Summary Reports Optimization in the Privacy Sandbox Attribution Reporting API

Hidayet Aksu
Google
aksu@google.com

Badih Ghazi
Google
badihghazi@gmail.com

Pritish Kamath
Google
pritishk@google.com

Ravi Kumar
Google
ravi.k53@gmail.com

Pasin Manurangsi
Google
pasin@google.com

Adam Sealfon
Google
adamsealfon@google.com

Avinash V Varadarajan
Google
avaradar@google.com

ABSTRACT

The Privacy Sandbox Attribution Reporting API has been recently deployed by Google Chrome to support the basic advertising functionality of attribution reporting (aka conversion measurement) after deprecation of third-party cookies. The API implements a collection of privacy-enhancing guardrails including contribution bounding and noise injection. It also offers flexibility for the analyst to allocate the contribution budget.

In this work, we present algorithms for optimizing the allocation of the contribution budget for summary reports from the Attribution Reporting API. We develop a synthetic data model that we find to accurately capture real-world conversion data, and extensively evaluate our method on two real-world datasets and two synthetic datasets. We show that optimizing the parameters that can be set by the analyst can significantly improve the utility achieved by querying the API while satisfying the same privacy bounds.

KEYWORDS

Attribution Reporting, Conversion Measurement, Contribution Bounding, Contribution Budgeting, Discrete Laplace Mechanism, Differential Privacy, Bias-Variance Trade-off

1 INTRODUCTION

In recent years, growing concerns around user privacy have led to new efforts by web browsers and mobile platforms to limit pervasive tracking of users by websites, apps, and ad technology (aka, *ad-tech*) providers. In particular, this led to the decision by several browsers and platforms, including Safari [25], Mozilla Firefox [27], and Google Chrome [22], to deprecate third-party cookies. However, third-party cookies had been widely used to support some of the most critical functionalities powering digital advertising, notably ad conversion measurement (aka attribution reporting), where an ad-tech seeks to determine the volume of conversions attributed to ads shown on different publishers and as part of different campaigns. Example conversion measurement queries include the number of conversions attributed to ad impressions shown on a given publisher, or the total conversion value for sales occurring during a weekend and attributed to a particular ad campaign. This made several platforms and browsers provide privacy-preserving APIs that can support ad conversion measurement functionalities after the deprecation of third-party cookies, including Private Click Measurement (PCM) on Safari [26], SKAdNetwork on iOS [8], the Interoperable Private Attribution (IPA) developed by Mozilla and Meta [23], Masked LARK from Microsoft [6, 21], and Privacy Sandbox Attribution Reporting API (ARA) from Google [2, 20].

The Attribution Reporting API, available on both the Chrome browser [20] and the Android operating system [2], offers *summary reports* that could be used to estimate counts and values of conversions attributed to ad campaigns (and broken down by other impression and conversion features). The privacy guardrails in the API [3] include contribution bounding as well as discrete Laplace noise injection, which can be used to provide a differential privacy (DP) [13] guarantee on the output summary reports. More precisely, for each impression, the API enforces a fixed bound on the contributions of all conversions attributed to it. Moreover, each of these contributions is required to be discrete. The contributions from different impressions (from possibly many users) are aggregated. Discrete Laplace noise is then added to the vector of contributions, and the result is the summary report.

While ARA provides a formal DP guarantee, the noise addition and contribution bounding procedure represents a paradigm shift in conversion measurement compared to third-party cookies; straightforward use of ARA might result in large amount of noise. Concerns have been raised that the degradation in accuracy after deprecation of third-party cookies can in turn impact the downstream business decisions made based on the measurements [15, 18].

This brings us to the main question of the paper: *How can an ad-tech obtain desired measurements via ARA that are as accurate as possible?* The flexibility of ARA allows the ad-tech to choose their own encoding of the attributed information. (See Section 3.1 for formal descriptions.) By adjusting such an encoding, the ad-tech can (implicitly) decide on several parameters, such as how the contribution budget is allocated across different conversions

Proceedings on Privacy Enhancing Technologies 2024(4), 605–621
© 2024 Copyright held by the owner/author(s).
https://doi.org/10.56553/popets-2024-0132

that are attributed to the same impression. These choices by the ad-tech can significantly impact the utility of the summary reports for any fixed privacy bar. The focus of this work is to optimize these summary reports to maximize utility for a given level of (differential) privacy.

1.1 Summary of Contributions

We make the following contributions to the problem of optimizing summary reports from the Attribution Reporting API (ARA):

- ▹ We formally define the problem of optimizing the utility of conversion aggregates based on summary reports from the ARA. In the process, we flesh out subtle but important details regarding how the constraints in the API (e.g., contribution bounding, discretization requirements, encoding of multiple conversions attributed to the same impression, noise injection) can shape the optimization problem.
- ▹ Given access to historical data that has not been contribution-bounded and is noise-free, we provide an optimization problem that yields the optimal choice of parameters to use the ARA on future data. Such historical data could be available to ad-techs who have thus far relied on third-party cookies for ad conversion measurement (prior to the deprecation of third-party cookies).
- ▹ We evaluate our algorithm on real-world conversion data, demonstrating that it significantly improves utility compared to baseline non-optimized summary reports. We also evaluate our algorithm on synthetic datasets (for attributed conversion counts and values), which are sampled using generative models that we fit to real conversion data. These data generation models might be of independent interest for future research on (privacy-preserving) ad conversion measurement. [1]
- ▹ While our algorithm can be used to optimize for a variety of error measures, we discuss several qualitative advantages of using a thresholded version of the root mean square relative error in ad conversion measurement.
- ▹ As our algorithm uses past (historical) data in order to set the parameters used to measure conversions on future data, we complement our empirical findings by proving generalization bounds showing that parameters optimized using historical bounds yield good results on similar future data.

We remark that, due to the rather specific nature of the Attribution Reporting API (and, to a lesser extent, DP conversion measurements), we are not aware of any previous work that studies the same setting as ours. Nonetheless, we discuss some related work in Section 7.

1.2 Overview of the Rest of the Paper

In Section 2, we provide some basic definitions related to ad conversion measurement, and to DP. In Section 3, we provide further background on the problem setup; ARA is formalized in more detail in Section 3.1, the problem of estimating conversion aggregates (the focus of our work) is defined in Section 3.2, and we briefly discuss the error metric in Section 3.3. In Section 4, we present our optimization algorithm for contribution budgeting. We describe our experimental evaluation and findings in Section 5. In Section 6, we prove generalization bounds that explain why our algorithm does not overfit to the historical data. We discuss some related work in Section 7. We discuss limitations and interesting directions for future research in Section 8, and provide concluding remarks in Section 9.

[1] Code from our evaluation can be found at https://github.com/google-research/google-research/tree/master/ara_optimization

2 PRELIMINARIES

We now define the main terminology used in this paper. Let $\mathcal{X}$ be the set of *impression features*, including a unique impression identifier for each impression, let $\mathcal{Y}$ be the set of *conversion features*, and let $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. We denote a dataset as $D = (z_1, \ldots, z_n) \in \mathcal{Z}^*$ where each *record* z_i is of the form (x_i, y_i), consisting of impression features $x_i \in \mathcal{X}$ and conversion features $y_i \in \mathcal{Y}$. The impression features are assumed to be *known* to the ad-tech, and can correspond to an a priori unbounded number of conversions, which are assumed to be *unknown* to the ad-tech.[2] We say that two datasets D, D' are *adjacent*, denoted as $D \sim D'$ if we can get one dataset from the other by removing all records corresponding to a single impression.

We use the following running example to illustrate our notation. Imagine a fictional gift shop called *Du & Penc* that uses digital advertising to reach its customers. Their holiday sales are captured in the dataset in Table 1, where each record contains impression features of (i) an impression ID, (ii) the campaign, and (iii) the city in which the ad was shown, as well as the conversion features of the (i) number of items bought and (ii) total dollar value of items bought as part of the conversion. *Du & Penc*, as an advertiser, want to know if their ads are working, so they track conversions to see how many people click on their ads and then take the desired action, such as buying a product. Therefore, they track conversions and run queries like 'total sales per advertising campaign' to ensure their ads are effective and generating revenue. This helps them identify which ads are performing well and which ones need improvement.

2.1 Differential Privacy

Definition 2.1 (DP [13]). For $\varepsilon \geq 0$, a randomized algorithm $\mathcal{A}$ is *ε-DP* if for all adjacent datasets $D \sim D'$, and for every possible output o, it holds that $\Pr[\mathcal{A}(D) = o] \leq e^{\varepsilon} \cdot \Pr[\mathcal{A}(D') = o]$.

For an extensive overview of DP, we refer the reader to the monograph [14]. A commonly used method in DP is the discrete Laplace mechanism. To define it, we recall the notion of ℓ_1-sensitivity, where for any vector $v \in \mathbb{R}^d$, we denote its *ℓ_1-norm* as $\|v\|_1 := \sum_{i=1}^{d} |v_i|$.

Definition 2.2 (ℓ_1-sensitivity). Let $\mathcal{Z}$ be any set, and $f : \mathcal{Z}^n \to \mathbb{R}^d$ be a d-dimensional function. Its *ℓ_1-sensitivity* is defined as $\Delta_1 f := \max_{D \sim D'} \|f(D) - f(D')\|_1$.

Definition 2.3 (Discrete Laplace Mechanism). The *discrete Laplace distribution* centered at 0 and with parameter $a > 0$, denoted by $\mathrm{DLap}(a)$, is the distribution whose probability mass function at integer k is $\frac{e^a - 1}{e^a + 1} \cdot e^{-a|k|}$. The *$d$-dimensional discrete Laplace mechanism* with parameter a applied to a function $f : \mathcal{Z}^n \to \mathbb{Z}^d$, on input a dataset $D \in \mathcal{Z}^n$, returns $f(D) + \xi$ where ξ is a d-dimensional noise random variable whose coordinates are sampled i.i.d. from $\mathrm{DLap}(a)$ (abbreviated as $\xi \sim \mathrm{DLap}(a)^{\otimes d}$).

[2] Note that for the historical data both impression and conversion features are assumed to be known.

	Impression features x			**Conversion features y**	
	Impression ID	Campaign	City	#items	value ($)
$z_1 \longrightarrow$	`123`	`Thanksgiving`	New York	3	21
$z_2 \longrightarrow$	`123`	`Thanksgiving`	New York	1	5
$z_3 \longrightarrow$	`456`	`Thanksgiving`	Boston	1	99
$z_4 \longrightarrow$	`123`	`Thanksgiving`	New York	2	23
$z_5 \longrightarrow$	`101`	`Christmas`	Boston	2	50
$z_6 \longrightarrow$	`789`	`Christmas`	New York	3	15
$z_7 \longrightarrow$	`101`	`Christmas`	Boston	1	5
...	...	...	...	...	...

Table 1: Running example showing the impression and conversion logs for *Du & Penc*, an online gift shop.

$x \in \mathcal{X}$	impression features
$y \in \mathcal{Y}$	conversion features
$z \in \mathcal{Z}$	all features (x, y) of an input record
$\mathcal{Z}_j$	one slice of the input domain $\mathcal{Z} = \mathcal{Z}_1 \sqcup \cdots \sqcup \mathcal{Z}_m$
ε	privacy parameter
$D \sim D'$	adjacent datasets
$V_D(q)_j$	aggregate value of query q on slice j of dataset D
Γ	ARA-defined contribution budget, set to 65,536
$\mathcal{A}, \mathcal{R}$	pre/post-processing algorithms specified by ad-tech
$\mathcal{C}, \mathcal{S}$	fixed algorithms performed by ARA
K	set of aggregation keys $\{k_1, \ldots, k_T\}$
w^z	histogram contribution $\mathcal{A}(z)$
w^z_k	aggregatable value corresponding to w^z and key k
$\mathcal{Z}^x$	records corresponding to impression x
$\mathcal{W}^x$	histogram contributions corresponding to x
$\overline{\mathcal{Z}}^x$	records corresponding to aggregatable reports for x
$\overline{\mathcal{W}}^x$	aggregatable reports for impression x
$\overline{w}^x$	sum of aggregatable reports for impression x
W	summary report produced by algorithm $\mathcal{S}$
U	final estimate by after post-processing by $\mathcal{R}$
d	number of queries, excluding the count query
m	number of slices of the data domain
C	count limit
C_ℓ	clipping threshold for query $\ell \in [d]$
α_ℓ	contribution budget fraction for $\ell \in [d]$
τ	parameter of the RMSRE_τ error metric

Table 2: Summary of notation.

Lemma 2.4. *For all $\varepsilon > 0$, the d-dimensional discrete Laplace mechanism with parameter $a \le \varepsilon/\Delta_1 f$ is ε-DP.*

Lemma 2.5. *For all $a > 0$, it holds that*

$$\mathbb{E}[\mathrm{DLap}(a)] = 0 \text{ and } \mathrm{Var}(\mathrm{DLap}(a)) = \frac{2e^a}{(e^a - 1)^2}.$$

The following is a well-known property of DP.

Lemma 2.6 (Post-processing). *If $\mathcal{A}$ is ε-DP, then for any (randomized) algorithm $\mathcal{A}'$, it holds that $\mathcal{A}'(\mathcal{A}(\cdot))$ is also ε-DP.*

3 BACKGROUND AND SETUP

3.1 ARA Summary Reports

We now describe *Summary Reports*, which is a pre-defined framework for ad measurement fixed in the ARA [24]. At a high-level, ARA allows the ad-tech to specify the encoding algorithm from each record to a vector. A contribution bounding procedure is then applied to ensure that the total contribution corresponding to each impression is bounded. Once this is done, they are sent to the aggregation service who sums these vectors and adds a discrete Laplace noise to the result. This (noisy) summary report is then returned to the ad-tech.

More formally, a mechanism M using ARA summary reports operates as follows. To begin with, the ARA has a parameter called *contribution budget*, which at this time of writing, is fixed to $\Gamma := 2^{16} = 65,536$ (see [5]). To use the API, the ad-tech needs to specify the following information beforehand:

- a set $K = \{k_1, \ldots, k_T\}$ of *aggregation keys*, and
- an encoding algorithm $\mathcal{A}$ that maps any record z to a *histogram contribution* $w^z \in \mathbb{Z}^K_{\ge 0}$, where w^z_k is called the *aggregatable value* corresponding to the *aggregation key* $k \in K$.

