < l$. Let $range$ be the array of the all the elements in range of the query. $range[a, b]$ is the subarray of $range$ that includes from $range[a]$ to $range[b]$. Let ind_l and ind_r be indices of the array $range$ that satisifies the following condition.

$$\forall j \in [0, n), j \leq ind_l \text{ or } j \geq ind_r \implies u(\mathcal{D}, range[j]) \leq -l$$

Let $\mathcal{D}$ and $\mathcal{D}'$ be two neighboring databases and $\Pr[r, \mathcal{D}]$ be the probability that our mechanism run over the database $\mathcal{D}$ outputs r. Then for $r' \in range[ind_l + 2 : ind_r - 2]$,

$$\begin{aligned}
\frac{\Pr[r', \mathcal{D}]}{\Pr[r', \mathcal{D}']} &= \frac{\frac{Util_{\mathcal{D}}[r']}{\sum_{r \in \mathcal{R}} Util_{\mathcal{D}}[r]}}{\frac{Util_{\mathcal{D}'}[r']}{\sum_{r \in \mathcal{R}} Util_{\mathcal{D}'}[r]}} \\
&= \frac{Util_{\mathcal{D}}[r']}{Util_{\mathcal{D}'}[r']} \cdot \frac{\sum_{r \in \mathcal{R}} Util_{\mathcal{D}}[r]}{\sum_{r \in \mathcal{R}} Util_{\mathcal{D}'}[r]} \\
&\leq \exp(\frac{\varepsilon}{2}) \cdot \exp(\frac{\varepsilon}{2}) \\
&= \exp(\varepsilon)
\end{aligned}$$

By symmetry, we can get the following inequality.

$$\frac{\Pr[r', \mathcal{D}]}{\Pr[r', \mathcal{D}']} \geq \exp(-\varepsilon) \text{ for } r' \in range[ind_l + 2 : ind_r - 2]$$

Also, the followings hold for $r' \in range[0 : ind_l - 1] \cup range[ind_r + 1 : n - 1]$ since $Util_{\mathcal{D}}[r'] = Util_{\mathcal{D}'}[r'] = 0$.

$$\Pr[r', \mathcal{D}] = \frac{Util_{\mathcal{D}}[r']}{\sum_{r \in \mathcal{R}} Util_{\mathcal{D}}[r]} = 0 \quad \text{and} \quad \Pr[r', \mathcal{D}'] = \frac{Util_{\mathcal{D}'}[r']}{\sum_{r \in \mathcal{R}} Util_{\mathcal{D}'}[r]} = 0$$

Finally, the following holds for $r' \in range[ind_l : ind_l + 1] \cup range[ind_r : ind_r - 1]$ since $Util_{\mathcal{D}}(range[ind_l]) = Util_{\mathcal{D}}(range[ind_r]) = 0$ and the sensitivity of the utility function u is 1.

$$Util_{\mathcal{D}'}[r'] \in [\exp(-\frac{\epsilon}{2}) Util_{\mathcal{D}}[r'], \exp(\frac{\varepsilon}{2}) Util_{\mathcal{D}}[r']]$$

or

$$Util_{\mathcal{D}'}[r'] \in [Util_{\mathcal{D}}[r'] - k, Util_{\mathcal{D}}[r'] + k]$$

Combining above, we get the following result.

$$\forall r \in \mathcal{R}, \Pr[r, \mathcal{D}] \leq \exp(\varepsilon) \Pr[r, \mathcal{D}'] + \delta \text{ for } \delta \leq \exp(\frac{\varepsilon}{2}) \cdot \frac{k}{N}$$

□

Theorem 5.3. ***setk*** *satisfies* $(\varepsilon, 0)$*-DP*

Proof. All adjacent entries in $Util_{\mathcal{D}}$ differ by a ratio bounded above by $\exp(\varepsilon/2)$ and the sensitivity of the utility function is 1. Therefore, for all $r' \in \mathcal{R}$ and neighboring databases $\mathcal{D}, \mathcal{D}'$, the following holds.

$$Util_{\mathcal{D}'}[r'] \in [\exp(-\frac{\epsilon}{2}) Util_{\mathcal{D}}[r'], \exp(\frac{\varepsilon}{2}) Util_{\mathcal{D}}[r']]$$

