Preserving Target Distributions With Differentially Private Count Mechanisms

Nitin Kohli
UC Berkeley Center for Effective Global Action
nitin.kohli@berkeley.edu

Paul Laskowski
UC Berkeley School of Information
paul@ischool.berkeley.edu

Abstract

Differentially private mechanisms are increasingly used to publish tables of counts, where each entry represents the number of individuals belonging to a particular category. A *distribution of counts* summarizes the information in the count column, unlinking counts from categories. This object is useful for answering a class of research questions, but it is subject to statistical biases when counts are privatized with standard mechanisms. This motivates a novel design criterion we term *accuracy of distribution*.

This study formalizes a two-stage framework for privatizing tables of counts that balances accuracy of distribution with two standard criteria of accuracy of counts and runtime. In the first stage, a *distribution privatizer* generates an estimate for the true distribution of counts. We introduce a new mechanism, called the cyclic Laplace, specifically tailored to distributions of counts, that outperforms existing general-purpose differentially private histogram mechanisms. In the second stage, a *constructor algorithm* generates a count mechanism, represented as a transition matrix, whose fixed-point is the privatized distribution of counts. We develop a mathematical theory that describes such transition matrices in terms of simple building blocks we call ϵ-scales. This theory informs the design of a new constructor algorithm that generates transition matrices with favorable properties more efficiently than standard optimization algorithms. We explore the practicality of our framework with a set of experiments, highlighting situations in which a fixed-point method provides a favorable tradeoff among performance criteria.

Keywords

Differential Privacy, Distribution Preservation, Optimal Count Mechanisms, Convex Polytopes, Fixed-Point Analysis

1 Introduction

When studying populations, data is often organized as a table of counts, where each row records the number of people belonging to a particular category. Common categories include geographic regions, schools, and companies, among others. We assume that there are N categories, numbered 0 through $N-1$. We assume that each category i has a corresponding count d_i. We also require that every individual belongs to no more than one category. An example of such a dataset is provided in Figure 1.A, with categories corresponding to U.S. states and counts recording the number of individuals who contracted bird flu.

A: Table of Counts

State	Cases
Alabama	0
...	...
California	38
Colorado	10
...	...
Wisconsin	1
Wyoming	0

B: Distribution of Counts

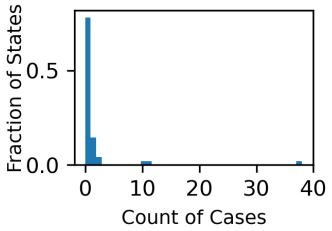

Figure 1: A. Table of counts of bird flu cases by U.S. state as of February 27, 2025. B. The corresponding distribution of counts. In this figure, $N = 50$ and $n = 41$.

For organizations that hold such data – including census bureaus [1], humanitarian groups [31], and public health departments [42] – there is a tension between releasing the data to facilitate wider research and analysis, and protecting the privacy of the individuals contained inside. In recent years, some organizations have started to publish *tables of privatized counts* in place of raw data. Here, we are referring specifically to tables that preserve the row structure of the original data; these are tables that contain an approximate count for every category in the original dataset. In a typical scenario, independent noise is added to each count, with the noise magnitude tailored to meet the common standard of differential privacy [14].

We can distinguish two broad classes of questions that can be answered using a table of counts. The first are questions that must be formulated using the name of at least one category. For example, a researcher might want to know how many bird flu cases have occurred in Colorado. We refer to these as *category-based questions*. In contrast, some questions can be written without referencing any category names. Examples include the following policy-relevant questions.

- What fraction of US flights have zero air marshals onboard? This number raises security concerns, and in particular was cited [25] in motivating a 2024 congressional bill [9].
- What fraction of company boards have more than 2 women? This threshold has been argued as important for changing company culture [32] and tracked in a recent report [38].
- What is the average number of students overseen by each guidance counselor in the U.S.? A 2025 study estimated the number to be 405, far exceeding the 250 recommended by the American School Counselor Association [36].

We refer to questions like these as *count distribution questions*, as it is possible to compute them after first preprocessing the count column. We assume each count d_i is in the set $\{0, 1, ...n-1\}$, with higher counts truncated if necessary. We assume that the upper

Proceedings on Privacy Enhancing Technologies 2026(2), 615–641

https://doi.org/10.56553/popets-2026-0063

bound $n-1$ is given publicly; if not, some privacy budget may be reserved to set n (for example, n can be set to a privatized 99^{th} percentile).

We first define the *histogram of counts* as a vector $\eta = (\eta_0, ..., \eta_{n-1})$, where $\eta_i = |\{j \in \{0, ..., N-1\} : d_j = i\}|$ is the number of categories with count i. For example, in Figure 1 η_0 equals the number of states with no bird flu cases. Note that $N = \sum_{i=0}^{n-1} \eta_i$. We then define the *distribution of counts* as a vector $\zeta = (\zeta_0, ..., \zeta_{n-1})$ where $\zeta_i = \eta_i / N$ denotes the proportion of categories with count i. Figure 1.B depicts the distribution of counts for the count table in Figure 1.A.

A problem arises when existing differential privacy techniques are used to answer count distribution questions. Within the literature, utility is most often captured via the typical error for a single count. Thus, canonical mechanisms — including the Laplace [14], geometric [21], Gaussian [13], and discrete Gaussian [8] — are not designed to preserve the distribution of counts. In general, each bin of the distribution of counts will have non-zero bias. For example, the geometric mechanism with outputs bounded to be non-negative tends to inflate the number of zeros in the count table.

Given the existence of such biases, one possible solution would be to independently create two separate data products: a table of privatized counts and a privatized distribution of counts (with the privacy budget split between the two). However, without additional constraints, there would be no guarantee of consistency between the two data products. As Long et al. explain, "Stakeholders [...] have expressed a strong desire for consistent data [...]. Such consistency has historically formed the bedrock of statistical use of the data, making it seamless to combine data across tables and easy to integrate it into traditional models and analyses" [28].

For the reasons above, we believe there is interest in a unified data product: a table of privatized counts that provides accurate answers to both category-based and count-distribution questions.

1.1 Distribution-Preserving Count Mechanisms

Conceptually, a *count mechanism* takes a "true" input count and outputs a (randomized) output count. Mathematically, it is useful to represent such a mechanism as an $n \times n$ transition matrix; this is a matrix $T \in \mathbb{R}_{\geq 0}^{n \times n}$ with $T\mathbb{1}_n = \mathbb{1}_n$, where $\mathbb{1}_n$ denotes the n-dimensional column vector of all ones. In alignment with the task of counting, all vectors and matrices will be indexed starting from 0. We will use capital letters for matrices and lowercase letters for their elements. For count mechanism T, $t_{i,j}$ represents the conditional probability that the output count is j given that the input count is i.

When the input count to a count mechanism T is drawn from a distribution z, written as an n-dimensional row vector, the distribution of the output count may be expressed as zT. We say that a count mechanism T has *fixed point* z if $zT = z$.

For the moment, assume that the true distribution of counts ζ is publicly known (we will relax this assumption in Section 1.2). We envision the following: The data holder selects count mechanism T with fixed point ζ. They then take each row i of the dataset, passes the count d_i as an input to T, and records the output in row i of the privatized table. The resulting output may be summarized with its distribution of privatized counts, which we label $\hat{z}$.