Having specified the algorithm $\mathcal{A}$, the summary report is generated as follows:

(1) Each record is mapped to a histogram contribution using $\mathcal{A}$. For each impression x, let $\mathcal{Z}^x := (z_{i_1}, \ldots, z_{i_c})$ be the sequence of records corresponding to the same impression x, i.e., $z_{i_j} = (x, y_{i_j})$, and let $\mathcal{W}^x$ be the sequence of corresponding histogram contributions $(w^{z_{i_1}}, \ldots, w^{z_{i_c}})$ obtained as $w^{z_{i_j}} = \mathcal{A}(z_{i_j})$.

(2) The histogram contributions for each impression x are filtered[3] by C (Algorithm 1), such that the ℓ_1-norm of the sum of the returned vectors, called *aggregatable reports*, is at most Γ. For ease of notation, we use $\overline{\mathcal{W}}^x$ to denote the sequence of aggregatable reports for impression x, and let $\overline{w}^x := \sum_{w \in \overline{\mathcal{W}}^x} w$. Similarly, we use $\overline{\mathcal{Z}}^x$ to denote the sequence of records corresponding to aggregatable reports for impression x.

(3) The aggregatable reports are passed on by C to the ARA *aggregation service* $\mathcal{S}$ that adds all the aggregatable reports and adds discrete Laplace noise, as given in Algorithm 2, and returns a *summary report* $W \in \mathbb{Z}^K$ back to the ad-tech.

[3]Algorithm 1 as written requires all the inputs to be provided at once. But in practice, it runs in an online manner, namely, the w_is arrive sequentially and the decision of whether to include w_i or not, is made without knowing the future $w_{i'}$s, and the algorithm only needs to remember a single 16-bit value b for each impression x.

Algorithm 1 ContributionBounding C.

```
Params: Contribution budget $\Gamma := 2^{16} = 65,536$.
Input: Sequence of histogram contributions $w_1, \ldots, w_c \in \mathbb{Z}^K_{\geq 0}$ corresponding to a single impression.
Output: Sequence of aggregatable reports $w_1, \ldots, w_{\hat{c}} \in \mathbb{Z}^K_{\geq 0}$ such that $\| \sum_{j=1}^{\hat{c}} w_j \|_1 \leq \Gamma$ and $\hat{c} \leq c$.

$S \leftarrow \emptyset$ and $b \leftarrow 0$
for $i \in \{1, \ldots, c\}$ do
    if $b + \|w_i\|_1 \leq \Gamma$ then
        $S \leftarrow S \cup \{i\}$ and $b \leftarrow b + \|w_i\|_1$
return $(w_i)_{i \in S}$
```

Algorithm 2 SummaryReport $\mathcal{S}$.

```
Params: ▷ Contribution budget $\Gamma := 2^{16} = 65,536$,
        ▷ Privacy parameter $\varepsilon > 0$.
Input: Aggregatable reports $w_1, \ldots, w_r \in \mathbb{Z}^K_{\geq 0}$
Output: Summary report $W \in \mathbb{Z}^K$

return $W \leftarrow \sum_j w_j + \mathrm{DLap}(\varepsilon/\Gamma)^{\otimes K}$
```

(4) Finally, the ad-tech can post-process the summary report W using any algorithm $\mathcal{R}$ to obtain the final estimate $U \in \mathbb{R}^{d \times m}$.

Note that algorithm $\mathcal{A}$ and C are executed on the browser/device on which the impression occurred, and only the aggregatable reports are passed on to $\mathcal{S}$ to produce the summary report, which is executed in a trusted execution environment [1], and $\mathcal{R}$ is run locally by the ad-tech as illustrated in Figure 1.

An important point to note is that algorithms C and $\mathcal{S}$ are fixed in ARA, and the only parts that the ad-tech can control are algorithms $\mathcal{A}$ and $\mathcal{R}$. The design of ARA summary reports ensures that, no matter what algorithms $\mathcal{A}$ and $\mathcal{R}$ are provided, it is guaranteed that the end-to-end algorithm generating the final result received by the ad-tech is ε-DP. We include a short proof of this statement.

Theorem 3.1. *For all $\mathcal{A}$ and $\mathcal{R}$, the end-to-end algorithm combining Items 1 to 4 above satisfies ε-DP.*

Proof. We show that for any per-record input preprocessing $\mathcal{A}$ performed in step 1, steps 2–3 are an instantiation of the Discrete Laplace mechanism and thus satisfy DP by Lemma 2.4. The conclusion then follows from the robustness of DP to arbitrary postprocessing (Lemma 2.6).

The condition checked in the loop of Algorithm C (step 2) ensures that regardless of how algorithm $\mathcal{A}$ transforms each individual record, the total contribution $\overline{w}^x$ of each impression x satisfies $\|\overline{w}^x\|_1 \leq \Gamma$. Algorithm $\mathcal{S}$ then adds Discrete Laplace noise scaled to this maximum sensitivity Γ, so the algorithm combining steps 1–3 instantiates the Discrete Laplace mechanism and is guaranteed to be ε-DP under adding or removing all conversions attributed to a single impression,[4] by Lemma 2.4. Finally, in Item 4, algorithm $\mathcal{R}$ post-processes the summary report W to obtain the final estimate $U \in \mathbb{R}^{d \times m}$; by Lemma 2.6, this remains ε-DP. □

Recall that the goal of our paper is to design a mechanism in this framework that minimizes a desired error metric. In order to design a mechanism, we need to specify: (i) the set K of aggregation keys, (ii) algorithm $\mathcal{A}$ mapping records to histogram contributions w^z, and (iii) the reconstruction algorithm $\mathcal{R}$ for recovering estimates of interest. As we will demonstrate through the rest of this paper, careful selection of these can lead to significant utility improvement.

3.2 Estimating Conversion Aggregates

As mentioned earlier, we are interested in computing aggregate statistics on attributed conversion where they can be "sliced" based on certain attributes. This problem is formalized below.

Consider a fixed partition of $\mathcal{Z}$ given as $\mathcal{Z}_1 \sqcup \cdots \sqcup \mathcal{Z}_m$, where the $\mathcal{Z}_j$'s are pairwise disjoint (where $\sqcup$ denotes a disjoint union). This partition naturally induces a partition of D given as $D_1 \sqcup \cdots \sqcup D_m$ where $D_j = D \cap \mathcal{Z}_j$; we refer to each D_j as a *slice* of D.[5] We use $X := \{x : \exists y \in \mathcal{Y} \text{ s.t. } (x, y) \in D\}$ and $X_j := \{x : \exists y \in \mathcal{Y} \text{ s.t. } (x, y) \in D_j\}$, whenever D is clear from context.

A *statistical query*, denoted *query* in the rest of the article, is an aggregatable value defined by a function $q : \mathcal{Z} \to \mathbb{R}$.[6] The *aggregate value*[7] associated with a query q on dataset D is $V_D(q) \in \mathbb{R}^m$, given as $V_D(q)_j := \sum_{z \in D_j} q(z)$, namely one aggregate for each slice of D. Given queries $q_1, \ldots, q_d$ of interest, the goal is to construct an ε-DP mechanism that estimates the corresponding aggregate values $V_D(q_1), \ldots, V_D(q_d)$ as "accurately" as possible.

In the *Du & Penc* example, one could consider a partition of the records, e.g., by *Campaign* or by *City* or by the pair *(Campaign, City)*. We could consider the following two queries: $q_1(z)$, which returns #items and $q_2(z)$, which returns the dollar value in any record z. In addition, we always consider another query of interest, given as $q_0(z) = 1$ for all $z \in \mathcal{Z}$; that is, $V_D(q_0)_j$ is precisely the number of records in the jth slice.

3.3 Error Metrics for Experiment Evaluation

Since the values estimated using Summary Reports API are noisy, API users should evaluate the impact of noise in reports carefully to ensure that they are useful. For example, when *Du & Penc* executes queries such as 'total sales per advertising campaign' to determine the effectiveness of individual campaigns, the presence of added noise may introduce slight or significant variations in the results. Consequently, the utility of summary reports is called into question when noise is present. A variety of utility metrics can be used to evaluate the impact of noise. We provide a list of all metrics that we have considered in Table 5 (Appendix A). There are several desirable criteria for metrics. As listed in Table 6 (Appendix A), we find out that only RMSRE_τ metric (defined below) satisfies all desired properties. Therefore, we focus on this metric for our main

[4]Note that this is a simpler setting compared to adding or removing an impression, which may create a "cascading effect" that leads to modifications in the conversions attributed to multiple impressions at once. However, since the current version of the ARA does not touch upon this issue, we will do the same in this paper.

[5]Slices may be any partition of the data domain. For example, slices might be specified by a GROUP BY statement in SQL.

[6]We emphasize that a query here is a function of a single data record, not a function of the whole dataset. This usage is similar to the statistical query model in learning theory, but here we allow queries to be real-valued, not just $[0, 1]$-valued.

[7]The aggregation here is specifically summation; we use the term aggregate for consistency with the API documentation [24]

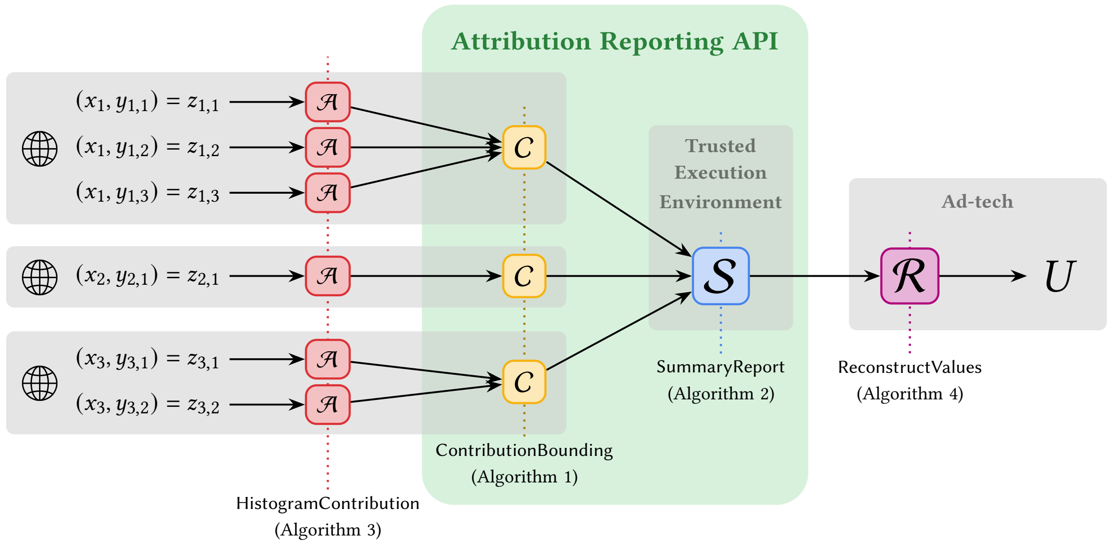

Attribution Reporting API
$(x_1, y_{1,1}) = z_{1,1}$
$(x_1, y_{1,2}) = z_{1,2}$
$(x_1, y_{1,3}) = z_{1,3}$
$(x_2, y_{2,1}) = z_{2,1}$
$(x_3, y_{3,1}) = z_{3,1}$
$(x_3, y_{3,2}) = z_{3,2}$
$\mathcal{A}$
C
Trusted Execution Environment
$\mathcal{S}$
Ad-tech
$\mathcal{R}$
U
SummaryReport (Algorithm 2)
ReconstructValues (Algorithm 4)
ContributionBounding (Algorithm 1)
HistogramContribution (Algorithm 3)

Figure 1: Illustrative usage of ARA summary reports. Algorithms C and $\mathcal{S}$ are fixed in the API. The ad-tech designs $\mathcal{A}$ and $\mathcal{R}$.

experimental evaluation. However, our approach can be applied just as well to other error metrics, as discussed in Appendix B.

Definition 3.2 ([19]). For a dataset $D \in \mathcal{Z}^*$, a query $q : \mathcal{Z} \to \mathbb{R}$, and a random vector $u \in \mathbb{R}^m$, the *root mean squared relative error* with parameter $\tau \in \mathbb{R}_{>0}$ is defined as

$$\mathrm{RMSRE}_\tau(u, q; D) \;:=\; \sqrt{\frac{1}{m} \sum_{j \in [m]} \mathbb{E}\left(\frac{u_j - V_D(q)_j}{\max\{\tau, V_D(q)_j\}}\right)^2},$$

where the expectation is over the randomness of u. Similarly, for parameters $\boldsymbol{\tau} = (\tau_0, \ldots, \tau_d)$, queries $(q_0, \ldots, q_d)$, and a randomized report $U \in \mathbb{R}^{(d+1)\times m}$, we define

$$\mathrm{RMSRE}_{\boldsymbol{\tau}}(U, (q_\ell)_{\ell=0}^d; D) \;:=\; \sqrt{\frac{1}{d+1} \sum_{\ell=0}^{d} \mathrm{RMSRE}_{\tau_\ell}(U_\ell, q_\ell; D)^2},$$

where U_ℓ is the ℓth row of U.

4 CONTRIBUTION BUDGETING ALGORITHM

In this section, we present our algorithm for contribution budgeting. Suppose we have d queries $q_1, \ldots, q_d : \mathcal{Z} \to \mathbb{R}_{\geq 0}$ for which an ad-tech desires to estimate the aggregate values. We now describe our approach for defining the aggregation keys K, the encoding algorithm $\mathcal{A}$ mapping records to histogram contributions, and the method $\mathcal{R}$ for reconstructing the values.