By leveraging this property, we get

$$\begin{aligned}
\frac{\Pr[\text{range}[i], \mathcal{D}]}{\Pr[\text{range}[i], \mathcal{D}']} &= \frac{\frac{Util_{\mathcal{D}}[i]}{\sum_{j=0}^{k-1} Util_{\mathcal{D}}[j]}}{\frac{Util_{\mathcal{D}'}[i]}{\sum_{j=0}^{k-1} Util_{\mathcal{D}'}[j]}} \\
&= \frac{Util_{\mathcal{D}}[i]}{Util_{\mathcal{D}'}[i]} \cdot \frac{\sum_{j=0}^{k-1} Util_{\mathcal{D}}[j]}{\sum_{j=0}^{k-1} Util_{\mathcal{D}'}[j]} \\
&\leq \exp(\frac{\varepsilon}{2}) \cdot \exp(\frac{\varepsilon}{2}) \\
&= \exp(\varepsilon)
\end{aligned}$$

By symmetry, we can also get the following inequality.

$$\frac{\Pr[\text{range}[i], \mathcal{D}]}{\Pr[\text{range}[i], \mathcal{D}']} \geq \exp(-\varepsilon)$$

Finally, we get the following result.

$$\Pr[\text{range}[i], \mathcal{D}] \leq \exp(\varepsilon) \Pr[\text{range}[i], \mathcal{D}']$$

□

5.2 Utility Analysis

In this section, we provide the utility analysis of our scheme. We bound the probability of returning a bad estimation (i.e. an estimation that is far from the true median) and show that such probability is exponentially small. Let $\mathcal{R}_{\text{OPT}} = \{r \in \mathcal{R} : u(\mathcal{D}, r) = \text{OPT}_u(\mathcal{D})\}$.

First, we model the entries of the lookup table as a sequence $A[i]$ and the true exponentially decaying sequence as $B[i]$. $B[i] = N \cdot \exp(-\frac{\varepsilon}{2} \cdot i)$ and $A[i]$ is filled the same way the lookup table is filled. Ideally, $A[i]$ should be equal to $B[i]$, which is an exactly exponentially decaying sequence. However, since $A[i]$ is an integer approximation of exponential decay such that each adjacent elements should only differ up to the multiplicative factor of e, we should bound the error derived from this approximation.

Lemma 5.1. *Let M be a sufficiently large integer and $a > 1$ be a constant. Let Sequences $A[i]$ and $B[i]$ be defined as follows.*

$$A[i] = \begin{cases} M & \text{if } i = 0 \\ \left\lceil \frac{A[i-1]}{a} \right\rceil & \text{otherwise} \end{cases}, \qquad B[i] = N \cdot e^{-i}$$

Then, for all i, $|A[i] - B[i]| < \frac{a}{a-1}$

Proof. We can express each element in the sequence A as follows.

$$A[i] = \frac{A[i-1]}{a} + \alpha_i \ (0 \leq \alpha_i < 1)$$

Let $E[i]$ be the error of i-th term between A and B. Then,

$$\begin{aligned} E[i] &= A[i] - B[i] \\ &= \left(\frac{T[i-1]}{a} + \alpha_i\right) - \frac{B[i-1]}{a} \\ &= \frac{E[i-1]}{a} + \alpha_i \end{aligned}$$

By solving the recurrence relation, we get

$$E[i] = \sum_{j=1}^{i} \frac{\alpha_i}{e^{i-j}}$$

Since $0 \leq \alpha_i < 1$ for all i, we get

$$E[i] \leq \sum_{j=1}^{i} \frac{1}{a^{i-j}} = \frac{a}{a-1}$$

□

Now, we provide the utility proof of ***set0*** and ***setk***.