Because of the fixed-point constraint, for every output count $j \in \{0, ..., n-1\}$, $\mathbb{E}[\hat{z}_j] = \zeta_j$, so the output distribution of counts is correct in expectation.[1] While this is a promising observation, one obstacle remains to using a fixed-point constraint in this way: distribution ζ is often not publicly known. We next develop a workflow for situations in which ζ cannot be directly used.

1.2 A Two-Stage Counting Framework

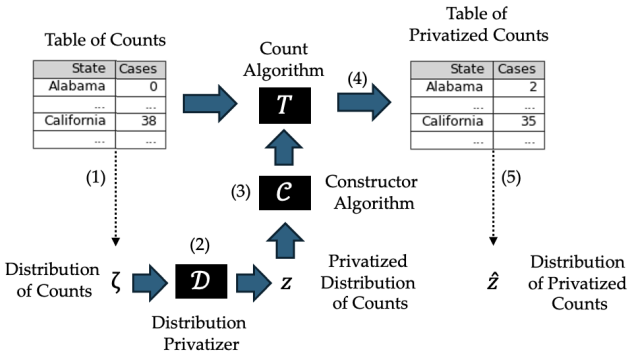

Figure 2: A framework for constructing and utilizing count mechanisms when the true distribution must be protected.

When choosing a count mechanism, a common obstacle is that the true distribution ζ is not public knowledge and must be kept secret by the data holder. To overcome this obstacle, we follow the strategy of adding a preprocessing step to generate a privatized approximation to ζ. The total privacy budget, which we label ϵ_t, must therefore be divided as $\epsilon_t = \epsilon_1 + \epsilon_2$, with ϵ_1 used to privatize ζ, and ϵ_2 reserved for the count mechanism.

When the distribution is privatized before a count mechanism is chosen, we will refer to the overall system as a *two-stage counting framework*. Figure 2 lays out the operation of such a framework, which involves five steps.

(1) The table of counts is summarized by its distribution of counts ζ.
(2) A distribution privatizer $\mathcal{D}$ is used to privatize ζ, using privacy budget ϵ_1, forming privatized distribution z (the *target distribution*).
(3) Constructor algorithm C is used to create an ϵ_2-differentially private count mechanism T. In normal operation, we require that T has fixed point z.
(4) Each row of the table of counts is passed into count mechanism T, one at a time, with the output recorded in a table of privatized counts using privacy budget ϵ_2.
(5) Users with count distribution questions may summarize the table of privatized counts with its distribution of privatized counts $\hat{z}$.

To fully implement this framework, one must specify a distribution privatizer $\mathcal{D}$, a constructor algorithm C, and several parameter

[1] To see why, let $\mathcal{I}_{i,j}$ be the indicator random variable corresponding to the event that row i of the count table (with count d_i) is mapped to j under T. Then $\mathcal{I}_{i,j} \sim \text{Bernoulli}(t_{d_i,j})$ and $\hat{z}_j = \frac{1}{N}\sum_{i=0}^{N-1} \mathcal{I}_{i,j} = \frac{1}{N}\sum_{l=0}^{n-1}\sum_{i:\, d_i = l} \mathcal{I}_{i,j}$ The inner sum is the sum of η_l Bernoulli random variables, each with expectation $t_{l,j}$, so $\mathbb{E}[\hat{z}_j] = \frac{1}{N}\sum_{l=0}^{n-1} \eta_l t_{l,j} = \sum_{l=0}^{n-1} \zeta_l t_{l,j} = (\zeta T)_j = \zeta_j$ where the last equality follows by the fixed-point constraint. Also $\text{Var}(\hat{z}_j) = \frac{1}{N}\left(\zeta_j - \sum_{l=0}^{n-1} \zeta_l t_{l,j}^2\right)$ so we expect the sampling error of the output distribution to be small for larger datasets.

values. To guide the choices that go into a deployment, we first identify a set of three performance criteria. These are intended to represent the top concerns of data users and data holders.

(1) **Accuracy of Distribution.** We conceptualize *distribution error* as a difference between ζ and $\hat{z}$. Although our framework involves a fixed-point constraint, there are two reasons that these distributions will not be identical. First, the fixed-point of T is z, which is an approximation to ζ. Second, there will be sampling error as each row of the table is selected randomly using probabilities in T. Distribution error can be operationalized using Wasserstein, total variation, or Kolmogorov-Smirnov distance. A researcher with a count distribution question may be primarily concerned with accuracy of distribution.
(2) **Accuracy of Counts.** We conceptualize *count error* as a typical distance between a single input count and the corresponding output count. This is commonly operationalized as mean squared error or expected absolute deviation. A researcher with a category-based question may primarily be concerned with accuracy of counts.
(3) **Runtime.** In many scenarios, the space of count mechanisms has high dimensionality and care must be taken that the constructor algorithm C can run efficiently. As we will see in Section 6, standard algorithms from linear programming are often impractical in our setting.

There are obstacles to designing a system that meets all three performance criteria to a satisfactory level. First, one might hope to adapt existing algorithms to play the role of the distribution privatizer $\mathcal{D}$ and the constructor algorithm C. While this works for some datasets, as we explore in Section 6, existing algorithms often lead to reduced accuracy or impractically long runtimes. Moreover, the development of new constructor algorithms is complicated by a lack of mathematical theory describing private count mechanisms with fixed-point constraints.

1.3 Summary of Contributions

Our study pioneers the development of count mechanisms that incorporate a fixed-point constraint to provide high accuracy of distribution. We make four main contributions: the first three are mathematical, corresponding to different portions of the two-stage counting framework; the last contribution is experimental, and explores the practicality of our approach. Table 1 provides a table of notation used throughout the paper.

Distribution Privatizer. We provide a novel mechanism to privatize a distribution of counts with a differential privacy guarantee (Theorem 1). Our mechanism, called the *cyclic Laplace mechanism for distributions of counts* (Definition 2), employs Laplace noise, but transfers probability between adjacent positions in the distribution of counts, which is specifically tailored to mask the presence or absence of an individual in this context. We provide evidence that this technique yields more accurate distributions than existing general-purpose differentially private histogram mechanisms.

Theory of Count Mechanisms with Fixed Points. We provide a compact description of count mechanisms that satisfy differential privacy and a fixed-point constraint using an atomic element that we call an ϵ-*scale* (Definition 6). Roughly speaking, an ϵ-scale is a probability distribution over $\{0, 1, 2, ..., n-1\}$ in which a differential privacy constraint binds for every pair of neighboring positions. The name comes from the notion that, if one were to take a "walk" along a scale, every step from one position to the next would either be a step up or step down by the same multiplicative constant. Our results will show that ϵ-scales are fundamental building blocks of differentially private count mechanisms.