4.1 Aggregation Keys & Histogram Contributions

We take the set of aggregation keys to be $K = ([d] \cup \{\perp\}) \times [m]$, i.e., we define $d+1$ aggregation keys corresponding to each slice $j \in [m]$. We choose a *count limit* C, and additionally, corresponding to each query q_ℓ, we choose a *clipping threshold* C_ℓ and a *contribution budget fraction* α_ℓ, subject to $\sum_{\ell=1}^d \alpha_\ell = 1$. We discuss how to choose the parameters C and $(C_\ell, \alpha_\ell)_{\ell \in [d]}$ shortly.

For any record z, suppose $v_1, \ldots, v_d$ are given as $v_\ell = q_\ell(z)$. We first clip each v_ℓ to be at most C_ℓ, namely, let $v'_\ell = \mathrm{clip}_{C_\ell}(v_\ell) := \min(v_\ell, C_\ell)$. Next, we scale v'_ℓ to lie in $[0, 1]$, by dividing by C_ℓ, namely, let $v''_\ell = v'_\ell / C_\ell$. We will choose a histogram contribution w^z that uses exactly $\lfloor \Gamma/C \rfloor$ of the total contribution budget, so that at most C conversions can be accounted for per impression. To this end, we rescale v''_ℓ by $\lfloor \alpha_\ell \Gamma / C \rfloor$ so that the sum of contributions is at most Γ/C, and moreover, we assign the remaining contribution mass to $\perp$. Finally, we apply a randomized rounding on the real values to make them integer-valued, as required by the API.

These steps are formalized in Algorithm 3, where CRR is a randomized method that *clips* and performs *randomized rounding*, defined as follows:

$$\mathrm{CRR}(v; C, C_\ell, \alpha_\ell) \;:=\; \mathrm{RR}\left(\left\lfloor \frac{\alpha_\ell \Gamma}{C} \right\rfloor \cdot \frac{\mathrm{clip}_{C_\ell}(v)}{C_\ell}\right), \tag{1}$$

$$\text{where} \quad \mathrm{RR}(\omega) \;:=\; \begin{cases} \lceil \omega \rceil & \text{w.p. } \omega - \lfloor \omega \rfloor \\ \lfloor \omega \rfloor & \text{w.p. } 1 - \omega + \lfloor \omega \rfloor\,. \end{cases} \tag{2}$$

We use RR instead of deterministic rounding because $\mathbb{E}[\mathrm{RR}(\omega)] = \omega$, which allows recovery of unbiased estimates from summary reports.

Lemma 4.1. *For any z, the vector w^z returned by Algorithm 3 satisfies $w^z \geq \mathbf{0}$ and $\|w^z\|_1 = \lfloor \Gamma/C \rfloor$.*

Proof. It is immediate to see that $\sum_{\ell,j} w^z_{\ell,j} + \sum_j w^z_{\perp,j} = \lfloor \Gamma/C \rfloor$. Let j be such that $z \in \mathcal{Z}_j$. To show that indeed all the values are non-negative, we observe that $w^z_{\ell,j'} = 0$ for all $j' \neq j$, and $0 \leq \mathrm{CRR}(v; C, C_\ell, \alpha_\ell) \leq \lfloor \alpha_\ell \Gamma / C \rfloor$ and hence, we have

$$\textstyle\sum_{\ell=1}^d w^z_{\ell,j} \;\leq\; \sum_{\ell=1}^d \left\lfloor \frac{\alpha_\ell \Gamma}{C} \right\rfloor \;\leq\; \left\lfloor \frac{\Gamma}{C} \sum_{\ell=1}^d \alpha_\ell \right\rfloor \;=\; \left\lfloor \frac{\Gamma}{C} \right\rfloor,$$

and hence $w^z_{\perp,j} = \lfloor \Gamma/C \rfloor - \sum_\ell w^z_{\ell,j} \geq 0$. □

Thus, we have that for any impression x, at most the first C conversions result in valid aggregatable reports.

Algorithm 3 HistogramContribution $\mathcal{A}$ (ℓ_∞ version).

```
Params: ▷ Queries $q_1, \ldots, q_d : \mathcal{Z} \to \mathbb{R}_{\geq 0}$,
        ▷ Partition of $\mathcal{Z} = \mathcal{Z}_1 \sqcup \cdots \sqcup \mathcal{Z}_m$,
        ▷ Parameters $C$, $(C_\ell, \alpha_\ell)_{\ell \in [d]}$ with $\sum_{\ell=1}^{d} \alpha_\ell = 1$,
        ▷ Aggregatable Keys $K = ([d] \cup \{\perp\}) \times [m]$.
Input: Record $z \in \mathcal{Z}$
Output: Histogram contribution $w^z \in \mathbb{Z}_{\geq 0}^K$
for slice $j \in [m]$ do
    for $\ell \in [d]$ do
        $w^z_{\ell,j} \leftarrow \begin{cases} 0 & \text{if } z \notin \mathcal{Z}_j \\ \mathrm{CRR}(q_\ell(z); C, C_\ell, \alpha_\ell) & \text{if } z \in \mathcal{Z}_j \end{cases}$
    $w^z_{\perp,j} \leftarrow \begin{cases} 0 & \text{if } z \notin \mathcal{Z}_j \\ \lfloor \frac{\Gamma}{C} \rfloor - \sum_{\ell=1}^{d} w^z_{\ell,j} & \text{if } z \in \mathcal{Z}_j \end{cases}$
return $w^z$
```

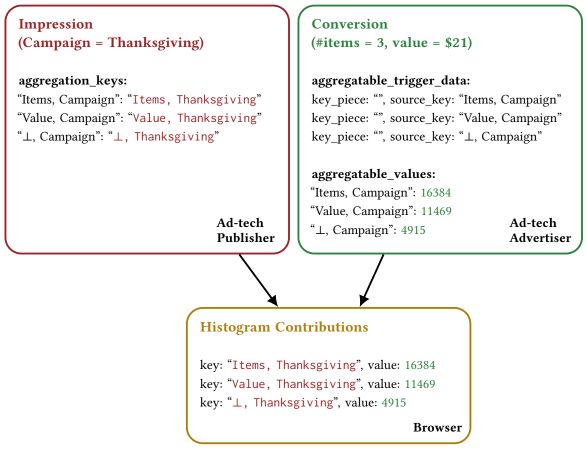

Impression
(Campaign = Thanksgiving)
aggregation_keys:
"Items, Campaign": "Items, Thanksgiving"
"Value, Campaign": "Value, Thanksgiving"
"⊥, Campaign": "⊥, Thanksgiving"
Ad-tech Publisher
Conversion
(#items = 3, value = $21)
aggregatable_trigger_data:
key_piece: "", source_key: "Items, Campaign"
key_piece: "", source_key: "Value, Campaign"
key_piece: "", source_key: "⊥, Campaign"
aggregatable_values:
"Items, Campaign": 16384
"Value, Campaign": 11469
"⊥, Campaign": 4915
Ad-tech Advertiser
Histogram Contributions
key: "Items, Thanksgiving", value: 16384
key: "Value, Thanksgiving", value: 11469
key: "⊥, Thanksgiving", value: 4915
Browser

Figure 2: Histogram contributions generation.

Figure 2 illustrates how this histogram contribution gets prepared at the browser/device, using the example of the *Du & Penc* dataset in Table 1. Consider queries $q_1(z)$ being the number of items purchased and $q_2(z)$ being the corresponding dollar value, and the slices corresponding to the "Campaign". In this case, the set of aggregatable keys K is $\{1, 2, \perp\} \times \{\texttt{Thanksgiving}, \texttt{Christmas}\}$, corresponding to all possible combinations of query and campaign, as well as the remaining contribution $\perp$ for each campaign.

Suppose we use count limit $C = 2$, clipping thresholds $C_1 = 2$, $C_2 = \$30$ and contribution budget fractions $\alpha_1 = \alpha_2 = 0.5$. Consider the record z_1 corresponding to Impression ID 123, corresponding to the Thanksgiving campaign. The conversion consists of $v_1 = 3$ items and value $v_2 = \$21$, which get clipped as $v_1' = \mathrm{clip}_{C_1}(v_1) = 2$ and $v_2' = \mathrm{clip}_{C_2}(v_2) = \$21$. These get rescaled as $v_1'' = v_1'/C_1 = 1$ and $v_2'' = v_2'/C_2 = 0.7$. Finally, v_1'' and v_2'' are further rescaled by $\lfloor \alpha_1 \Gamma / C \rfloor$ and $\lfloor \alpha_2 \Gamma / C \rfloor$, respectively, which are both equal to $\Gamma/4 = 16384$. Thus, z_1 gets mapped to the following aggregatable values $w_k^{z_1}$:

$$\begin{aligned} w_{1,\texttt{Thanksgiving}} &= \mathrm{RR}(16384 \cdot v_1'') = \mathrm{RR}(16384) = 16384 \\ w_{2,\texttt{Thanksgiving}} &= \mathrm{RR}(16384 \cdot v_2'') = \mathrm{RR}(11468.8) \overset{\text{e.g.,}}{=} 11469 \\ w_{\perp,\texttt{Thanksgiving}} &= 32768 - 16384 - 11469 = 4915 \\ w_{1,\texttt{Christmas}} &= w_{2,\texttt{Christmas}} = w_{\perp,\texttt{Christmas}} = 0 \end{aligned}$$

The histogram contribution w^{z_1} consists of these aggregatable values.

We have $\|w^{z_1}\|_1 = 32768 = \Gamma/C$; this holds for all records z, not just z_1. Since records z_1, z_2, and z_4 in Table 1 all correspond to the same impression x, the histogram contribution w^{z_4} will end up being ignored in the aggregate, since $\|w^{z_1} + w^{z_2} + w^{z_4}\| > \Gamma$. That is, $\overline{\mathcal{Z}}^x = (z_1, z_2)$, and the aggregatable reports corresponding to records z_1 and z_2 will simply be the histogram contributions w^{z_1} and w^{z_2}, while no aggregatable report will be produced for z_4.

4.2 Reconstruction of Values

Given the summary report W (from Algorithm 2), we can post-process to obtain estimates of $V_D(q_\ell)$; see Algorithm 4. We use $\overline{\mathcal{Z}}_j^x$ to denote $\overline{\mathcal{Z}}^x \cap \mathcal{Z}_j$.

Algorithm 4 ReconstructValues $\mathcal{R}$

```
Params: ▷ Queries $q_1, \ldots, q_d : \mathcal{Z} \to \mathbb{R}_{\geq 0}$,
        ▷ Partition of $\mathcal{Z} = \mathcal{Z}_1 \sqcup \cdots \sqcup \mathcal{Z}_m$,
        ▷ Parameter $C$ and scale parameters $\beta_1, \ldots, \beta_d$.
          For $\mathcal{A}$ as in Algorithm 3, $\beta_\ell = C_\ell \big/ \lfloor \frac{\alpha_\ell \Gamma}{C} \rfloor$.
Input: Summary report $W \in \mathbb{Z}^K$ for $K = [m] \times ([d] \cup \{\perp\})$
Output: $U \in \mathbb{R}^{(d+1) \times m}$ with $U_{k,j}$ corresponding to slice $j$, query $q_\ell$.
for slice $j \in [m]$ do
    for $\ell \in [d]$ do
        $U_{\ell,j} \leftarrow W_{\ell,j} \cdot \beta_\ell$
    $U_{0,j} \leftarrow (W_{\perp,j} + \sum_{\ell \in [d]} W_{\ell,j}) \big/ \lfloor \frac{\Gamma}{C} \rfloor$
return $U$
```

Theorem 4.2. *For U returned by conjunction of Algorithms 1, 2, 4 and 6 as in Figure 1, it holds for all $\ell \in [d]$, that*

$$\begin{aligned} \mathbb{E}[U_{\ell,j}] &= \textstyle\sum_{x \in X_j} \sum_{z \in \overline{\mathcal{Z}}_j^x} \mathrm{clip}_{C_\ell}(q_\ell(z)), \\ \textit{and} \quad \mathbb{E}[U_{0,j}] &= \textstyle\sum_{x \in X_j} |\overline{\mathcal{Z}}_j^x|. \end{aligned}$$

Moreover, the variances are given as,

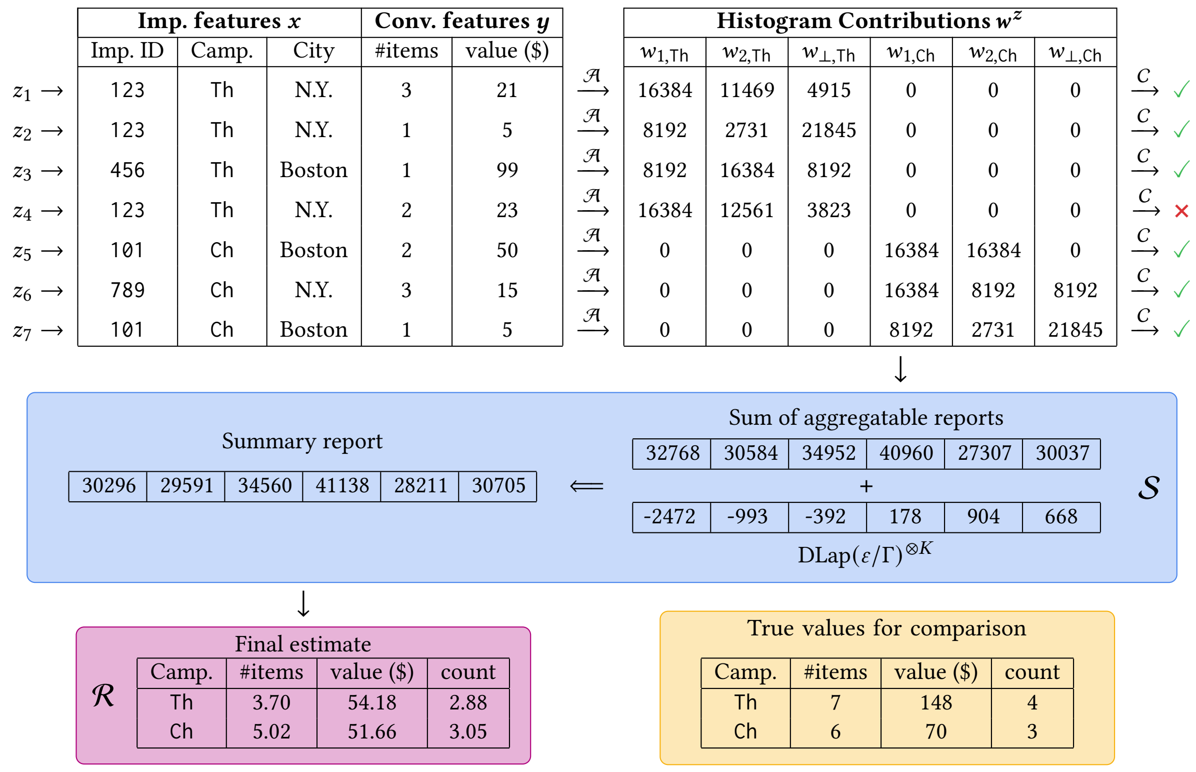
$$\begin{aligned} \mathrm{Var}(U_{\ell,j}) &\leq \left(\frac{\sum_{x \in X_j} |\overline{\mathcal{Z}}_j^x|}{4} + \mathrm{Var}(\mathrm{DLap}(\varepsilon/\Gamma)) \right) \cdot C_\ell^2 \Big/ \left\lfloor \frac{\alpha_\ell \Gamma}{C} \right\rfloor^2, \\ \textit{and} \quad \mathrm{Var}(U_{0,j}) &= \mathrm{Var}(\mathrm{DLap}(\varepsilon/\Gamma)) \cdot (d+1) \Big/ \left\lfloor \frac{\Gamma}{C} \right\rfloor^2. \end{aligned}$$