Theorem 5.4. *(Utility of **set0**) Let $x \in \mathcal{R}$ be the output of our scheme on database $\mathcal{D}$. Let the conceptual lookup table be T. Assume that l is the first index i such that $T[i] = 0$.*
If $c > -l$,

$$\Pr[u(\mathcal{D}, x) \leq c] \leq \frac{|\mathcal{R}|}{|\mathcal{R}_{OPT}|} \cdot \left(\exp\left(\frac{\varepsilon \left(c - OPT_u(\mathcal{D}) \right)}{2} \right) + \frac{e^{\varepsilon/2}}{N \cdot (e^{\varepsilon/2} - 1)} \right)$$

If $c \leq -l$,

$$\Pr[u(\mathcal{D}, x) \leq c] = 0$$

Proof. For $c > -l$ and $r \in \mathcal{R}$ such that $u(\mathcal{D}, r) \leq c$, the unnormalized probability mass that our scheme outputs r is at most $\exp\left(\frac{\varepsilon(c - \mathrm{OPT}_u(\mathcal{D}))}{2}\right) \cdot N + \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}-1}$ since the approximation of the scaled exponential values are accurate up to additive error of $\frac{e^{\varepsilon/2}}{e^{\varepsilon/2}-1}$ until $T[l-1]$ due to Lemma 5.1.

There can be at most $|\mathcal{R}|$ "bad" elements that lies beyond the utility score of c. Moreover, by the design of the `sub_min` module, the highest calibrated utility score is always 0. By the definition of $|\mathcal{R}_{\mathrm{OPT}}|$, there are exactly $|\mathcal{R}_{\mathrm{OPT}}|$ elements in $\mathcal{R}$ that possesses an unnormalized probability mass of N. Combining these facts leads to the following inequality.

$$\begin{aligned}
\Pr[u(\mathcal{D}, x) \leq c] &\leq \frac{|\mathcal{R}| \cdot \left(\exp\left(\frac{\varepsilon(c - \mathrm{OPT}_u(\mathcal{D}))}{2}\right) \cdot N + \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}-1} \right)}{|\mathcal{R}_{\mathrm{OPT}}| \cdot N} \\
&= \frac{|\mathcal{R}|}{|\mathcal{R}_{\mathrm{OPT}}|} \cdot \left(\exp\left(\frac{\varepsilon \left(c - \mathrm{OPT}_u(\mathcal{D}) \right)}{2} \right) + \frac{e^{\varepsilon/2}}{N \cdot (e^{\varepsilon/2} - 1)} \right)
\end{aligned}$$

The second inequality follows from the fact that the entries of the conceptual lookup table after $T[l]$ are set to 0. □

Theorem 5.5. *(Utility of **setk**) Let $x \in \mathcal{R}$ be the output of our scheme. Let the conceptual lookup table be T. Then,*

$$\Pr[u(\mathcal{D}, x) \leq c] \leq \frac{|\mathcal{R}|}{|\mathcal{R}_{OPT}|} \cdot \left(\exp\left(\frac{\varepsilon \left(c - OPT_u(\mathcal{D}) \right)}{2} \right) + \frac{e^{\varepsilon/2}}{N \cdot (e^{\varepsilon/2} - 1)} \right)$$

Proof. For $r \in \mathcal{R}$ such that $u(\mathcal{D}, r) \leq c$, the unnormalized probability mass that our scheme outputs r is at most $\exp\left(\frac{\varepsilon(c - \mathrm{OPT}_u(\mathcal{D}))}{2}\right) \cdot N + \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}-1}$ due to Lemma 5.1.

There can be at most $|\mathcal{R}|$ "bad" elements that lies beyond the utility score of c. Moreover, by the design of the `sub_min` module, the highest calibrated utility score is always 0. By the definition of $|\mathcal{R}_{\mathrm{OPT}}|$, there are exactly $|\mathcal{R}_{\mathrm{OPT}}|$ elements in $\mathcal{R}$ that possesses an unnormalized probability mass of N. Combining these facts leads to the following inequality.

$$\begin{aligned}
\Pr[u(\mathcal{D}, x) \leq c] &\leq \frac{|\mathcal{R}| \cdot \left(\exp\left(\frac{\varepsilon(c - \mathrm{OPT}_u(\mathcal{D}))}{2}\right) \cdot N + \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}-1} \right)}{|\mathcal{R}_{\mathrm{OPT}}| \cdot N} \\
&= \frac{|\mathcal{R}|}{|\mathcal{R}_{\mathrm{OPT}}|} \cdot \left(\exp\left(\frac{\varepsilon \left(c - \mathrm{OPT}_u(\mathcal{D}) \right)}{2} \right) + \frac{e^{\varepsilon/2}}{N \cdot (e^{\varepsilon/2} - 1)} \right)
\end{aligned}$$