Notation	Description
n	Maximum allowed count is $n-1$
N	Number of categories
η	Histogram of counts
ζ	Distribution of counts
$\mathcal{D}$	Distribution privatizer with budget ϵ_1
C	Constructor algorithm with budget ϵ_2
z	Fixed point, output of $\mathcal{D}$
T	Count algorithm, output of C
$\mathbb{1}_n$	n-dimensional vector of 1's
F	Fixed-point polytope for z
U	Unfixed point polytope
R_F	Representation Fixed-point polytope of z
R_U	Representation Unfixed point polytope
M_F	Movement matrices for R_F
M_U	Movement matrices for R_U
s	ϵ-scale
p	pattern of an ϵ-scale
k	Number of ϵ-scales (2^{n-1})
Ψ	Scale matrix
$ex(\cdot)$	Extreme points of a polytope
$\langle \cdot, \cdot \rangle$	Frobenius inner product
κ	Column selector function

Table 1: Frequently used notation.

We begin by formally defining $F_\epsilon[z]$ as the set of count mechanisms that satisfy ϵ-differential privacy with fixed-point z (Definition 5); we often just write F for brevity when ϵ and z are clear from context. Viewed geometrically, F is a convex polytope, and hence it can be characterized using its extreme points.

We show that any differentially private fixed-point count mechanism in F can be described using a conic combination of ϵ-scales (Corollary 1). Building on this observation, we construct a *representation polytope* R_F that encodes how much of each scale contributes to each column of a mechanism T (Definition 7). This polytope is related to F by a linear map Ψ whose matrix representation is comprised of the ϵ-scales. In Theorem 3, we prove that any mechanism in F has a representation (not necessarily unique) in R_F. Furthermore, Proposition 1 links each extreme point of F to at least one extreme point of R_F through Ψ.

These results suggest that we can learn about the extreme points of F by studying the extreme points of R_F. In Theorem 11, we characterize these extreme points in terms of the scales that make them up. Specifically, the extreme points of R_F are exactly those for which certain weighted differences of scales are linearly independent – a condition we call Ψ-*affinely simplified* (Definition 8).

Taken together, our results show that every extreme matrix of F can be represented by a relatively small number of ϵ-scales (Corollary 2), fulfilling certain conditions (Theorem 11). The algebraic structure that we discover provides a framework for reasoning

about differentially private fixed-point count mechanisms, which can be leveraged for algorithmic design.

Constructor Algorithm. Next, we consider the design of a constructor algorithm. Our primary focus is on the development of constructor algorithms whose output has fixed-point z, which we refer to as *fixed-point constructor algorithms*. Since all such constructors share the same fixed-point constraint, we expect them to behave similarly in terms of accuracy of distribution (we will later confirm this in experiments). We therefore follow an approach in which we focus on accuracy of counts and use C to find a count mechanism with low count error. It is sometimes possible to minimize count error exactly using standard linear programming algorithms; however, the runtime of existing solvers can be prohibitively long for large n. For such settings, we provide Algorithm 1, which leverages our scale representation and builds a conic combination of scales in a greedy fashion to output an extreme point of $F_{\epsilon_2}[z]$ (Theorem 6). Our algorithm runs as fast as $O(n^2)$ (Theorem 5), demonstrating its practicality even when n is large.

Our mathematical framework can also be utilized to design constructor algorithms that do not involve a fixed-point constraint. When used in a two-stage framework, this forms a relevant baseline against which fixed-point methods can be compared. We let U denote the set of ϵ-differentially private count mechanisms that may send z to a different distribution (Definition 4). We leverage our framework and a key result from Ghosh et al. [21] to create a *two-staged unfixed optimum constructor* (Algorithm 2), which returns the lowest count-error mechanism in U in $O(n^2)$ time (Theorem 7).

Empirical Results and Practical Guidance. In Section 6, we report on a set of experiments in which our fixed-points methods are used to generate tables of privatized counts. We are interested in how different fixed-point constructors perform relative to each other, and also how they perform relative to four *unfixed baselines* – mechanisms representing prior work that do not enforce a fixed-point constraint. We divide these questions into three components, aligned with our three performance criteria: accuracy of distribution, accuracy of counts, and runtime.

First, we find that the inclusion of a fixed-point constraint usually yields a significant improvement to accuracy of distribution, relative to all unfixed baselines. This is to be expected, as the unfixed baselines do not encode a preference for accuracy of distribution. Next, we find that that switching from an unfixed baseline to a fixed-point method generally degrades accuracy of counts. This is also to be expected, as fixed-point methods must output a count mechanism in polytope F, which is a strict subset of the unfixed polytope U. Nevertheless, we find the difference to be moderate in magnitude – sometimes a matter of a few percentage points. Finally, we find that the fastest runtime is always attained by one of the unfixed baselines. Fixed-point constructors that minimize count error exactly (i.e., using the interior point method or simplex algorithm) exhibit large growth rates and are impractical for all but the smallest values of n; in these situations, our heuristic constructor provides a promising alternative, with execution time that is experimentally comparable to the fastest unfixed baseline.

Taken together, our experimental results shed light on the practical tradeoffs from adopting a fixed-point constraint. Throughout Section 6, we highlight scenarios where a fixed-point constraint provides substantial improvements in terms of accuracy of distribution, with modest performance losses in both accuracy of counts and runtime.

Organization of Paper: In Section 2, we discuss related works. In Section 3, we propose a distribution privatizer that is ϵ-differentially private and argue that it has favorable error characteristics. In Section 4 we develop a mathematical framework that describes mechanisms in F using ϵ-scales, and provide conditions that characterize all extreme points. In Section 5, we discuss constructor algorithms, including novel algorithms that leverage ϵ-scales to run efficiently. In Section 6, we explore the practicality of our approach with a set of experiments. We conclude with a discussion of future avenues of inquiry in Section 7. For readability, all proofs of the mathematical results described in the main text are deferred to the Appendix.

2 Related Works

Differential privacy was introduced in 2006 by Dwork, McSherry, Nissim, and Smith [14]. One of the earliest and most studied problems in the field is that of privately computing a count of individuals [14]. The modern toolkit for privatizing counts includes the Laplace [14] and Gaussian mechanisms [13], and their discrete analogs (the geometric [17, 21] and discrete Gaussian mechanisms [8]), sometimes followed by a postprocessing step to ensure that desirable data properties are met [6, 7, 10].

A number of authors present methods for privatizing distributions, which can serve as a distribution privatizer in our two-stage framework. Typically in this literature, the outputs corresponding to bins are correlated, in order to improve the utility for certain sets of queries. For example, Xiao et al. [44] apply wavelet transformations to privatize ordinal data while maintaining high accuracy for range count queries. Hay et al. [24] introduce hierarchical methods to improve the accuracy of range queries for differentially private histograms. Qardaji et al. [39] improve upon the hierarchical method of Hay et al. by specifying an error measure and recommending optimal branching factors. Like the above studies, our cyclic Laplace mechanism also utilizes noise that is shared between histogram bins, but our sharing strategy is tailored specifically to the setting of distributions of counts. Sharing a focus on privatized distributions, Li et al. [35] propose a mechanism that adds noise to individual datapoints before reconstructing a distribution. Their noise is drawn from a square wave to meet the standard of local differential privacy, a stricter standard than the notion of (central) differential privacy considered in this paper.