Proof. It is easy to see that

$$W_{\ell,j} = \textstyle\sum_{x \in X_j} \sum_{z \in \overline{\mathcal{Z}}_j^x} \mathrm{RR}\left(\left\lfloor \frac{\alpha_\ell \Gamma}{C} \right\rfloor \cdot \frac{\mathrm{clip}_{C_\ell}(q_\ell(z))}{C_\ell} \right) + \mathrm{DLap}\left(\frac{\varepsilon}{\Gamma}\right).$$

Using the fact that $\mathbb{E}[\mathrm{RR}(\omega)] = \omega$ and $\mathrm{Var}(\mathrm{RR}(\omega)) \leq \frac{1}{4}$, we get

$$\begin{aligned} \mathbb{E}[W_{\ell,j}] &= \textstyle\sum_{x \in X_j} \sum_{z \in \overline{\mathcal{Z}}_j^x} \left\lfloor \frac{\alpha_\ell \Gamma}{C} \right\rfloor \cdot \frac{\mathrm{clip}_{C_\ell}(q_\ell(z))}{C_\ell}, \\ \mathrm{Var}(W_{\ell,j}) &\leq \textstyle\sum_{x \in X_j} \frac{|\overline{\mathcal{Z}}_j^x|}{4} + \mathrm{Var}\left(\mathrm{DLap}\left(\frac{\varepsilon}{\Gamma}\right)\right). \end{aligned}$$



	Imp. features x			Conv. features y			Histogram Contributions w^z							
	Imp. ID	Camp.	City	#items	value ($)		$w_{1,\mathsf{Th}}$	$w_{2,\mathsf{Th}}$	$w_{\perp,\mathsf{Th}}$	$w_{1,\mathsf{Ch}}$	$w_{2,\mathsf{Ch}}$	$w_{\perp,\mathsf{Ch}}$		
$z_1 \longrightarrow$	123	Th	N.Y.	3	21	$\xrightarrow{\mathcal{A}}$	16384	11469	4915	0	0	0	$\xrightarrow{C}$	✓
$z_2 \longrightarrow$	123	Th	N.Y.	1	5	$\xrightarrow{\mathcal{A}}$	8192	2731	21845	0	0	0	$\xrightarrow{C}$	✓
$z_3 \longrightarrow$	456	Th	Boston	1	99	$\xrightarrow{\mathcal{A}}$	8192	16384	8192	0	0	0	$\xrightarrow{C}$	✓
$z_4 \longrightarrow$	123	Th	N.Y.	2	23	$\xrightarrow{\mathcal{A}}$	16384	12561	3823	0	0	0	$\xrightarrow{C}$	✗
$z_5 \longrightarrow$	101	Ch	Boston	2	50	$\xrightarrow{\mathcal{A}}$	0	0	0	16384	16384	0	$\xrightarrow{C}$	✓
$z_6 \longrightarrow$	789	Ch	N.Y.	3	15	$\xrightarrow{\mathcal{A}}$	0	0	0	16384	8192	8192	$\xrightarrow{C}$	✓
$z_7 \longrightarrow$	101	Ch	Boston	1	5	$\xrightarrow{\mathcal{A}}$	0	0	0	8192	2731	21845	$\xrightarrow{C}$	✓

$\downarrow$

Sum of aggregatable reports

32768	30584	34952	40960	27307	30037

+

-2472	-993	-392	178	904	668

$\mathrm{DLap}(\varepsilon/\Gamma)^{\otimes K}$

$\Longleftarrow$ $\mathcal{S}$

Summary report

30296	29591	34560	41138	28211	30705

$\downarrow$

$\mathcal{R}$ Final estimate

Camp.	#items	value ($)	count
Th	3.70	54.18	2.88
Ch	5.02	51.66	3.05

True values for comparison

Camp.	#items	value ($)	count
Th	7	148	4
Ch	6	70	3

Figure 3: Summary report generation on dataset in Table 1 using $C = 2$, $C_1 = 2$, $C_2 = \$30$ and $\alpha_1 = \alpha_2 = 0.5$, with slices defined for each campaign.

Similarly, we have that

$$\sum_{\ell=1}^{d} W_{\ell,j} + W_{\perp,j} \;=\; \sum_{x\in X_j} \left\lfloor \frac{\Gamma}{C} \right\rfloor \cdot |\mathcal{Z}_j^x| + \sum_{\ell=0}^{d} \mathrm{DLap}\left(\varepsilon/\Gamma\right),$$

and hence

$$\mathbb{E}\left[\sum_{\ell=1}^{d} W_{\ell,j} + W_{\perp,j}\right] \;=\; \sum_{x\in X_j} \left\lfloor \frac{\Gamma}{C} \right\rfloor \cdot |\mathcal{Z}_j^x|,$$

$$\mathrm{Var}\left(\sum_{\ell=1}^{d} W_{\ell,j} + W_{\perp,j}\right) \;=\; (d+1)\cdot \mathrm{Var}(\mathrm{DLap}\left(\varepsilon/\Gamma\right)).$$

The proof is now complete by observing that $U_{\ell,j}$ and $U_{0,j}$ are simply scaled versions of $W_{\ell,j}$ and $W_{\perp,j}+\sum_{\ell=1}^{d} W_{\ell,j}$ respectively. □

In Figure 3, we provide a complete walk-through of how each of the algorithms $\mathcal{A}$, C, $\mathcal{S}$, and $\mathcal{R}$ (in Figure 1) operate on the dataset in Table 1. We start with the input records $z_1, \ldots, z_7$ in the top left. Algorithm $\mathcal{A}$ is then applied to each record, clipping the number of items to $C_1 = 2$, rescaling by $\lfloor \alpha_1 \Gamma / C \rfloor / C_1 = 8192$ and applying randomized rounding, and similarly clipping the price to \$30, rescaling by $\lfloor \alpha_2 \Gamma / C \rfloor / C_2 = 8192$ and applying randomized rounding, as described in Section 4.1. Algorithm C then applies contribution bounding, producing aggregatable reports using the histrogram contributions for six of the records but not for z_4, since the ℓ_1 contribution budget Γ for impression 123 has already been used up by records z_1 and z_2. For each query $\{1, 2, \perp\}$ and campaign $\{\mathsf{Th}, \mathsf{Ch}\}$, Algorithm $\mathcal{S}$ then sums the aggregatable reports for that aggregation key and adds Discrete Laplace noise, producing the summary report in the middle left of the figure. Finally, algorithm $\mathcal{R}$ takes this summary report and rescales the values to obtain the final estimate in the lower left. This entails multiplying the first and fourth values in the summary report by $\beta_1 = C_1 / \left\lfloor \frac{\alpha_1 \Gamma}{C} \right\rfloor = 1/8192$ to get the estimated number of items for the two campaigns, and the second and fifth values by $\beta_2 = C_2 / \left\lfloor \frac{\alpha_2 \Gamma}{C} \right\rfloor = 15/8192$ to get the estimated values for the two campaigns. The estimated counts for each campaign are obtained by adding all three summary report values corresponding to the campaign (the first three values for campaign Th, the last three for campaign Ch) and scaling by $1/\lfloor \Gamma/C \rfloor = 1/32768$.

We can compare the estimates obtained to the true total values for each campaign in the lower right of the figure. The estimated values in this walk-through are far from the true values for two reasons: the dataset is too small to provide good accuracy, and the value of $C_2 = \$30$ means that large values such as \$99 and \$50 get clipped to \$30.

4.3 Optimization of Parameters

Having understood the variance in the estimates, we turn to the question of understanding the optimal choice of parameters C, $(C_\ell, \alpha_\ell)_{\ell\in[d]}$, with the goal of minimizing $\mathrm{RMSRE}_\tau(U, (q_\ell)_{\ell=0}^{d}; D)$. In this section, we optimize the choice of parameters, *using knowledge of the dataset D*. This is admittedly circular, as we are using an ε-DP mechanism to learn information about a dataset *we do not know*. However, the eventual goal is that we will optimize the parameters using a historical dataset D', which is "similarly behaved"

Algorithm 5 ParameterOptimization

```
Params: ▷ Queries $q_1, \dots, q_d : \mathcal{Z} \to \mathbb{R}_{\geq 0}$
        ▷ Partition of $\mathcal{Z} = \mathcal{Z}_1 \sqcup \cdots \sqcup \mathcal{Z}_m$.
Input: Dataset $D \in \mathcal{Z}^*$
Output: Parameters $C, (C_\ell, \alpha_\ell)_{\ell \in [d]}$ with $\sum_{\ell=1}^{d} \alpha_\ell = 1$.
$R_{\text{current}} \leftarrow \infty$
$C_{\max} \leftarrow \max_{x \in X} |\mathcal{Z}^x|$     (max conversions for an impression)
for $\widehat{C} = 1, 2, \dots, C_{\max}$ do
    $(\widehat{C}_\ell, \widehat{\alpha}_\ell)_{\ell \in [d]} \leftarrow \operatorname{argmin}_{(C_\ell, \alpha_\ell)_{\ell \in [d]}} R(\widehat{C}, (C_\ell, \alpha_\ell)_{\ell \in [d]})$
    if $R(\widehat{C}, (\widehat{C}_\ell, \widehat{\alpha}_\ell)_{\ell \in [d]}) < R_{\text{current}}$ then
        $(C^*_\ell, \alpha^*_\ell)_{\ell \in [d]} \leftarrow (\widehat{C}_\ell, \widehat{\alpha}_\ell)_{\ell \in [d]}$
        $C^* \leftarrow C$
        $R_{\text{current}} \leftarrow R(\widehat{C}, (\widehat{C}_\ell, \widehat{\alpha}_\ell)_{\ell \in [d]})$
return $C^*, (C^*_\ell, \alpha^*_\ell)_{\ell \in [d]}$
```

to D and as we show in Section 6, this is a reasonable choice as long as the two datasets are drawn from the same distribution.

We stress that, since our optimization procedure is performed using the dataset D' that is assumed to be public, this has no effect on the DP guarantee of the sensitive dataset D.

We focus on minimizing $\mathsf{RMSRE}_{\boldsymbol{\tau}}(U, (q_\ell)_{\ell=0}^{d}; D)^2$ as a function of $C, (C_\ell, \alpha_\ell)_{\ell \in [d]}$. The optimization problem we solve is as follows.

Problem 1. Given a fixed dataset D, a partition of the data universe into m slices $\mathcal{Z}_1, \cdots, \mathcal{Z}_m$, and $d+1$ queries $q_0, \dots, q_d$, let $C, (C_\ell, \alpha_\ell)_{\ell \in [d]}$ be decision variables. For each $j \in [m]$, $q \in \{q_0, \dots, q_d\}$, let $V_D(q)_j := \sum_{z \in D \cap \mathcal{Z}_j} q(z)$ and $U = \mathcal{R} \circ \mathcal{S} \circ C^k \circ \mathcal{A}^{|D|}$, where k denotes the number of impressions in D and the algorithms are called on dataset D with the parameters specified by the other variables. Minimize the objective function

$$R(C, (C_\ell, \alpha_\ell)_{\ell \in [d]}) := \sqrt{\frac{1}{(d+1)m} \sum_{\ell=0}^{d} \sum_{j \in [m]} \mathbb{E}\left(\frac{U_{\ell,j} - V_D(q_\ell)_j}{\max\{\tau_\ell, V_D(q_\ell)_j\}}\right)^2},$$

subject to the constraints $C > 0$, $C_\ell > 0$, and $\alpha_\ell > 0$ for all $\ell \in [d]$, and $\sum_{\ell=1}^{d} \alpha_\ell = 1$.

As discussed in Appendix B, we can also apply our approach with other error metrics by replacing RMSRE_τ in the objective function with the desired error metric.