□

6 Experiments

All experiments were executed on a workstation equipped with an Intel Core i7-14700F CPU (20 cores / 28 threads) and 32 GB DDR4 RAM. Workloads ran inside Windows Subsystem for Linux 2 (WSL 2) on Windows 11 Pro 22H2, with Ubuntu 24.04.1 LTS. The WSL was constrained to 24 logical processors and 30 GB RAM, with 64 GB swap space. All Circom compilations (–r1cs, –wasm, –sym) and Groth16 proof routines used Circom 2.2.2 and SnarkJS v0.7.5 under Node.js v22.15.0.

We used Groth16 as the underlying zk-SNARK protocol for the key generation, witness generation, proof generation and verification procedure. Poseidon hash function is used for the commitment that binds the inputs of the Data Providers. Our lookup table consists of 128 regular entries and the rest of the entries are set to 0 or $k = \left\lceil \frac{1}{\exp(\varepsilon/2)-1} \right\rceil$ depending on the method ***set0*** and ***setk***.

Table 1–4 and Figure 8, 9–11 report the total runtime of our scheme, (Table 2–4 and Figure 9–11 are provided in the Appendix) configured with an exponentiation lookup table to satisfy $(\varepsilon = 0.5, \delta = 0), (\varepsilon = 1, \delta = 0), (\varepsilon = 0.5, \delta = e^{-31.5})$, and $(\varepsilon = 0.5, \delta = e^{-6})$-DP. The measured times include witness generation, proof generation, and verification for input sizes ranging from 1,000 to 7,000. The domain of the median query were fixed to $[0, n = 100)$ in all test cases. Each benchmark was executed 50 times, and the mean values are presented.

Table 1: Experimental Result ($\varepsilon = 0.5, \delta = 0$)

Size of Input (m)	Witness Gen. Time	Proof Gen. Time	Verification Time
1000	0.694 s	17.984 s	16.613 ms
2000	1.024 s	29.489 s	29.196 ms
3000	1.446 s	76.886 s	87.684 ms
4000	2.307 s	117.781 s	166.876 ms
5000	2.498 s	208.001 s	238.317 ms
6000	3.149 s	217.078 s	273.572 ms
7000	3.813 s	259.031 s	277.333 ms

Table 2: Experimental Result ($\varepsilon = 1, \delta = 0$)

Size of Input (m)	Witness Gen. Time	Proof Gen. Time	Verification Time
1000	0.541 s	14.246 s	19.400 ms
2000	1.123 s	27.566 s	26.598 ms
3000	1.708 s	77.733 s	104.073 ms
4000	2.026 s	121.496 s	146.208 ms
5000	2.630 s	206.174 s	272.224 ms
6000	3.179 s	223.024 s	262.876 ms
7000	4.107 s	263.898 s	389.152 ms

Table 3: Experimental Result ($\varepsilon = 0.5, \delta = 1$)

Size of Input (m)	Witness Gen. Time	Proof Gen. Time	Verification Time
1000	0.704 s	17.928 s	17.678 ms
2000	1.030 s	29.631 s	27.286 ms
3000	1.460 s	77.753 s	100.707 ms
4000	2.272 s	118.993 s	142.104 ms
5000	2.441 s	206.990 s	240.001 ms
6000	3.098 s	219.309 s	264.621 ms
7000	3.695 s	258.434 s	280.222 ms

Table 4: Experimental Result ($\varepsilon = 1, \delta = 1$)

Size of Input (m)	Witness Gen. Time	Proof Gen. Time	Verification Time
1000	0.547 s	14.142 s	19.336 ms
2000	1.064 s	27.570 s	26.258 ms
3000	1.722 s	78.757 s	92.317 ms
4000	1.988 s	122.262 s	151.983 ms
5000	2.572 s	207.425 s	272.152 ms
6000	3.294 s	220.124 s	240.677 ms
7000	4.062 s	262.176 s	339.360 ms