Several studies address privatizing counts, and are especially relevant to the constructor algorithm of our two-stage framework. In a series of papers, Geng et al. show that the staircase mechanism is the differentially private algorithm that minimizes a certain class of cost functions with real-valued outputs [18–20]. Li et al. introduce the matrix mechanism to privately answer a collection of count-based queries using (potentially correlated) Laplace and Gaussian noise to minimize the total error [33]. Sadeghi et al. utilize techniques from linear programming to develop a differentially private count mechanism that adds integer-valued noise over a fixed and finite range, and numerically demonstrate its improved privacy properties over the discrete Gaussian mechanism [40].

In the study closest to ours, Ghosh et al. study the problem of minimizing count error for count mechanisms without a fixed-point constraint [21]. Under mild assumptions on an objective function, they find that all optimal mechanisms are equivalent to a geometric mechanism under post-processing – a result that we leverage in developing our two-staged unfixed optimum constructor. Unlike Ghosh et al., we provide a linear algebra framework that provides new intuition for the structure of count mechanisms, and we focus on a scenario with a fixed-point constraint.

Kairouz et al. [30] and Holohan et al. [27] both mathematically analyze the behavior of optimal algorithms satisfying local differential privacy [16, 46]. As in our setting, the authors represent mechanisms as transition matrices and apply linear constraints corresponding to a differential privacy guarantee. In contrast to the task of counting, the authors allow neighboring datasets to be associated with any pair of inputs, not just consecutive rows. This leads to the local differential privacy polytope. We study different polytopes in this paper, but our focus on characterizing extreme points is shared by the above authors.

The mathematical techniques we use to analyze our polytopes of interest, U and F, are rooted in the algebraic study of matrix polytopes more broadly. The classic Birkhoff–von Neumann Theorem states that permutation matrices are the extreme points of the set of doubly stochastic matrices [12]. Jurkat and Ryser [29] generalize this result to the transportation polytope with row and column sums equal to $\varphi \in \mathbb{R}^{1\times n}$. Hartfiel [22] characterizes the extreme points of the set of stochastic matrices with a fixed point z. Since we incorporate privacy constraints into the definition of our polytopes, our results can be viewed as extensions of the above results, though our algebraic tools are somewhat different.

3 Privatizing Distributions of Counts

In this section, we consider the design of a distribution privatizer $\mathcal{D}$, which forms part of our two-stage counting framework. The role of this component is to take the secret distribution of counts ζ and return an approximation meeting ϵ-differential privacy. One existing candidate algorithm is the n-dimensional Laplace mechanism, which adds independent Laplace noise[2] to each dimension proportional to L_1 global sensitivity [14]. Compared to a generic probability distribution however, a distribution of counts brings additional structure that the classic Laplace mechanism does not leverage. We therefore provide a design for an alternative algorithm.

A *randomized algorithm* $\mathcal{R}$ accepts an input and outputs a random variable that takes values in some set $\mathbb{O}$. Differential privacy depends on the concept of *neighboring inputs* [14]. We adopt the presence-absence variant of neighboring, which means that two inputs are neighbors if one input corresponds to adding or removing exactly one individual from the other. In the context of count tables, this means that two count tables d and d' are neighboring if and only if there exists a row $i \in \{0, ..., N-1\}$ such that $|d_i - d'_i| = 1$ and for all $j \neq i$, $d_j = d'_j$.

Definition 1. A distribution privatizer satisfies ϵ-differential privacy if for every pair of neighboring count tables d and d' and for every measurable set $\Theta \subseteq \mathbb{O}$, $\mathbb{P}(\mathcal{R}(d) \in \Theta) \leq e^{\epsilon}\mathbb{P}(\mathcal{R}(d') \in \Theta)$.

[2]The Laplace(b) distribution is the probability distribution over $\mathbb{R}$ whose density function is $\varphi(x; b) = \frac{1}{2b}\exp(-|x|/b)$, expectation is 0, and variance is $2b^2$.

Throughout this study, we assume that $\epsilon > 0$. When a distribution privatizer is used in the two-stage counting framework, we use the value of $\epsilon = \epsilon_1$.

Definition 2. We define the *cyclic Laplace mechanism for distributions of counts* as a randomized algorithm that accepts a parameter ϵ and table of counts d as inputs. It then computes its distribution of counts ζ and independently samples $L_0, ..., L_{n-1} \sim$ Laplace$(1/(N\epsilon))$, defines $L_n = L_0$, and outputs $V(\zeta) = (V_0, ..., V_{n-1})$ where $V_i = \zeta_i + L_i - L_{i+1}$ for all $i \in \{0, ..., n-1\}$.

Observation 1. *By construction,* $\sum_{i=0}^{n-1} V_i = \sum_{i=0}^{n-1} \zeta_i = 1$. *However, there is no guarantee that* $V(\zeta)$ *is a probability distribution because some elements may be negative, so post-processing may be required.*

To provide some intuition for this mechanism, consider our earlier example in Figure 1. A switch to a neighboring dataset could represent an extra individual in Alabama contracting bird flu, increasing the count for Alabama from 0 to 1, while all other rows remain unchanged. As a result, there would then be one extra 1 in the column of counts, and one fewer 0. The histogram of counts therefore changes only in positions 0 and 1, and only by 1 in these positions. In general, switching to a neighboring dataset always has the effect of shifting one unit from one histogram position to an adjacent position. In the cyclic Laplace mechanism for distributions of counts, such a change can be "masked" if the Laplace random variable corresponding to the pair of positions changes its value by one. This intuition is formalized in the proof of Theorem 1.

Theorem 1. *The cyclic Laplace mechanism for distributions of counts satisfies* ϵ*-differential privacy.*

A benefit of the cyclic Laplace mechanism for distributions of counts is that it does not alter the cumulative sum of the distribution of counts by very much. For index $i \in \{0, ..., N-1\}$, the variance of the cumulative sum of the output of the cyclic Laplace mechanisms for distributions of counts is given by,

$$\mathrm{Var}\left[\sum_{j\leq i} V_j\right] = \mathrm{Var}\left[\sum_{j\leq i} \zeta_j + L_j - L_{j+1}\right] = \mathrm{Var}\left[L_0 - L_{j+1}\right] = \frac{4}{N^2\epsilon^2}$$

This result can be compared to the classic Laplace mechanism, which would output $\tilde{V}(\zeta) = (\tilde{V}_0, ..., \tilde{V}_{n-1})$ where $\tilde{V}_j = \zeta_j + \tilde{L}_j$ for independent Laplace random variables $\tilde{L}_j \sim$ Laplace$(2/(N\epsilon))$ (the L_1 sensitivity is $2/N$ since adding an individual increases η_i for one i and reduces $\eta_{i'}$ for an adjacent i'). The resulting variance is

$$\mathrm{Var}\left[\sum_{j\leq i} \tilde{V}_j\right] = \mathrm{Var}\left[\sum_{j\leq i} \zeta_j + \tilde{L}_j\right] = \mathrm{Var}\left[\sum_{j\leq i} \tilde{L}_j\right] = (i+1)\frac{8}{N^2\epsilon^2}$$

which is larger by a factor of $4(i+1)$, meaning that the cumulative sum gets progressively less precise from position 0 to position $n-1$.