Denoting $\pi_{\ell,j} := 1/\max\{\tau_\ell, V_D(q_\ell)_j\}^2$, we can rewrite the objective as

$$R(C, (C_\ell, \alpha_\ell)_{\ell \in [d]})^2 := \frac{1}{(d+1)m} \sum_{\ell=0}^{d} \sum_{j \in [m]} \pi_{\ell,j} \cdot \mathbb{E}(U_{\ell,j} - V_D(q_\ell)_j)^2. \tag{3}$$

That is, R is a linear combination of the following terms for $\ell \in \{0, \dots, d\}$ and $j \in [m]$, for which we can use the bias-variance decomposition, namely

$$\mathbb{E}(U_{\ell,j} - V_D(q_\ell)_j)^2 = (V_D(q_\ell)_j - \mathbb{E}\, U_{\ell,j})^2 + \mathsf{Var}(U_{\ell,j}).$$

Thus, we can choose the optimal parameters using a procedure as described in Algorithm 5. Namely, we enumerate over various values of C, and fixing $C = \hat{C}$, we optimize over the choice of $(C_\ell, \alpha_\ell)_{\ell \in [d]}$, and finally choose the value of C and $(C_\ell, \alpha_\ell)_{\ell \in [d]}$ that minimizes $R(C, (C_\ell, \alpha_\ell)_{\ell \in [d]})$. The challenging step is the one computing $\operatorname{argmin}_{(C_\ell, \alpha_\ell)_{\ell \in [d]}} R(\widehat{C}, (C_\ell, \alpha_\ell)_{\ell \in [d]})$. In our experiments, we use the method `scipy.optimize.minimize` [7], but in general any off-the-shelf optimizer could be used. In the worst-case, even if we do not minimize the objective exactly, it is still better than choosing the parameters in an ad hoc manner.

We show below that in fact $R(\widehat{C}, (C_\ell, \alpha_\ell)_{\ell \in [d]})$ is a non-convex objective in the parameters $(C_\ell, \alpha_\ell)_{\ell \in [d]}$, which can be hard to optimize in general. Nevertheless, we show that the objective is convex in $(C_\ell)_{\ell \in [d]}$ and $(\alpha_\ell)_{\ell \in [d]}$ separately. To recall,

Definition 4.3. A function $f : \mathbb{R}^t \to \mathbb{R}$ is *convex* if for all $x, y \in \mathbb{R}^t$ and $\lambda \in [0,1]$, it holds that $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$.

Bias term. To simplify notation, let $\mathrm{rem}_{C_\ell}(v) := v - \mathrm{clip}_{C_\ell}(v) = \max\{0, v - C_\ell\}$. For $\ell \in [d]$, we have

$$\begin{aligned} &V_D(q_\ell)_j - \mathbb{E}\, U_{\ell,j} \\ &= \textstyle\sum_{x \in X_j} \sum_{z \in \mathcal{Z}_j^x} q_\ell(z) - \sum_{x \in X_j} \sum_{z \in \overline{\mathcal{Z}}_j^x} \mathrm{clip}_{C_\ell}(q_\ell(z)) \\ &= \textstyle\sum_{x \in X_j} \sum_{z \in \mathcal{Z}_j^x \setminus \overline{\mathcal{Z}}_j^x} q_\ell(z) + \sum_{x \in X_j} \sum_{z \in \overline{\mathcal{Z}}_j^x} \mathrm{rem}_{C_\ell}(q_\ell(z)) \\ &=: B_{\ell,j}(C) + A_{\ell,j}(C_\ell, C) \end{aligned} \tag{4}$$

We observe that $\mathrm{rem}_{C_\ell}(q_\ell(z))$ is convex in C_ℓ, and hence $A_{\ell,j}(C_\ell)$ is a convex function in C_ℓ. Moreover, since $(V_D(q_\ell)_j - \mathbb{E}\, U_{\ell,j})$ is non-negative, we have the following, where we use the fact that the square of a non-negative convex function is convex.

Observation 4.4. *$(V_D(q_\ell)_j - \mathbb{E}\, U_{\ell,j})^2$ is convex in C_ℓ.*

Variance term. To simplify the optimization, we use the following relaxations in our calculations, that are obtained by (i) approximating $\mathsf{Var}(\mathrm{DLap}(a)) \approx 2/a^2$ for $a \ll 1$, since we consider $\varepsilon \ll \Gamma$, (ii) ignoring the variance due to randomized rounding,[8] and (iii) approximating $\lfloor \alpha_\ell \Gamma / C \rfloor$ and $\lfloor \Gamma / C \rfloor$ as $\alpha_\ell \Gamma / C$ and Γ / C respectively.

Relaxation 4.5. *We use the following approximations.*

$$\mathsf{Var}(U_{\ell,j}) \approx \frac{2C^2 C_\ell^2}{\alpha_\ell^2 \varepsilon^2} \quad \textit{and} \quad \mathsf{Var}(U_{0,j}) \approx \frac{2(d+1)C^2}{\varepsilon^2}.$$

Finally, we note that the function a^2/b^2 is non-convex in (a, b).

Observation 4.6. *Under Relaxation 4.5, for fixed $C \in \mathbb{Z}_{\geq 0}$ and $\ell \in [d]$, $\mathsf{Var}(U_{\ell,j})$ is*

- ▷ *convex in C_ℓ for a fixed α_ℓ.*
- ▷ *convex in α_ℓ for a fixed C_ℓ.*
- ▷ *non-convex in joint variables (C_ℓ, α_ℓ).*

Putting it together. Thus, combining Observation 4.4 and Observation 4.6, we have the following.

Theorem 4.7. *For fixed $C \in \mathbb{Z}_{\geq 0}$, $\mathsf{RMSRE}_{\boldsymbol{\tau}}(U, (q_\ell)_{\ell=0}^{d}; D)^2$ is*

- ▷ *convex in $(C_\ell)_{\ell \in [d]}$ for fixed $(\alpha_\ell)_{\ell \in [d]}$, and*
- ▷ *convex in $(\alpha_\ell)_{\ell \in [d]}$ for fixed $(C_\ell)_{\ell \in [d]}$,*
- ▷ *non-convex in joint variables $(C_\ell, \alpha_\ell)_{\ell \in [d]}$.*

5 EXPERIMENTAL EVALUATION

5.1 Datasets

5.1.1 *Real-World Datasets.*

[8]The variance due to rounding in $W_{\ell,j}$ is at most $|X_j| \cdot C/4$, which we view as much smaller than $\mathsf{Var}(\mathrm{DLap}(\varepsilon/\Gamma)) \approx (\Gamma/\varepsilon)^2$. This is reasonable because, e.g., if $\varepsilon = 1$, then $(\Gamma/\varepsilon)^2 = 2^{32}$, which is typically order of magnitude larger than $|X_j| \cdot C/4$ in practice.

Ad-tech Real Estate Dataset. This dataset consists of approximately 100,000 real estate conversions from a 30-day period. We consider the following attributes: three known impression level features `F1`, `F2`, `F3`, and two unknown conversion features `Price` and `Quantity`.

Ad-tech Travel Dataset. This dataset consists of approximately 30,000 travel conversions from a 30-day period. We consider the following attributes: three known impression level features `F1`, `F2`, `F3`, and two unknown conversion features `Price` and `Quantity`.

These proprietary datasets were provided to us by an ad-tech that wished to remain anonymous. The datasets were collected and used consistently with local regulations and terms of use.

5.1.2 *Synthetic Data.*

The impact of various options to use the ARA can be evaluated by testing different configurations. However, such empirical evaluation would require access to a conversion dataset. Access to conversion datasets can be restricted and slow due to privacy concerns, or such data may not be available to practitioners. One way to address these difficulties is to use synthetic data that replicates the characteristics of real data that is bucketed by the summary reports in ARA.

In this context, we present a method for generating synthetic data through statistical modeling of actual conversion datasets. Initially, we performed an empirical analysis of these real conversion datasets to uncover relevant characteristics for ARA. Specifically, we examined the count and value distributions within these real conversion datasets. Subsequently, we designed a pipeline that employs the acquired distribution knowledge to create a realistic synthetic dataset, customizable by provided input parameters. In the following sections, we elaborate on the distributions as well as the process of generating data using this pipeline.

Name	Feature type	Side
`campaignId`	Categorical(16)	Impression
`geography`	Categorical(8)	
`productCategory`	Categorical(2)	
`conversionType`	Categorical(5)	Conversion
`value`	$\in \mathbb{R}_{\geq 0}$	

Table 3: Impression and conversion side features.

5.1.3 *Dataset Generation.*

Let us assume *Du & Penc* runs various Ad campaigns with the features shown in Table 3. Records relevant to the ARA are outlined below:

(1) **Impressions:** For every display of an advertisement, an impression record is generated on the client side. For a specific key, e.g., '`campaignId=1 & geography=3 & productCategory=2`', there could be a few or numerous impressions. Modeling the distribution of these impressions is the initial aspect to address.
(2) **Conversions:** An impression might lead to zero, one, or multiple conversion events. These conversion events are defined within the ad-tech context and encompass various activities, such as *click*, *add-to-cart*, *purchase*, *spend-30-seconds*, and *achieved-level-2* to provide a few examples. So the next aspect to model is the count of conversions per impression, as well as the conversion features associated to it, such as `conversionType`.
(3) **Value Contributions:** Not every conversion yields the same return for advertisers. For instance, a purchase of \$25 might be more desirable than one of \$5. Beyond simply considering the number of conversions, it is crucial to take into account the value that these conversions generate. This leads us to the distribution of conversion values, which captures this aspect.

We propose a pipeline that generates both counts and values as shown in Figure 4. Here, the data is not aggregated so that event level processing such as count bounding and contribution budgeted could be performed. First, we define the distributions that are used in the pipeline.

Definition 5.1 (Power Law Distributions)**.** The *Power Law distribution* with parameter $b > 0$ is the distribution supported on positive integers, whose probability mass function at integer k is

$$\Pr[X = k] = \begin{cases} \frac{k^{-b}}{\sum_{k=k_{\min}}^{k_{\max}} k^{-b}} & k_{\min} \leq k \leq k_{\max} \\ 0 & \text{otherwise}, \end{cases}$$

where b is the shape parameter, $k_{\min} > 0$ is the lower bound and $k_{\max}$ is the upper bound.

A well-behaved distribution typically exhibits parameters of $1 < b < 3$, $k_{\min} = 1$ and $k_{\max} = \infty$. However, when working with real datasets, it is common to observe power-law behavior within a specific range and arbitrary b parameter.

Definition 5.2 (Poisson Distribution)**.** The *Poisson distribution* with parameter $\lambda > 0$ is a discrete probability distribution whose probability mass function at integer k is

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!},$$

where the parameter λ is the average rate of events.

Definition 5.3 (Log-Normal Distribution)**.** The *Log-Normal distribution* with parameters μ and σ is a continuous probability distribution whose probability density function is

$$f(x) = \frac{1}{\sigma x \sqrt{2\pi}} \exp\left(-\frac{(\log x - \mu)^2}{2\sigma^2}\right).$$

Having defined the relevant distributions, we now describe the data generation in more detail. For convenience of the theoretical analysis in the next section, we will describe the data generation for a general choice of distributions:

- ▹ $\mathcal{D}_{\text{count}}^{\text{imp}}$: the distribution of number of impressions per slice. In our experiments, this is set to the power-law distribution (with pre-specified parameter b).
- ▹ $\mathcal{D}_{\text{count}}^{\text{conv}}$: the distribution of the number of conversions per impression. In our experiments, this is set to the Poisson distribution (with pre-specified parameter λ). The conversions are subsequently divided uniformly among the different values of attributed keys.
- ▹ $\mathcal{D}_{\text{value}}^{\text{conv}}$: the distribution of conversion values. In our experiments, this is set to the Log-normal distribution (with pre-specified parameter μ, σ). Figure 5 displays conversion values extracted from two datasets alongside the corresponding Log-Normal distribution fit.

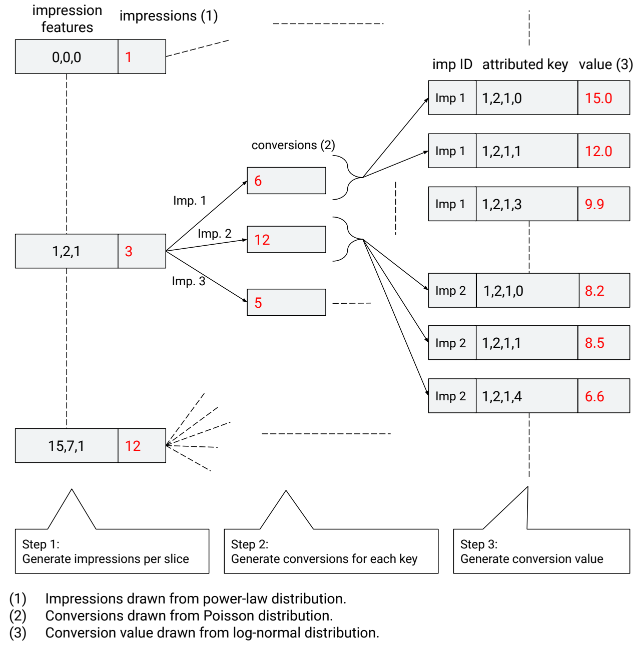

impression features
impressions (1)
0,0,0
1
imp ID
attributed key
value (3)
Imp 1 1,2,1,0 15.0
Imp 1 1,2,1,1 12.0
Imp 1 1,2,1,3 9.9
conversions (2)
6
Imp. 1
Imp. 2
12
1,2,1
3
Imp. 3
5
Imp 2 1,2,1,0 8.2
Imp 2 1,2,1,1 8.5
Imp 2 1,2,1,4 6.6
15,7,1
12
Step 1: Generate impressions per slice
Step 2: Generate conversions for each key
Step 3: Generate conversion value
(1) Impressions drawn from power-law distribution.
(2) Conversions drawn from Poisson distribution.
(3) Conversion value drawn from log-normal distribution.

Figure 4: Overall dataset generation steps with features in Table 3 used for illustration.

Our data generation pipeline works in the following stages:

Step 1 For each combination of impression features, sample #impressions is independently sampled from the distribution $\mathcal{D}^{\text{imp}}_{\text{count}}$ [9]. There will be

$$T = \prod_{\mathcal{X}_i \in \mathcal{X}} |\mathcal{X}_i|$$

attributed slices, where $\mathcal{X}$ represents the set of dimensions within the impression side. To illustrate, in the sample case shown in Table 3, there will be $T = 16 \times 8 \times 2$ slices.

Step 2 For each impression, independently sample #conversion from the distribution $\mathcal{D}^{\text{conv}}_{\text{count}}$, and distribute each one uniformly at random between the various conversion features. In the case of Table 3, there are 5 values of `conversionType`.