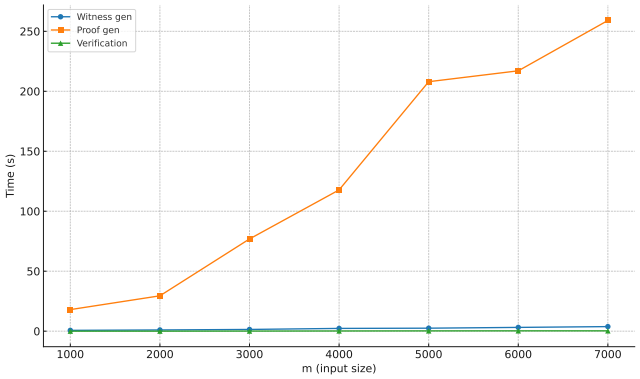

Figure 8: Experimental Result $\varepsilon = 0.5, \delta = 0$

7 Conclusion

We propose the first verifiable exponential mechanism for median estimation using zk-SNARKs. Our construction enables inverse CDF sampling via arithmetic circuits, using precomputed lookup tables to approximate exponential values efficiently. We introduce two table-filling strategies, *set0* and *setk*, which determine the privacy guarantees (pure vs. approximate DP). Our design ensures strong utility and fast proof verification. While tailored for median estimation—due to the discrete rank-based utility function—extending the approach to general real-valued utility functions remains open. Reducing the circuit complexity, currently $O(mn)$, is another key direction for future work.

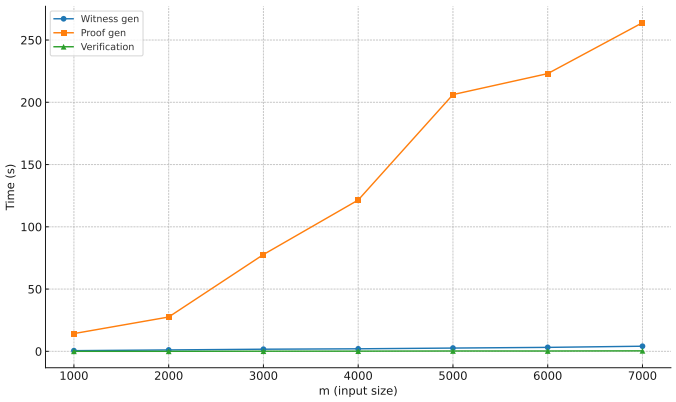

Figure 9: Experimental Result $\varepsilon = 1, \delta = 0$

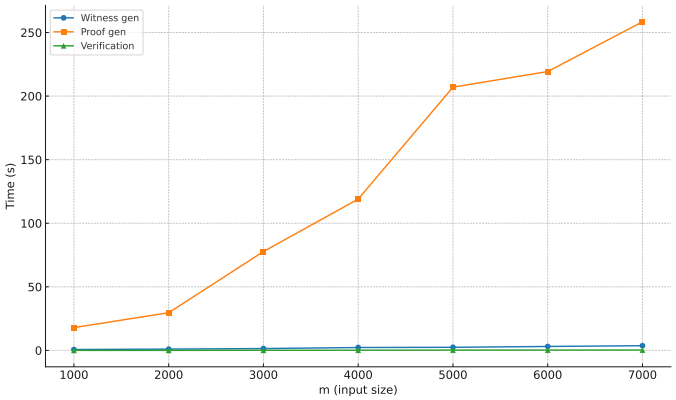

Figure 10: Experimental Result $\varepsilon = 0.5, \delta = e^{-31.5}$

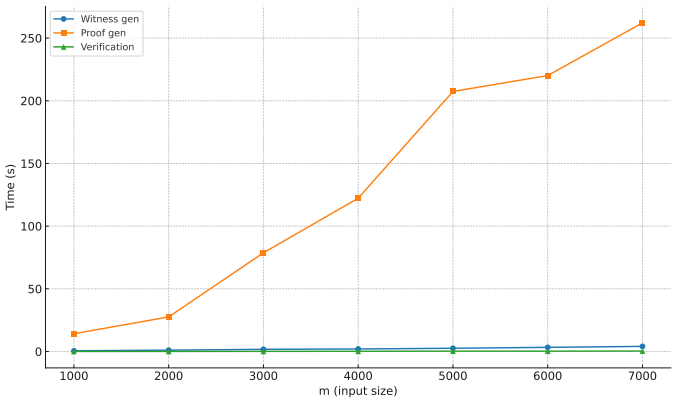

Figure 11: Experimental Result $\varepsilon = 1, \delta = e^{-63}$

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16*, pages 308–318. Association for

Computing Machinery, 2016. ISBN 978-1-4503-4139-4. doi: 10.1145/2976749.2978318.