Note that the sums in the variances above are not technically cumulative distributions, because V and $\tilde{V}$ may not be valid probability distributions. To use either of these mechanisms in the two-stage counting framework, it is necessary to post-process the output so it is a valid probability distribution. Unfortunately, the simple expressions for variance above may not hold after post-processing.

We perform experiments to measure the distributional accuracy of the cyclic Laplace in practice. As points of comparison, we consider three mechanisms: the classic Laplace, the Privelet mechanism

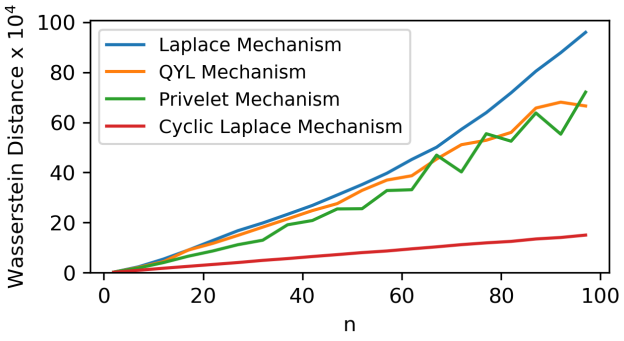

Figure 3: Comparison of distribution error for the cyclic Laplace mechanism against other mechanisms with $\epsilon_1 = 1$.

with the Haar wavelet transform of Xiao et al. [44], and the Hierarchical mechanism with weighted averaging, mean consistency, and a branching factor of 16 as recommended by Qardaji et al. [39] (hereafter the QYL mechanism). The latter two mechanisms are designed to improve accuracy when answering range queries.

We use a dataset describing schools which will be introduced later in Section 6, top coding the counts at $n-1$ for different values of n. We operationalize distribution error using Wasserstein-1 and Kolmogorov-Smirnov (KS) distance. Figure 3 shows the results of this study for Wasserstein distance; the results for KS distance are similar. The cyclic Laplace mechanism outperforms the other mechanisms for all values of n.

More generally, this cyclic approach can be combined with other differentially private mechanisms as building blocks, such as the Gaussian mechanism (see Appendix A for details).

4 Describing Fixed-Point Count Mechanisms

In this section, we provide a compact description of count mechanisms that satisfy differential privacy and the fixed-point constraint. A key aim of this effort is to provide a framework to assist in designing constructor algorithms for the two-stage counting framework.

4.1 Mathematical Formulation

Recall that from Section 1.1 that a count mechanism is represented by an $n \times n$ transition matrix T. To incorporate differential privacy into this setting, we compare the conditional probability of an output $j \in \{0, ..., n-1\}$ based on the presence or absence of an individual; in the count setting, this requires us to compare an input of i against an input of $i+1$, for $i \in \{0, ..., n-2\}$. This is expressed for a single column as follows.

Definition 3. A column vector $v \in \mathbb{R}^n$ is *ϵ-neighbor indistinguishable* if $e^{-\epsilon} v_{i+1} \leq v_i \leq e^{\epsilon} v_{i+1}$ for all $0 \leq i \leq n-2$.

Since $\epsilon > 0$ throughout our study, it follows from the definition that all entries of an ϵ-neighbor indistinguishable vector are non-negative. Denote the j^{th} column of a matrix T as T_j. We can now define differential privacy over count mechanisms as follows.[3]

Definition 4. A count mechanism T satisfies *ϵ-differential privacy* if and only if every column T_j is ϵ-neighbor indistinguishable. Let $U_\epsilon \subset \mathbb{R}^{n \times n}$ denote the set of all *ϵ-differentially private count mechanisms*. Since we have placed no fixed-point restrictions on U_ϵ, we will additionally refer to it as *unfixed*. When ϵ is clear from context, we will denote this set as U.

In the context of the two-stage counting framework, we require that T satisfies ϵ_2-differential privacy.

The privacy constraints above can be understood as a set of linear constraints on a vector space. To make this explicit, for $i \in \{0, ..., n-2\}$, we define π_i^+ to be the n-dimensional row vector with position i equal to 1, position $i+1$ equal to $-e^{\epsilon}$, and all other positions are 0; similarly, π_i^- is the n dimensional row vector with position i equal to -1, position $i+1$ equal to $e^{-\epsilon}$, and all other positions are 0. Let $\Pi \in \mathbb{R}^{(2n-2) \times n}$ be the matrix whose rows are given by the $2n-2$ differential privacy constraints $\pi_0^-, \pi_0^+, ..., \pi_{n-2}^-, \pi_{n-2}^+$. Given this notation, a column vector v is ϵ-neighbor indistinguishable if and only if $\Pi v \preceq 0_{2n-2}$, where $\preceq$ denotes the element-wise inequality $\leq$. Note that a constraint of this form defines a convex cone in $\mathbb{R}^n$. This yields the following standard result from geometry.

Observation 2. *Any conic combination of ϵ-neighbor indistinguishable vectors is also ϵ-neighbor indistinguishable.*

Similarly, a count mechanism T is ϵ-differentially private if and only if $\Pi T \preceq 0_{(2n-2) \times n}$. Next, we restrict the set of such mechanisms by adding a fixed-point constraint.

Definition 5. Let $F_\epsilon[z] \subset U_\epsilon$ denote the set of all ϵ-differentially private count mechanisms with fixed point z. When ϵ and z are clear from context, we omit them and write F instead.

We follow the convention of Holohan et al. and refer to a *convex polyhedron* as the intersection of finitely many closed half-spaces, and refer to a bounded convex polyhedron as a *convex polytope* [27]. We have defined both U and F as a finite set of linear constraints on $\mathbb{R}^{n \times n}$. Moreover, both sets are bounded because each element of a count mechanism T must be between 0 and 1. Hence U and F are both convex polytopes.

A fact from geometry is that any convex polytope can be characterized by its extreme points. We recall that A is an *extreme point* of polytope P if A cannot be written as a non-trivial convex combination of elements of P. That is, A cannot be written as $A = \alpha X + (1-\alpha)Y$ for some distinct $X, Y \in P$ and $\alpha \in (0,1)$. We denote the set of extreme points of a convex polytope P as $ex(P)$. We therefore view $ex(U)$ and $ex(F)$ as representatives of desirable classes of count mechanisms. We now aim to describe them in a way that builds intuition.

4.2 ϵ-Scales as Building Blocks

We now introduce the important mathematical object we call a scale. This is a probability vector in which a differential privacy constraint binds for every pair of consecutive entries. Let Δ_n denote the set of probability column vectors of length n, $\Delta_n = \{v \in \mathbb{R}^n_{\geq 0} : v^\top \mathbb{1}_n = 1\}$.

[3] We have chosen to define differential privacy over distribution privatizers and count mechanisms separately because they accept different inputs (the table of counts versus a single count). While we have defined these separately, they are actually connected as follows. We say two non-negative integers x, y are *neighboring counts* if $|x-y| = 1$. Then, we two count tables d and d' are neighboring if and only if $d_i = d'_i$ for $N-1$ rows $i \in \{0, ..., N-1\}$ and there exists a unique row $j \in \{0, ..., N-1\}$ such that d_j and d'_j are neighboring counts. These definitions ensure, by parallel composition, that if a count mechanism satisfies ϵ-differential privacy, the independent application of the mechanism to every row in a count table also satisfies ϵ-differential privacy.