Step 3 For each conversion, independently sample the conversion value from the distribution $\mathcal{D}^{\text{conv}}_{\text{value}}$.

Name	Step 1	Step 2	Step 3	
	b	λ	μ	σ
`synth-real-estate`	1.03	10	0.87	0.43
`synth-travel`	1.14	10	1.95	1.14

Table 4: Synthetic datasets utilized in evaluations with parameters that mimic the corresponding real datasets.

Table 4 presents two synthetic datasets that were employed in evaluations with parameters. It is possible to generate numerous datasets with specific parameters that closely mimicking the characteristics of a target dataset. This could be particularly useful for emulating privacy-restricted proprietary ad datasets.

[9]Sampling from discrete power-law distributions with arbitrary parameters b is not a straightforward process. To address this challenge, we adopted the approximation method outlined in Appendix D of the work by Clauset et al. in [10]

5.2 Setup

We evaluate our algorithms on two real-world datasets and two synthetic datasets, which are described in more detail in Section 5.1. Each dataset is partitioned into a training set and a test set. For the real-world datasets, the partition is based on timestamps; for the synthetic data, separate training and test sets are generated independently from the data distribution. The training set is used to choose contribution budgets and clipping threshold parameters, and the error is evaluated on the test set. For the synthetic datasets, the training set is also used to choose a count limit C; for the real-world datasets only click-level or conversion-level data is available, so the count limit is set to 1.

We optimize using the Sequential Least-SQuares Programming (SLSQP) optimizer provided in `scipy.optimize.minimize` [7], which is a quasi-Newton method. This optimizer runs in time cubic in the number of variables of the optimization problem, which is twice the number of queries d. In our setting, for each dataset and parameter choice, we have $d \in \{1, 2\}$ and optimization took between 1–75 seconds. Our experiments are based on 15 days of data, but ad-techs may choose to perform optimization more or less frequently depending on their data distribution's temporal stability.

We compare our optimization-based algorithm to several *baseline* approaches that use fixed contribution budgets α_ℓ and clipping thresholds C_ℓ set to a fixed quantile of the training data. For the first baselines, we use equal contribution budgets for each query, including the count query, so that $\alpha_0 = \cdots = \alpha_d = \frac{1}{d+1}$. This is the approach outlined in the API documentation [20], and represents the only previously published approach that we are aware of for using the API. The clipping threshold quantiles used are 95% and 99% for the real-world datasets, and 90% and 95% for the synthetic datasets. Note that to choose these thresholds, the baselines also require access to training data.

We also compare against some more tailored baselines that take into account domain-specific knowledge that an ad-tech using the API may possess. In particular, the count query tends to be easier to answer accurately (i.e. incurs lower error for a fixed contribution budget) than the other queries, due to the fact that all values have the same scale and no signal is lost due to clipping. Consequently, it may be desirable to allocate a lower contribution budget to the count query and a higher contribution budget to the remaining query or queries. For each dataset we consider two additional contribution budgeting strategies, setting the contribution budgets to be in a 2:2:1 or 10:10:1 ratio for the real-world datasets and a 2:1 or 5:1 ratio for the synthetic datasets. For each contribution budget strategy, we again choose clipping thresholds determined by the 95% and 99% quantiles of the training data for the real-world datasets and the 90% and 95% quantiles for the synthetic datasets.

For each dataset, we partition it into slices based on one or more impression features, and estimate multiple queries corresponding to each slice. For the real-world datasets we consider three queries for each slice, corresponding to the count and two additional conversion features depending on the dataset. For the synthetic datasets

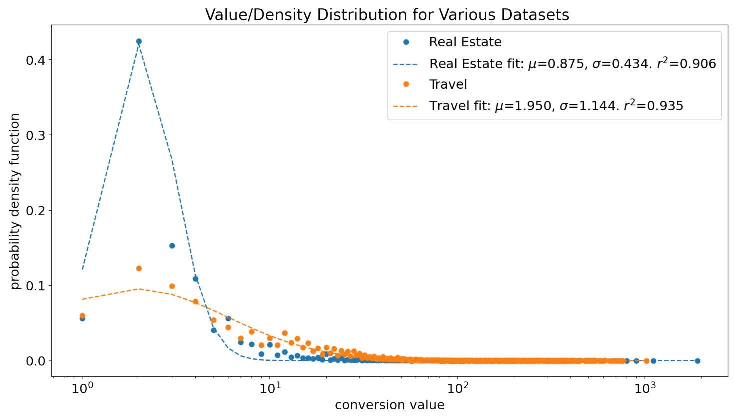


Figure 5: Scatter plots of real-world datasets illustrating the probability of observing a conversion value. The fitted curves represent best log-normal distribution models that effectively capture the underlying patterns in the data.

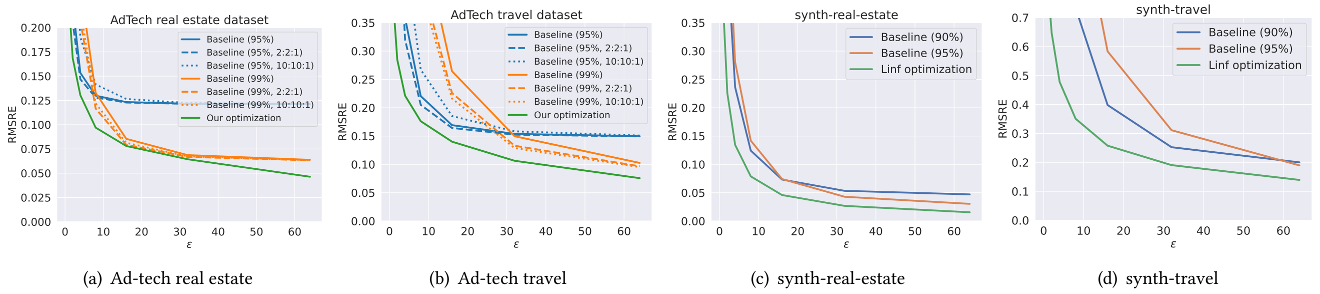


Figure 6: RMSRE$_\tau$ for privacy budgets $\{1, 2, 4, 8, 16, 32, 64\}$ for our algorithms and baselines on two real-world and two synthetic datasets. Our optimization-based approach consistently achieves lower error than baselines that use a fixed quantile for the clipping threshold and split the contribution budget equally among the queries.

we consider two queries for each slice, corresponding to the count and a single conversion feature.

For the error metric RMSRE$_\tau$, for each query q_ℓ we choose τ_ℓ to be five times the median value of the query on the records of the training dataset. This ensures invariance of the error metric to rescaling the data, and allows us to combine the errors from features of different scales by taking $\boldsymbol{\tau} = (\tau_0, \dots, \tau_d)$.

5.3 Results

We see in Figure 6 that the estimates produced by our algorithms have substantially lower error than the baselines, on both the real-world and synthetic datasets. Moreover, the excess error incurred by each baseline depends on the data and overall privacy budget, with different baselines performing better or worse in different parameter regimes. In contrast, our optimization-based approach is able to adapt to the privacy budget and data, and consistently outperforms the baselines, often by a large margin. For the two real-world datasets the improvements range from 2–36% to 14–66%, respectively, depending on the value of the privacy parameter ε. For the two synthetic datasets the improvements range from 36–60% to 18–83%, respectively, again depending on the value of ε.[10]

In Appendix B and Figures 7–8 we show that our algorithmic approach can also be applied to other error metrics such as RMSE and ARE, and provides accuracy improvements for these as well. In fact, for most datasets and parameter choices our algorithm achieves an even larger improvement over the baselines, as large as a 96% improvement for RMSE and a 85% improvement for ARE on the real-world datasets. This underscores the flexibility of our approach: in contrast to the baselines, which perform inconsistently depending on the dataset, error metric, and choice of the privacy parameter, our adaptive approach achieves strong performance in each setting.

In Appendix C and Figure 9 we show that an ℓ_1-based variant of our optimization-based algorithm in Figure 9 provides modest additional accuracy improvements on some datasets.

[10]These improvements are calculated by comparing our algorithm to the best of six baselines for each dataset and each value of ε. Since we run multiple baselines on the test data and only compare to the best, this may allow the baselines to overfit to the test data. Consequently, this evaluation may actually understate our improvement on some inputs.

6 GENERALIZATION BOUNDS

Since we optimize the parameters on the historical dataset, it is important to ensure that we are not overfitting to this training dataset in such a way that it performs badly on the actual (i.e., test) dataset. To support our empirical findings, in this section, we formally prove—in a simplified setting—a generalization bound showing that the expected RMSRE on the actual dataset is close to optimal even with this procedure.

We work in the data generation model as in the previous section. For the purpose of theoretical analysis, we consider a simplified setting where: (i) there is only one conversion per impression (i.e., $\mathcal{D}^{\text{conv}}_{\text{count}}$ is the point-mass distribution that is always equal to one) and (ii) that there is only a single query (i.e., $d = 1$).

Recall the notations from Section 4. Due to (i), we always set the per-impression count capping to $C = 1$; this also gives $\mathcal{Z}^x = \overline{\mathcal{Z}}^x$ for all impression x. For convenience, we also define the following notations:

$$\begin{aligned}
\mathrm{bi}_{C_1}(D_j) &:= \sum_{x \in X_j} \sum_{z \in \mathcal{Z}^x} \mathrm{rem}_{C_1}(q_1(z)_j), \\
\mathrm{nc}_{C_1}(D_j) &:= \sum_{x \in X_j} \sum_{z \in \mathcal{Z}^x} \mathbf{1}[q_1(z)_j > C_1], \\
\pi(D_j) &:= \max\{\tau_1, V_D(q_1)_j\}, \\
R_{D_j}(C_1) &:= \frac{1}{\pi(D_j)^2}\left(\mathrm{bi}_{C_1}(D_j)^2 + \frac{2C_1^2}{\varepsilon^2}\right),
\end{aligned}$$

where recall that X_j is the set of all x such that $(x, y) \in D_j$ for some y, and R is similar to Equation (3), but here, we only have one argument, namely C_1. $\mathrm{bi}_{C_1}(D_j)$ is the **bi**as incurred in the estimate due to clipping; this is similar to the term in Equation (4), and $\mathrm{nc}_{C_1}(D_j)$ counts the **n**umber of conversion values that were **c**lipped by the threshold at C_1. As stated earlier, we assume that the number of impressions in the jth slice is generated by $\mathcal{D}^{\text{imp}}_{\text{count}}$ (with one conversion per impression) and the value of each conversion is generated independently by $\mathcal{D}^{\text{conv}}_{\text{value}}$. We denote this entire compound distribution by $\mathcal{D}_{\text{comp}}$.

Finally, we let

$$\tilde{R}_{\mathcal{D}_{\text{comp}}}(C_1) := \mathbb{E}_{D \sim \mathcal{D}_{\text{comp}}}\left[R_D(C_1)\right],$$

denote the expected loss $R_D(C_1)$, where D is drawn from the distribution $\mathcal{D}_{\text{comp}}$.

In this simplified setting, the optimization objective reduces to just minimizing

$$\begin{aligned}
R_{D_1,\dots,D_m}(C_1) &:= \frac{1}{m}\left(R_{D_1}(C_1) + \cdots + R_{D_m}(C_1)\right) \\
&= \frac{1}{m}\sum_{j \in [m]} \frac{1}{\pi(D_j)^2}\left(\mathrm{bi}_{C_1}(D_j)^2 + \frac{2C_1^2}{\varepsilon^2}\right).
\end{aligned}$$

Differentiating this (w.r.t. C_1), we get

$$\begin{aligned}
&\frac{\partial}{\partial C_1} R_{D_1,\dots,D_m}(C_1) \\
&= \frac{1}{m}\sum_{j \in [m]} \frac{1}{\pi(D_j)^2}\left(\frac{4C_1}{\varepsilon^2} - 2\mathrm{bi}_{C_1}(D_j)\cdot \mathrm{nc}_{C_1}(D_j)\right).
\end{aligned}$$

In other words, the optimum contribution bounding threshold $C_1^* = C_1^*(D_1, \dots, D_j)$ is such that

$$\sum_{j \in [m]} \frac{1}{\pi(D_j)^2}\left(\frac{4C_1^*}{\varepsilon^2} - 2\mathrm{bi}_{C_1^*}(D_j)\cdot \mathrm{nc}_{C_1}(D_j)\right) = 0.$$

Let $\mu_n(\mathcal{D})$ denote the nth moment of the distribution $\mathcal{D}$ over $\mathbb{R}$, namely, $\mu_n(\mathcal{D}) = \mathbb{E}_{x\sim\mathcal{D}}\,|x|^n$. We can get the following generalization bound. Note that the LHS is the expected error of the fresh (independent) slice if we optimize based on the $D_1, \dots, D_m$ (i.e., historical data) drawn from the same distribution $\mathcal{D}_{\text{comp}}$, while the RHS is the expected error with respect to the optimal threshold for the distribution.

Theorem 6.1. *For any distributions $\mathcal{D}^{\text{imp}}_{\text{count}}, \mathcal{D}^{\text{conv}}_{\text{value}}$ such that the moments $\mu_4(\mathcal{D}^{\text{imp}}_{\text{count}})$ and $\mu_2(\mathcal{D}^{\text{conv}}_{\text{value}})$ are finite. For any $\zeta, \theta > 0$, there exists $m \in \mathbb{N}$ such that, with probability $1-\zeta$ over $D_1, \dots, D_m \sim \mathcal{D}_{\text{comp}}$, and using $C_1^* = C_1^*(D_1, \dots, D_m)$, we have*

$$\tilde{R}_{\mathcal{D}_{\text{comp}}}(C_1^*) \;\le\; \min_{\tilde{C}_1 \ge 0} \tilde{R}_{\mathcal{D}_{\text{comp}}}(\tilde{C}_1) + \theta.$$

Proof. Let $\tilde{C}_1^* = \operatorname{argmin}_{\tilde{C}_1 \ge 0} \tilde{R}_{\mathcal{D}_{\text{comp}}}(\tilde{C}_1)$ denote the optimal clipping threshold of the distribution. Since $\tilde{R}_{\mathcal{D}_{\text{comp}}}(\tilde{C}_1)$ is a continuous function, there exists $\lambda > 0$ such that

$$\tilde{R}_{\mathcal{D}_{\text{comp}}}(C_1) - \tilde{R}_{\mathcal{D}_{\text{comp}}}(\tilde{C}_1^*) \le \theta. \quad \forall C_1 \in [\tilde{C}_1^* - \lambda, \tilde{C}_1^* + \lambda]. \tag{5}$$

Furthermore, let $\nu := \mathbb{E}[\frac{1}{\pi(X)^2}] > 0$.