Andris Ambainis, Markus Jakobsson, and Helger Lipmaa. Cryptographic randomized response techniques. *IACR Cryptol. ePrint Arch.*, 2003:27, 2003. URL `http://eprint.iacr.org/2003/027`.

Ari Biswas and Graham Cormode. Interactive proofs for differentially private counting. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, page 1919–1933, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400700507. doi: 10.1145/3576915.3616681. URL `https://doi.org/10.1145/3576915.3616681`.

Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS 2012)*, pages 326–349. ACM, 2012.

Tariq Bontekoe, Hassan Jameel Asghar, and Fatih Turkmen. Efficient verifiable differential privacy with input authenticity in the local and shuffle model. *CoRR*, abs/2406.18940, 2024. doi: 10.48550/ARXIV.2406.18940. URL `https://doi.org/10.48550/arXiv.2406.18940`.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 486–503. Springer Berlin Heidelberg, 2006a.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography (TCC 2006)*, pages 265–284. Springer, 2006b.

A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 211–222. ACM, 2003.

Gonzalo Munilla Garrido, Johannes Sedlmeir, and Matthias Babel. Towards verifiable differentially-private polling. In *ARES 2022: The 17th International Conference on Availability, Reliability and Security, Vienna,Austria, August 23 - 26, 2022*, pages 6:1–6:11. ACM, 2022. doi: 10.1145/3538969.3538992. URL `https://doi.org/10.1145/3538969.3538992`.

Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing (STOC 1985)*, pages 291–299. ACM, 1985.

Anupam Gupta, Katrina Ligett, Frank McSherry, Aaron Roth, and Kunal Talwar. Differentially private combinatorial optimization. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1106–1125. SIAM, 2010.

S.P. Kasiviswanathan, H.K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011. doi: 10.1137/100806601.

Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. Preventing manipulation attack in local differential privacy using verifiable randomization mechanism. In Ken Barker and Kambiz

Ghazinour, editors, *Data and Applications Security and Privacy XXXV - 35th Annual IFIP WG 11.3 Conference, DBSec 2021, Calgary, Canada, July 19-20, 2021, Proceedings*, volume 12840 of *Lecture Notes in Computer Science*, pages 43–60. Springer, 2021.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007)*, pages 94–103. IEEE, 2007.

Arjun Narayan, Ariel Feldman, Antonis Papadimitriou, and Andreas Haeberlen. Verifiable differential privacy. In Laurent Réveillère, Tim Harris, and Maurice Herlihy, editors, *Proceedings of the Tenth European Conference on Computer Systems, EuroSys 2015, Bordeaux, France, April 21-24, 2015*, pages 28:1–28:14. ACM, 2015. doi: 10.1145/2741948.2741978. URL `https://doi.org/10.1145/2741948.2741978`.

Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125, 2008. URL `https://api.semanticscholar.org/CorpusID:8843913`.

Giuseppe Vietri, Borja Balle, Akshay Krishnamurthy, and Zhiwei Steven Wu. Private reinforcement learning with pac and regret guarantees. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 9754–9764, 2020.

Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965. ISSN 01621459, 1537274X. URL `http://www.jstor.org/stable/2283137`.

Jianqi Wei, Yuling Chen, Xiuzhang Yang, Yun Luo, and Xintao Pei. A verifiable scheme for differential privacy based on zero-knowledge proofs. *J. King Saud Univ. Comput. Inf. Sci.*, 37(3), 2025. doi: 10.1007/S44443-025-00028-Z. URL `https://doi.org/10.1007/s44443-025-00028-z`.

Hui Zang and Jean Bolot. Anonymization of location data does not work: a large-scale measurement study. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking (MobiCom '11)*, pages 145–156, New York, NY, USA, 2011. ACM.