Definition 6. We define an ϵ-*scale* as a probability vector $s \in \Delta_n$ such that for all $i \in \{0, ..., n-2\}$, either $s_i = e^{\epsilon} s_{i+1}$ or $s_i = e^{-\epsilon} s_{i+1}$.

Using our algebraic notation, we can equivalently define an ϵ-scale as a vector $s \in \mathbb{R}^n$ such that $\mathbb{1}_n^\top s = 1$ and either $\pi_i^+ s = 0$ or $\pi_i^- s = 0$ for all $i \in \{0, ..., n-2\}$.

An elementary counting argument shows that there are $k = 2^{n-1}$ ϵ-scales. For the remainder of this paper, we fix a matrix $\Psi \in \mathbb{R}_{\geq 0}^{n \times k}$ whose columns are the k ϵ-scales in any arbitrary order. In our notation, the j^{th} scale is denoted Ψ_j, for $j = 0, ..., k-1$.

Lemma 1 shows that scales can be viewed as building blocks of each column of any matrix in U.

Lemma 1. *If $T \in U$, then every column of T can be written as a conic combination of ϵ-scales.*

Since $F \subset U$, every column of matrices in F can be written as conic combinations of scales as well.

Corollary 1. *If $T \in F$, then every column of T can be written as a conic combination of ϵ-scales.*

Hinting at the utility of scales, we begin by highlighting a result that characterizes certain extreme points of U in terms of ϵ-scales.

Theorem 2. *For $T \in U$ with all positive entries, $T \in ex(U)$ if and only if the columns of T are linearly independent multiples of ϵ-scales.*

Given this result, one might wonder whether ϵ-scales can be leveraged to give a compact description of the extreme points of F. This turns out to be true; however, additional machinery is required, which we develop in the remainder of this section.

4.3 Representing F with ϵ-Scales

As we showed in the prior subsection, every matrix $T \in F$ can be formed from ϵ-scales. To proceed further, we require a more precise accounting of how much of each scale contributes to each column.

By Corollary 1, every column of $T \in F$ can be expressed as a conic combination of scales. This means that there exist non-negative $b_{i,j}$ such that $T_j = \sum_{i=0}^{k-1} b_{i,j} \Psi_i$. In other words, $b_{i,j}$ quantifies the amount of scale i that enters column j of T. Arranging these $b_{i,j}$ into a $k \times n$ matrix B, we have $T = \Psi B$. We refer to B as a *representation matrix*, and denote the set of all such B as R_F. A more mathematically precise definition is given below.

Definition 7. We define the *fixed point representation polytope*[4] R_F as

$$R_F = \left\{ B \in \mathbb{R}_{\geq 0}^{k \times n} : \Psi B \mathbb{1}_n = \mathbb{1}_n \text{ and } z\Psi B = z \right\}$$

For the remainder of this paper, we slightly overload notation and use Ψ to refer to both the matrix Ψ and the linear map formed by left multiplication by Ψ. The next theorem shows that Ψ relates any matrix in F to at least one matrix in R_F.

Theorem 3. *Ψ is an affine surjection from R_F to F.*

Thus, the surjection Ψ allows us to express a matrix $T \in F$ using a matrix B in the new space R_F where the neighbor-indistinguishability constraints within each column of T are replaced by linear constraints on the rows and columns of B. In Proposition 1, we show the linear map Ψ also provides a relation between the extreme points of F and the extreme points of R_F.

[4] R_F is a polytope, as it is bounded and formed from a finite number of linear constraints.

Proposition 1. *For all $T \in ex(F)$ there exists $B \in ex(R_F)$ such that $T = \Psi B$.*

Hence, every extreme point in F can be written as an extreme point in R_F under the mapping Ψ. However, this representation is not necessarily unique. In particular, it is possible for multiple extreme points of R_F to map to the same extreme point in F. Additionally, it is possible for an extreme point of R_F to map to a non-extreme point of F under Ψ. We demonstrate both phenomena when $n = 3$ in Example 1 of Appendix B.1. Additionally, in Corollary 3 of Appendix B.2, we provide a geometric condition on extreme points of R_F that indicates exactly when they are mapped to an extreme point of F.

4.4 Characterizing the Extreme Points of R_F

Up to this point, we have argued that differentially private count mechanisms with fixed point z are represented by the extreme points of F, and there is a relationship between these extreme points and those of the representation polytope R_F. Next, we characterize the extreme points of R_F in terms of the scales they encode.

Given a column vector $x \in \mathbb{R}^k$, let $H(x)$ be the set of vectors

$$h = (z\Psi_u)^{-1}\Psi_u - (z\Psi_v)^{-1}\Psi_v$$

where v is the smallest index such that $x_v > 0$ and u is another index such that $x_u > 0$.[5] Note that, under this definition, if a vector x has either 0 or 1 positive entry, $H(x) = \emptyset$. Also, for $h \in H(x)$, $zh = 0$ by routine computation.

Definition 8. We call a $k \times n$ matrix B Ψ*−affinely simplified* if the additive union[6] $\uplus_{j=0}^{n-1} H(B_j)$ is linearly independent.

Let $M_F(\Psi)$ be the set of $k \times n$ matrices μ such that $z\Psi\mu = 0_n^\top$ and $\Psi\mu\mathbb{1}_n = 0_n$. By construction of $M_F(\Psi)$, if $X, Y \in R_F$, then $X - Y \in M_F(\Psi)$. Conceptually, M_F represents "movement matrices" that can potentially be added to matrices in R_F to move through the affine space. The following lemma and theorem formalize the equivalence between the extreme points of R_F, the notion of Ψ-affinely simplified matrices, and these movement matrices.

Lemma 2. *Suppose $B \in R_F$. Then B is Ψ-affinely simplified if and only if there does not exist a non-zero matrix $\mu \in M_F(\Psi)$ such that $\mu_{i,j}$ is zero whenever $b_{i,j}$ is zero.*

Theorem 4. *Given scale matrix Ψ, $B \in ex(R_F)$ if and only if $B \in R_F$ is Ψ-affinely simplified.*

Hence, the extreme points of our representation space are those that are Ψ-affinely simplified. Additionally, Theorem 4 can be used to quantify combinatorial properties of $B \in ex(F)$, such as the number of positive entries.

Corollary 2. *Let $\mathcal{P}$ denote set of indices where z is positive. Then extreme matrices in R_F contain at least $|\mathcal{P}|$ and at most $|\mathcal{P}| + n - 1$ positive entries.*

Thus the extreme points of F can be represented using relatively few scales. This suggests that constructor algorithms C that work

[5] Note that $(z\Psi_j)^{-1}$ is well-defined for all $j \in \{0, ..., k-1\}$, since $z\Psi_j > 0$ because z is a probability row vector and Ψ_j is an ϵ-scale.

[6] The *additive union* of sets X and Y is the multiset denoted as $A \uplus B$, where the multiplicity of $s \in X \uplus Y$ equals the sum of the multiplicity of s in X and of s in Y [5].

at the granularity of scales could be used to build an extreme point of F efficiently. We showcase an example of this in the next section.