Recall that

$$\frac{\partial}{\partial C_1}\tilde{R}_X(\tilde{C}_1^*) = \frac{1}{\pi(X)^2}\left(\frac{4\tilde{C}_1^*}{\varepsilon^2} - 2\mathrm{bi}_{\tilde{C}_1^*}(X)\cdot \mathrm{nc}_{\tilde{C}_1^*}(X)\right).$$

Note that $\mathbb{E}_{X\sim\mathcal{D}_{\text{comp}}}\left[\frac{\partial}{\partial C_1}\tilde{R}_X(\tilde{C}_1^*)\right] = 0$ (due to $\tilde{C}_1^*$ being the minimizer). Furthermore, we have

$$\begin{aligned}
&\mathbb{E}_{X\sim\mathcal{D}_{\text{comp}}}\left[\left(\frac{\partial}{\partial C_1}\tilde{R}_X(\tilde{C}_1^*)\right)^2\right] \\
&\le O\left(\frac{1}{\pi^2}\left(\frac{(\tilde{C}_1^*)^2}{\varepsilon^4} + \mathrm{bi}_{\tilde{C}_1^*}(X)^2\cdot \mathrm{nc}_{\tilde{C}_1^*}(X)^2\right)\right) \\
&\le O\left(\frac{1}{\pi^2}\left(\frac{(\tilde{C}_1^*)^2}{\varepsilon^4} + \mathrm{bi}_0(X)^2\cdot \mathrm{nc}_0(X)^2\right)\right) \\
&= O\left(\frac{1}{\pi^2}\left(\frac{(\tilde{C}_1^*)^2}{\varepsilon^4} + \mu_2(\mathcal{D}^{\text{conv}}_{\text{value}})\mu_4(\mathcal{D}^{\text{imp}}_{\text{count}})\right)\right),
\end{aligned}$$

which is finite under the assumption in the theorem statement. Thus, $\frac{\partial}{\partial C_1}\tilde{R}_X(\tilde{C}_1^*)$ when $X \sim \mathcal{D}_{\text{comp}}$, $\tilde{R}'_X(\tilde{C}_1^*)$ has a finite variance. Similarly, the term $\frac{1}{\pi(X)^2}$ has a finite variance, simply because its maximum value is at most $1/\tau_1^2$. Thus, for any $\zeta, \theta > 0$, there exists m_0 such that for any $m \ge m_0$, with probability $1 - \theta$ over $D_1, \dots, D_m \sim \mathcal{D}_{\text{comp}}$, both of the following hold:

$$\left|\frac{\partial}{\partial C_1} R_{D_1,\dots,D_m(C_1)}\right| = \left|\frac{1}{m}\sum_{j\in[m]} \frac{\partial}{\partial C_1} R_{D_j}(C_1^*)\right| \le 4\lambda\nu/\varepsilon^2,$$

and

$$\frac{1}{m}\sum_{j\in[m]}\frac{1}{\pi(D_j)^2} \geq \nu/2.$$

Now, notice that the objective $R_{D_1,\ldots,D_m}(C_1)$ is $\left(\frac{4}{m\varepsilon^2}\sum_{j\in[m]}\frac{1}{\pi(D_j)^2}\right)$-strongly convex. As a result, when the above two inequalities hold we have

$$|\tilde{C}_1^* - C_1^*| \leq \frac{\left|\frac{\partial}{\partial C_1}R_{D_1,\ldots,D_m}(C_1)\right|}{\left(\frac{4}{m\varepsilon^2}\sum_{j\in[m]}\frac{1}{\pi(D_j)^2}\right)} \leq \lambda.$$

From (5), this implies $\tilde{R}_{\mathcal{D}_{\text{comp}}}(C_1^*) - \tilde{R}_{\mathcal{D}_{\text{comp}}}(C_1^*)$. □

We remark that, due to the use of continuity argument of $\tilde{R}_{\mathcal{D}_{\text{comp}}}$ (at $\tilde{C}_1^*$), we do not achieve any explicit bound in the rate of convergence. It remains an interesting question to extend this argument to get a specific rate. Similarly, it remains interesting to incorporate the privacy budgets (i.e., α_ℓ's) in the presence of multiple queries to the bounds as well.

7 RELATED WORK

The work closest to ours is that on optimizing hierarchical queries when using the Attribution Reporting API [12]. In our terminology, this corresponds to aggregating with respect to multiple partitions of $\mathcal{Z}$ that are refinements of each other; our work is complementary in that we focused on a single partition. In addition, our setting involves aggregating different conversion values for each slice. Thus, while [12] optimized for contribution budget allocation across different slices, our work optimizes the contribution budget allocation across the different queries for each slice. Furthermore, [12] also involved post-processing the estimates that ensured consistency of estimates and reduced the overall noise; this was done by generalizing the methods in [11, 16]. Such post-processing is not relevant in our context as we do not have any consistency constraints that are satisfied by the noiseless data. Hence, it is possible to combine the techniques in our work with the techniques in [12] to consider a setting where we have hierarchical queries with multiple conversion values to aggregate.

Private aggregation by contribution bounding and adding noise is a common technique in DP. It was shown in [9] that in order to minimize the ℓ_1-error, the optimal threshold is to set the contribution bound to be the $(1-1/\varepsilon n)$th percentile of the data. On the other hand, in order to minimize the ℓ_2^2-error, it was shown in [17] that bounding the range and adding Laplace noise achieves the smallest error, thereby establishing a bias-variance-privacy trilemma; this is precisely what we get in our approach as well, where we clip the value range and add (discrete) Laplace noise, by optimizing the clip threshold using historical data.

8 LIMITATIONS AND FUTURE DIRECTIONS

One limitation of our approach is that it relies on access to noiseless historical data in order to optimize the contribution bounding parameters for querying future data. While such non-contribution bounded noiseless data might still be available for long-running campaigns, new campaigns launched well after the deprecation of third-party cookies would benefit from methods for continuously updating the contribution bounding parameters based solely on the outputs of privacy-preserving APIs.

A very interesting direction for future work is to develop algorithms that rely on noisy privatized historical data or privately generated aggregate statistics for historical data instead of noiseless historical data. An approach based on this work would be to learn the parameter(s) of the synthetic data distribution using past data, and then sample repeatedly from this distribution to construct a synthetic dataset that can be used for privacy budgeting for queries on future data.

Another limitation is that the optimization problem is discrete, since the contribution budgets must be positive integers. To avoid the computational challenges of discrete optimization, we instead work with a continuous approximation of the optimization problem. Our results show that working with this continuous approximation is sufficient for obtaining a high-accuracy solution to the original discrete problem.

In addition to summary reports, ARA offers *event-level reports* [4] which are also subject to (a different type of) contribution bounding and noising; our method does not take these reports into account when setting the contribution bounds for summary reports. It would be interesting to explore whether event-level reports can be leveraged to optimize the summary reports in ARA.

As described in Section 3.1, summary reports in ARA are currently restricted by on-client attribution and by the *separate* computation of the contributions of different attributed conversions. It would be interesting to determine the utility improvement that could be achieved if the contributions of different attributed conversions can be computed *jointly*, e.g., if attribution were to be done off-client either in a trusted execution environment or via a secure multi-party computation protocol, or alternatively if the contributions of an attributed conversion can simply take into account the contributions of previously attributed conversions on the same client.

9 CONCLUSION

In this work, we studied the optimization of summary reports in the ARA, which is currently deployed on hundreds of millions of Chrome browsers. To the best of our knowledge, there has been no prior work formulating the contribution budgeting optimization problem for ARA. We hope that our rigorous formulation will equip researchers with the right abstraction of the problem as well as the API to develop DP algorithms for ad conversion measurement with better privacy-utility trade-offs.

Our recipe, which leverages past data that is noiseless and that has not been bounded, in order to bound the contributions in future data when querying it with DP, is quite general and applicable to settings (beyond advertising) where a system is queried continuously over time, and a DP constraint is being continuously enforced.

ACKNOWLEDGMENTS

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

[1] Aggregation service for the attribution reporting api. https://github.com/WICG/attribution-reporting-api/blob/main/AGGREGATION_SERVICE_TEE.md.

[2] Attribution reporting: Aggregatable reports api. https://developer.android.com/design-for-safety/privacy-sandbox/attribution#aggregatable-reports-api.

[3] Attribution Reporting API with Aggregatable Reports: Privacy Considerations. https://github.com/WICG/attribution-reporting-api/blob/main/AGGREGATE.md#privacy-considerations.

[4] Attribution Reporting: Event-level Reports. https://developer.android.com/design-for-safety/privacy-sandbox/attribution#event-level-reports.

[5] Contribution budget for summary reports. https://developer.chrome.com/docs/privacy-sandbox/attribution-reporting/contribution-budget/.

[6] MaskedLARk. https://github.com/microsoft/maskedlark.

[7] scipy.optimize.minimize.

[8] SKAdNetwork. https://developer.apple.com/documentation/storekit/skadnetwork/.

[9] Amin, K., Kulesza, A., Munoz, A., and Vassilvtiskii, S. Bounding user contributions: A bias-variance trade-off in differential privacy. In *ICML* (2019), pp. 263–271.

[10] Clauset, A., Shalizi, C. R., and Newman, M. E. Power-law distributions in empirical data. *SIAM Review 51*, 4 (2009), 661–703.

[11] Cormode, G., Procopiuc, C., Srivastava, D., Shen, E., and Yu, T. Differentially private spatial decompositions. In *ICDE* (2012), pp. 20–31.

[12] Dawson, M., Ghazi, B., Kamath, P., Kumar, K., Kumar, R., Luan, B., Manurangsi, P., Mundru, N., Nair, H., Sealfon, A., and Zhu, S. Optimizing Hierarchical Queries for the Attribution Reporting API. In *AdKDD* (2023).

[13] Dwork, C., McSherry, F., Nissim, K., and Smith, A. D. Calibrating noise to sensitivity in private data analysis. In *TCC* (2006), pp. 265–284.

[14] Dwork, C., and Roth, A. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science 9*, 3–4 (2014), 211–407.

[15] Gozman, V. The slow death of third-party cookies. Forbes, September 2022. https://www.forbes.com/sites/theyec/2022/09/12/the-slow-death-of-third-party-cookies/amp/.

[16] Hay, M., Rastogi, V., Miklau, G., and Suciu, D. Boosting the accuracy of differentially-private histograms through consistency. *VLDB* (2010).

[17] Kamath, G., Mouzakis, A., Regehr, M., Singhal, V., Steinke, T., and Ullman, J. A bias-variance-privacy trilemma for statistical estimation. In *TPDP* (2023).

[18] Kruppa, M., and Haggin, P. Google is finally killing cookies. Advertisers still aren't ready. Wall Street Journal, January 2024. https://www.wsj.com/tech/google-is-finally-killing-cookies-advertisers-still-arent-ready-7582fcac.

[19] Nadan, A., White, A., Cucu, A., Nalpas, M., and Mastromatto, Z. Experiment with summary report design decisions, November 2022. https://developer.chrome.com/docs/privacy-sandbox/summary-reports/design-decisions/.

[20] Nalpas, M., and White, A. Attribution Reporting, May 2021. https://developer.chrome.com/en/docs/privacy-sandbox/attribution-reporting/.

[21] Pfeiffer III, J. J., Charles, D., Gilton, D., Jung, Y. H., Parsana, M., and Anderson, E. Masked LARk: Masked learning, aggregation and reporting workflow. *arXiv preprint arXiv:2110.14794* (2021).

[22] Schuh, J. Building a more private web: A path towards making third party cookies obsolete, January 2020. https://blog.chromium.org/2020/01/building-more-private-web-path-towards.html.

[23] Thomson, M. Privacy Preserving Attribution for Advertising, February 2022. https://blog.mozilla.org/en/mozilla/privacy-preserving-attribution-for-advertising/.

[24] White, A. Summary reports, November 2022. https://developer.chrome.com/docs/privacy-sandbox/summary-reports/.

[25] Wilander, J. Full Third-Party Cookie Blocking and More, March 2020. https://webkit.org/blog/10218/full-third-party-cookie-blocking-and-more/.

[26] Wilander, J. Introducing Private Click Measurement, PCM, February 2021. https://webkit.org/blog/11529/introducing-private-click-measurement-pcm/.

[27] Wood, M. Today's Firefox Blocks Third-Party Tracking Cookies and Cryptomining by Default, 2019. https://blog.mozilla.org/en/products/firefox/todays-firefox-blocks-third-party-tracking-cookies-and-cryptomining-by-default/.

A ERROR METRICS FOR EVALUATING REPORTS

In Table 5 above, we specify all metrics considered for utility evaluation.

To choose a particular metric, we considered the desirable properties of an error metric that further can be used as an objective function. Ideally, a good error metric should have the following properties:

(1) **Decision Stability**: Some of our metrics are parameterized. (E.g., $\mathsf{RMSRE}_{\boldsymbol{\tau}}$ is parameterized by $\boldsymbol{\tau}$; see Definition 3.2.) For a good metric, the decision from our optimization procedure (e.g, count bound, contribution budgeting, etc.) should not be too sensitive to the choice of parameters.

(2) **Utility Stability**: The utility measured by the metric is robust to perturbations in the input (e.g., true conversion count). For instance, the metric's output should not change too much if the true conversion count is slightly changed. This is important because the true conversion count is often difficult to measure accurately.

(3) **Ease of Optimization**: The metric is easy to calculate, and the objective function based on it is easy to be optimized.