5 Constructor Algorithms

In this section, we turn our attention to the design of the constructor algorithm. This component of the two-stage counting framework outputs a single count mechanism from within a set of allowable count mechanisms. We first discuss fixed-point constructors before describing the two-staged unfixed optimum constructor.

5.1 Fixed-Point Constructors

The goal of a fixed-point constructor is to accept target distribution z and output a single mechanism inside $F_\epsilon[z]$. Recall from the introduction that we identified three performance criteria for a two-stage counting framework. The first criterion, accuracy of distribution, is supported by the inclusion of a fixed-point constraint. Moreover, in the experiments we describe later, we observe that all fixed-point constructors we create behave very similarly with respect to this criterion. We therefore set accuracy of distribution aside and focus the following discussion on the other criteria: accuracy of counts and runtime.

Beginning with accuracy of counts, we consider a natural class of count-error measures – those that are linear in the coefficients of the transition matrix. Given a count mechanism T, and an $n \times n$ matrix of weights W, we define the count error of T under W with the Frobenius inner product $\langle W, T\rangle = tr[W^\top T] = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} w_{i,j}t_{i,j}$. For example, $w_{ij} = z_i|i-j|$ yields expected absolute deviation:

$$\langle W, T\rangle = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} z_i|i-j|t_{i,j} = \mathbb{E}_{i\sim z}\mathbb{E}_{j\sim \text{Row } i \text{ of } T}\left[|i-j|\right]$$

Similarly, setting $w_{ij} = z_i(i-j)^2$ encodes mean squared error. Since $F_\epsilon[z]$ is a polytope, the optimum of such a function can be achieved at one of the extreme points $ex(F_\epsilon[z])$ (Lemma 4).

Given a linear count-error function, it is useful to divide the task of choosing a mechanism in F into two cases. First, when n is small, a mechanism that minimizes count error can be found using standard linear programming algorithms. These include open-source and commercial variants of the simplex algorithm and the interior point method. We refer to a constructor algorithm that minimizes count error over F as a *count-error-minimizing fixed-point constructor*.

For larger values of n, the runtime of linear programming algorithms can be prohibitively large. For situations like these, we provide a heuristic algorithm that outputs a single extreme point of $F_\epsilon[z]$. Our algorithm is inspired by our scale representation, but only requires polynomial space, and can run as fast as $O(n^2)$.

5.2 Heuristic Constructor Algorithm

To motivate our algorithm, we begin with a piece of intuition. To make the output count close to the input count, we want the probability near the diagonal to be high, and the probability far away from the diagonal to be low. This leads us to the notion of a *single-peaked scale at position j*, which increases to a single peak at row j, then decreases until the end. The following definitions make this notion precise.

Definition 9. A *pattern* is an $(n-1)$-tuple $p \in \{-1,+1\}^{n-1}$. We say an ϵ-scale s *has pattern* p if, for all $j \in \{0,...,n-2\}$, $p_j = 1$ if $s_j < s_{j+1}$, and $p_j = -1$ otherwise.

Definition 10. The *single-peaked pattern* at $j \in \{0,...,n-1\}$ is the pattern p such that $p_i = 1$ for all $i < j$ and $p_i = -1$ otherwise. We will refer to an ϵ-scale with a single-peaked pattern at j as a *single-peaked scale at position j*.

One might hope that a mechanism T can be constructed solely from singled-peaked scales. This is possible for $T \in U$, but is not possible for $T \in F$ in general given the fixed-point constraint. However, single-peaked scales can provide a useful starting point for constructing a count mechanism in F. This informs the design of Algorithm 1.

The core idea underlying our algorithm is as follows. Imagine starting with the $n \times n$ zero matrix A and then adding scales to it until the row sum and fixed-point constraints of F are fulfilled. By Observation 2, adding positive multiples of ϵ-scales always maintains ϵ-neighbor indistinguishability. As we add scales, we can keep track of how much still needs to be added to each row, $r = \mathbb{1}_n - A\mathbb{1}_n$, and how much still needs to be added to each column, $c = z - zA$. At the end of the algorithm, we must have $r = 0_n$ and $c = 0_n^\top$. If at any point when adding scales, an element of c becomes negative, it will not be possible to arrive at $c = 0_n^\top$ by adding more scales. Similarly, if at any point r becomes not ϵ-neighbor-indistinguishable, it will not be possible to add more scales to end with $r = 0_n$ (Proposition 2 quantifies how much of a scale s can be added without triggering one of these two conditions).

Applying this idea, Algorithm 1 proceeds iteratively, filling one column at a time. When operating on a particular column j, if it is possible to add some multiple of the single-peaked scale at position j while preserving the ϵ-neighbor-indistinguishability of r, we greedily add as much of the scale as possible. Otherwise, we adjust the single-peaked scale as necessary so that it matches the pattern of r at any row where a privacy bound holds for r. This guarantees that we are able to add some positive multiple of the scale to A. The details can be observed in the proof of Theorem 5. We repeat the process of adding scales to column j until the column is at capacity ($c_j = 0$).

When execution begins, and any time a column has been filled to capacity, the algorithm must select a new column j to operate on. The selection of a new column is delegated to a function κ, which is an input parameter. The only requirement for the results of this section is that κ always returns a column index j with $c_j > 0$. For example, κ could select the column j for which c_j is maximal and non-zero. In principle, one could choose many other options for κ, which may result in varying runtimes and have a significant effect on the accuracy of counts. We observe this phenomenon for three possible choices for κ in the experiments of Section 6.

The following two theorems demonstrate that Algorithm 1 can run efficiently and outputs an extreme point of F.

Theorem 5. *If the worst-case runtime of selection algorithm κ is $O(\tau)$, then the worst-case runtime of Algorithm 1 is $O\left(\max\{\tau n, n^2\}\right)$.*

Theorem 6. *Algorithm 1 outputs a matrix in $ex(F)$.*

Selection Guidelines for κ in Practice. The reader might wonder how to choose a column selector in order to have the lowest count

Algorithm 1: Heuristic Constructor Algorithm

Input:

- Fixed point z
- Privacy parameter ϵ
- Column selection function κ

Initialize an $n \times n$ matrix A of 0's, $r = \mathbb{1}_n$, and $c = z$
while $r \neq 0_n$ *(equivalently,* $c \neq 0_n^\top$*)* **do**
  Select a column $j = \kappa(z, r, c, A)$ such that $c_j > 0$.
  while $c_j > 0$ **do**
    Assign p as the single-peaked pattern for j.
    Adjust p as follows: for any i which $\frac{r_{i+1}}{r_i} = \exp(-p_i\epsilon)$, reassign $p_i \leftarrow -p_i$.
    Let s be the ϵ-scale with pattern p.
    Compute $q = \max\{\gamma : \gamma zs \leq c_j$ and $r - \gamma s$ is ϵ-neighbor indistinguishable$\}$.
    Update $A_j \leftarrow A_j + qs$, $r \leftarrow r - qs$, and $c_j \leftarrow c_j - qzs$.
  end
end
Return A

error for a given dataset. Surprisingly, once a data holder generates the privatized distribution of counts z, they can then compute the count error analytically for any number of columns selectors without incurring additional privacy loss.