(4) **Ease of Extension**: The metric should be easy to extend to aggregates after keyspace aggregation. The metric is for a slice in aggregate API, which is any possible combination of keys (keyspace value). For example, a slice for an advertiser may look like: `impression_date='8/1', biddability='True'`. To get the total number of biddable conversions, one needs to sum up all noised counts from slices with `biddability='True'`. This accumulates a bunch of Laplace noise random variables, which is no longer Laplace. It is desirable that the slice error metric can be easily adapted to aggregates after keyspace aggregation.

(5) **Defined at Zero**: The metric should be well-defined when conversion query value is zero. This is important since the conversion data can be sparse.

(6) **Differentiates Small/Large Values**: The metric should differentiate between large and small query values. Intuitively, this is because noise added to large values will usually have less effect on downstream tasks compared to the same amount of noise added to small values.

Table 6 provides the list of criteria that each metric satisfies.

Intuition for RMSRE_τ. RMSRE_τ can be seen as a hybrid between additive and multiplicative error. When the query values are smaller than the threshold τ, it becomes (a scaled version of) the root mean squared error RMSE. Recall that

$$\mathsf{RMSE}(u, q; D) := \sqrt{\frac{1}{m} \sum_{j \in [m]} \mathbb{E}\left(u_j - V_D(q)_j\right)^2}.$$

Meanwhile, if the query values are larger than τ, then it becomes the root mean square relative error, defined as

$$\mathsf{RMSRE}(u, q; D) := \sqrt{\frac{1}{m} \sum_{j \in [m]} \mathbb{E}\left(\frac{u_j - V_D(q)_j}{V_D(q)_j}\right)^2}.$$

To give an intuition as to why RMSRE_τ is a good metric, we can compare them with RMSE and RMSRE. The main advantage of RMSRE_τ over RMSE is that RMSRE_τ can distinguish between the small and large values (criteria (6) above). Meanwhile, RMSRE_τ is defined even when the query values are zero, whereas RMSRE is undefined (criteria (5) above).

B OPTIMIZATION FOR DIFFERENT ERROR METRICS

We now show that our optimization algorithm also works well for alternate error metrics besides RMSRE_τ. In particular, we show experimental results optimizing for the error metrics RMSE in Figure 7 and for $\mathsf{ARE}_{0.1}$ in Figure 8. In each plot we see that our algorithm outperforms the baselines, often substantially.

Our method can also be applied to a variety of other error metrics, including those described in Table 5. For each of these metrics, we can compute or estimate the expected error or expected squared error for a given setting of hyperparameters, and optimize over these hyperparameters. For RMSE, like RMSRE_τ, we can represent the error as a combination of the variance from noise addition and the bias from clipping; for ARE we can compute the error by evaluating the cumulative density function of the error distribution at inputs that depend on the bias from clipping. Many of the other error metrics in Table 5 can be computed similarly.

In some of the experiments, particularly for RMSE on the real-world datasets (the first two panels of Figure 7), the baselines perform quite poorly, and it is likely that using a different quantile would improve performance somewhat. Unlike the baselines, which have varied performance depending on the dataset, the error metric, and the privacy parameter ϵ, our optimization-based algorithm consistently has the lowest error in each setting.

In a few of these experiments, the optimizer was unsuccessful at finding the minimum-error parameters. This is most visible in the small-ϵ values of Figure 7(a), where the reported errors for our algorithm are not quite monotonically decreasing in ϵ as we'd expect. It is likely that tweaking to the optimizer parameters or using a different optimization library would improve the accuracy on these inputs. Despite not necessarily achieving the true optimum and exiting with a failure status on a few inputs, the optimizer still obtained hyperparameters with error much smaller than the baseline error.

C ℓ_1 VERSION OF ALGORITHM 3

We present a variant of HistogramContribution in Algorithm 6, which uses ℓ_1-clipping instead of ℓ_∞-clipping that is employed in Algorithm 3. We show in Figure 9 that this variant can achieve a modest improvement to performance on some inputs. The main intuition for this algorithm is that ℓ_1-clipping results in less loss of signal, especially when the different query values are negatively correlated or only weakly correlated with each other. Below, we show that the histogram contributions generated this way respect the same ℓ_1-norm constraint as Algorithm 3.

Figure 9 shows the improvement achieved by this variant on the ad-tech real estate and travel datasets. For the synthetic datasets the ℓ_1-optimization is equivalent to ℓ_∞-optimization, since there is only a single non-count query on these datasets.

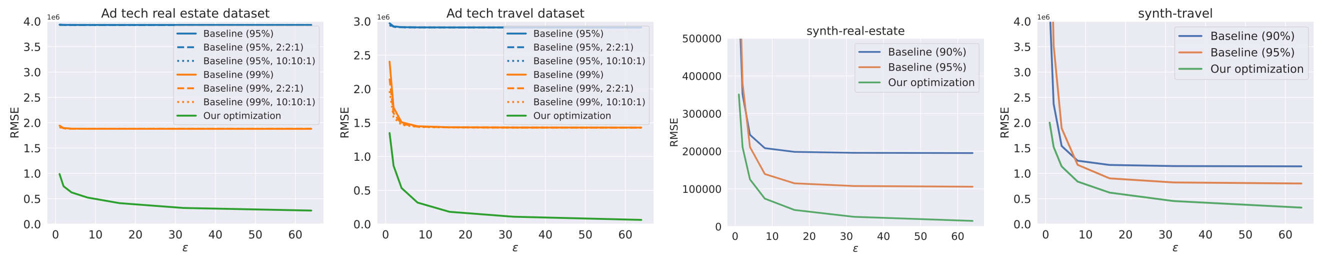
Lemma C.1. *For any z, the vector w^z returned by Algorithm 6 satisfies $w^z \geq \mathbf{0}$ and $\|w^z\|_1 = \lfloor \Gamma / C \rfloor$.*

Short Name	Error Metric (slice j)	Sample Parameters	Interpretation
ARE_α	$\Pr\left[\frac{\lvert U_j - V(q)_j\rvert}{V(q)_j} > \alpha\right]$	$\alpha \in \{0.1, 0.2\}$	Probability of seeing large relative error.
AME_τ	$\Pr\left[\lvert U_j - V(q)_j\rvert > \tau\right]$	$\tau \in \{1, 5\}$	Probability of seeing large magnitude errors.
$\mathrm{APME}_{\alpha,\tau}$	$\Pr\left[\frac{\lvert U_j - V(q)_j\rvert}{V(q)_j} > \alpha \cap \lvert U_j - V(q)_j\rvert > \tau\right]$	$\alpha \in \{0.2\}, \tau \in \{1, 5\}$	Probability of seeing large magnitude and relative errors.
EARE	$\mathbb{E}\left[\frac{\lvert U_j - V(q)_j\rvert}{V(q)_j}\right]$		Expected absolute relative error (to true value).
RMSE	$\sqrt{\mathbb{E}\left[\left(U_j - V(q)_j\right)^2\right]}$		Root mean squared error.
RMSRE	$\sqrt{\mathbb{E}\left[\left(\frac{U_j - V(q)_j}{V(q)_j}\right)^2\right]}$		Root mean squared relative error.
EARE_τ	$\mathbb{E}\left[\frac{\lvert U_j - V(q)_j\rvert}{\max(\tau, V(q)_j)}\right]$	$\tau \in \{3, 5, 10\}$	Mean absolute relative error at threshold τ.
EAREO	$\mathbb{E}\left[\frac{\lvert U_j - V(q)_j\rvert}{U_j}\right]$		Expected absolute relative error (to observation).
RMSRE_τ	$\sqrt{\mathbb{E}\left[\left(\frac{\lvert U_j - V(q)_j\rvert}{\max(\tau, V(q)_j)}\right)^2\right]}$	$\tau \in \{3, 5, 10\}$	Root mean squared relative error at threshold τ.

Table 5: Error metrics considered for noise impact measurement. U_j is observed and $V(q)_j$ true value.

Metric $\longrightarrow$	ARE_α	AME_τ	$\mathrm{APME}_{\alpha,\tau}$	EARE	RMSE	RMSRE	EARE_τ	EAREO	RMSRE_τ
Decision Stability	✗	✓	✓	✓	✓	✓	✓	✓	✓
Utility Stability	✓	✓	✓	✓	✓	✓	✓	✗	✓
Ease of Optimization	✓	✓	✗	✓	✓	✓	✓	✗	✓
Ease of Agg. Extension	✗	✗	✓	✗	✓	✓	✗	✗	✓
Defined at Zero	✗	✗	✓	✗	✓	✗	✓	✓	✓
Differentiates Small / Large	✓	✗	✓	✓	✗	✓	✓	✓	✓

Table 6: Desirable properties for different metrics.



(a) Ad-tech real estate (b) Ad-tech travel (c) synth-real-estate (d) synth-travel

Figure 7: RMSE for privacy budgets $\{1, 2, 4, 8, 16, 32, 64\}$ for our algorithms and baselines on two real-world and two synthetic datasets.

Proof. It is immediate to see that $\sum_{\ell,j} w^z_{\ell,j} + \sum_j w^z_{\perp,j} = \lfloor \Gamma / C \rfloor$. Let j be such that $z \in \mathcal{Z}_j$. Clearly $w^z_{\ell,j'} = 0$ for all $j' \neq j$. We have

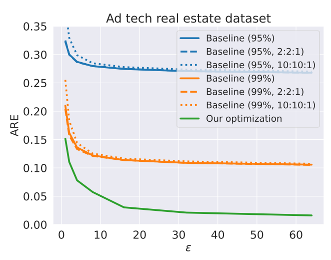
$$\sum_{\ell=1}^d w^z_{\ell,j} \;\le\; \sum_{\ell=1}^d \lfloor u_i \rfloor \;\le\; \left\lfloor \sum_{\ell=1}^d u_i \right\rfloor \;=\; \left\lfloor \frac{\Gamma}{C} \right\rfloor ,$$

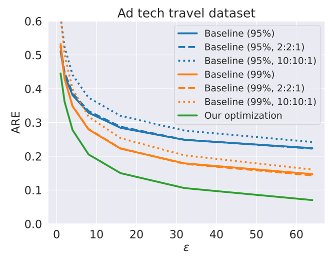
and hence $w^z_{\perp,j} = \lfloor \Gamma / C \rfloor - \sum_\ell w^z_{\ell,j} \ge 0$. □

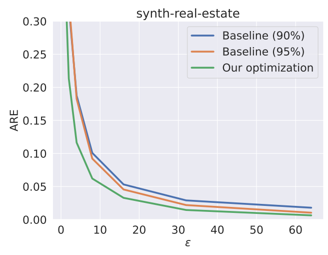
We can use the same algorithm for reconstructing the estimates (Algorithm 4), with scales $\beta_\ell = C_\ell / \lfloor \Gamma / C \rfloor$.

In Figure 9 we show that this variant with ℓ_1-clipping has slightly lower error on the real-world datasets.

The synthetic datasets have only a single non-count query, and in this setting ℓ_1-clipping is equivalent to the ℓ_∞-clipping used in the other experiments.


(a) Ad-tech real estate


(b) Ad-tech travel


(c) synth-real-estate

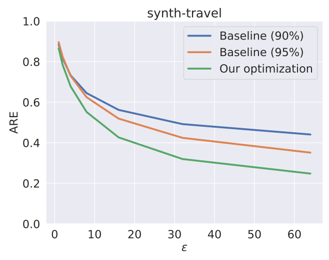
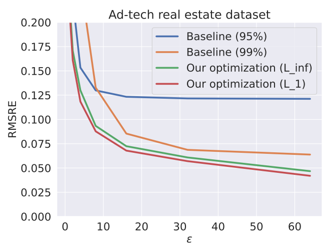

(d) synth-travel

Figure 8: ARE$_\alpha$ for $\alpha = 0.1$ and privacy budgets $\{1, 2, 4, 8, 16, 32, 64\}$ for our algorithms and baselines on two real-world and two synthetic datasets.


(a) Ad-tech real estate

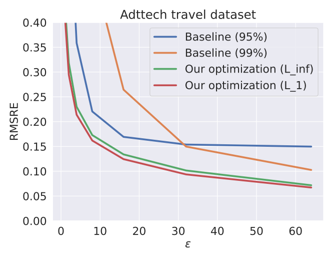

(b) Ad-tech travel

Figure 9: RMSRE$_\tau$ for privacy budgets $\{1, 2, 4, 8, 16, 32, 64\}$ for our ℓ_∞-based and ℓ_1-based optimization algorithms as well as baselines on two real-world datasets.

Algorithm 6 HistogramContribution (ℓ_1 version)

```
**Params:** ▷ Queries $q_1, \ldots, q_d : \mathcal{Z} \to \mathbb{R}_{\geq 0}$
            ▷ Partition of $\mathcal{Z} = \mathcal{Z}_1 \sqcup \cdots \sqcup \mathcal{Z}_m$
            ▷ Parameters $C$, $(C_\ell)_{\ell \in [d]}$
**Input:** Record $z \in \mathcal{Z}$
**Output:** Histogram contribution $w^z \in \mathbb{Z}_{\geq 0}^K$ for $K = ([d] \cup \{\perp\}) \times [m]$
**for** slice $j \in [m]$ **do**
    **if** $z \notin \mathcal{Z}_j$ **then**
        $(w^z_{1,j}, \ldots, w^z_{d,j}, w^z_{\perp,j}) \leftarrow \mathbf{0}$
    **else**
        $\boldsymbol{v} \leftarrow \left(\frac{q_1(z)}{C_1}, \ldots, \frac{q_d(z)}{C_d}\right) \in \mathbb{R}^d$
        $\boldsymbol{u} \leftarrow \frac{\boldsymbol{v}}{\max\{1, \|\boldsymbol{v}\|_1\}} \cdot \frac{\Gamma}{C}$
        $(w^z_{1,j}, \ldots, w^z_{d,j}) \leftarrow (\lfloor u_1 \rceil, \ldots, \lfloor u_d \rceil)$
        $w^z_{\perp,j} = \lfloor \frac{\Gamma}{C} \rceil - \sum_{\ell=1}^{d} w^z_{\ell,j}$
**return** $w^z$
```