The rationale is an iterated application of the post-processing guarantee. First, distribution z is the only input to Algorithm 1 that is computed from the underlying data. Therefore, when Algorithm 1 with a column selector κ produces an output T using z, no additional privacy loss is incurred. Second, the metrics we have chosen to represent count error $\langle W, T\rangle$ depend only on z and T, so they can be computed without additional privacy loss. This argument holds for any number of columns selectors κ tested. Hence, a data holder can create a transition matrix for every available column selector and keep the one with the lowest count error without incurring additional privacy loss beyond the ϵ_1 used to generate z.

5.3 Two-Stage Unfixed Optimum Constructor

While the main focus of this section was the creation of fixed-point constructors, we will want to compare our techniques against a set of baselines that do not enforce a fixed-point constraint. One such baseline can be developed by adapting our two-stage framework. We envision a constructor algorithm that minimizes count error over the unfixed polytope U instead of F. Even though U does not depend on the distribution z, the distribution privatization step is still required since common measures of count error depend on the distribution of counts (e.g., expected total deviation is given by $\langle W, T\rangle$, with $w_{ij} = z_i|i - j|$). For that reason, we now develop a constructor that selects a count mechanism from U. This will form the basis of our *two-stage unfixed optimum*.

As was the case for F, it would theoretically be possible to minimize count error over U with standard linear programming algorithms, but these would take impractically long for some of the datasets we study. We therefore seek to leverage the structure of polytope U to find the lowest count error point more efficiently.

First, in Appendix D, we show that count mechanisms in U can be represented using scales in a similar way as those in F, yielding a new representation polytope, R_U (Theorem 10). Next, in Appendix E we leverage a result from Ghosh et al. [21], which tells us exactly which scales make up the optimal point in U for a natural class of count error measures, which we call *row-wise concentrating*.

Definition 11. A weight matrix $W \in \mathbb{R}_{\geq 0}^{n\times n}$ is *row-wise concentrating* if $w_{i,j}$ is non-decreasing in $|i - j|$ for each $i \in \{0, ..., n - 1\}$.

Theorem 12 tells us that the scales that make up a count mechanism in U that minimizes a row-wise concentrating weight matrix are exactly the single peaked scales. We let Σ be the $n \times n$ matrix, where column j is the single-peaked scale at position j.

We will further restrict attention to a narrower class of weight matrices, which will enable certain optimizations, stated below.

Definition 12. A weight matrix W is *row-wise convex* if for every $i \in \{0, ..., n - 1\}$ and every $j \in \{0, ..., n - 2\}$, $w_{i,j+1} - w_{i,j}$ is non-decreasing in j.

Note that commonly used count-error measures, including expected absolute deviation and mean squared error, are row-wise concentrating and row-wise convex. Row-wise convexity is useful because it ensures that once we find a column for a single-peaked scale that minimizes count error in a local sense, it is also optimal over all columns (Lemma 9). Putting these ideas together, Algorithm 2 places the single peaked scales into a transition matrix one at a time.

Algorithm 2: Two-Stage Unfixed Optimum Constructor

Input:

- Privacy parameter ϵ
- Weight matrix W that is both row-wise concentrating and row-wise convex

Initialize Σ and $\{\omega_l\}_{l=0}^{n-1}$ as described in Observation 3.
Initialize an $n \times n$ matrix A of 0's, $j = 0$, and $l = 0$.
while $l \leq n - 1$ **do**
  $current\_error \leftarrow \omega_l(W_j)^\top \Sigma_l$
  if $j < n - 1$ **then**
    $next\_error \leftarrow \omega_l(W_{j+1})^\top \Sigma_l$
  else
    $next\_error \leftarrow \infty$
  end
  if $next\_error > current\_error$ **then**
    $A_j \leftarrow A_j + \omega_l \Sigma_l$
    $l \leftarrow l + 1$
  else
    $j \leftarrow j + 1$
  end
end
Return A

The following two theorems establish that the two-stage unfixed optimum runs efficiently and outputs a count mechanism that minimizes count error.

Theorem 7. *The worst-case runtime of Algorithm 2 is $O(n^2)$.*

Theorem 8. *Algorithm 2 outputs a count mechanism $O \in ex(U)$ that minimizes count error. That is, for all $\hat{O} \in U$, $\langle W, O\rangle \leq \langle W, \hat{O}\rangle$.*

6 Experiments

In this section, we explore the *practicality* of fixed-point methods for privatizing a table of counts. We perform a set of simulations in which we measure the performance of different privatization methods. Our simulations are grouped into three experiments, corresponding to our three performance criteria: accuracy of distribution, accuracy of counts, and runtime. As we detail in this section, there are some scenarios in which a fixed-point method can dramatically improve accuracy of distribution with only moderate cost to accuracy of counts and runtime; however, considerable nuance can be found through each of our experiments.

Each of our simulations represents a complete privatization of a table of counts. All experiments were written in Python 3.12.7 and executed on a single machine. We include the following privatization methods.

Fixed-Point Methods. We implement our two-stage fixed-point framework. In the role of distribution privatizer $\mathcal{D}$, we select the cyclic Laplace mechanism, guided by the favorable results of Section 3. While other choices for $\mathcal{D}$ are possible, fixing the distribution privatizer allows us to focus on the effects of other experimental parameters. As a robustness check, we replicated key experiments with each of the other distribution privatizers from Section 3 and observed similar trends. For the constructor algorithm C, we implement the following choices.

- The simplex algorithm provided by the open-source SciPy Python library.
- The interior point method provided by the open-source SciPy Python library.
- Our heuristic algorithm, Algorithm 1. Furthermore, we explore several options for the column selector κ.
  - Max probability. This selector always chooses the remaining column j with maximum z_j.
  - Min probability. This selector always chooses the remaining column j with minimum z_j.
  - Sandwich. This selector always follows a prescribed pattern: $0, n-1, 1, n-2, 2\ldots$ until every column has been chosen.

Unfixed Baselines. As points of comparison, we implement a set of privatization algorithms that do not enforce a fixed-point constraint.

- Two-stage unfixed optimum given in Algorithm 2.
- Truncated geometric of Ghosh et al. [21]. This mechanism may not coincide with the output of the two-stage unfixed optimum, because it does not include the post-processing described by Ghosh el al. As such, there is no guarantee that the truncated geometric minimizes a given notion of count error.
- Staircase mechanism of Geng et al. [18], which minimizes count error for a certain class of mechanisms whose outputs are in $\mathbb{R}$. Because this mechanism outputs a real number, we post-process the output by mapping to the nearest integer in $\{0, \ldots, n-1\}$.[7]
- Discrete Gaussian of Canonne et al. [8]. This is the integer-analog of the commonly used Gaussian mechanism, and satisfies the weaker privacy notion of approximate (ϵ, δ)-differential privacy. We set $\delta = 1/(N+1)$, maximizing the benefit to this baseline while satisfying the convention that $\delta < 1/N$ [15, 37]. We post-process the output of this mechanism to ensure that it lies between 0 and $n-1$.

Note that the latter three mechanisms do not utilize a probability distribution as input. Accordingly, they are implemented as standalone algorithms, not part of a two-stage framework.

For each of our privatization methods, we vary the following experimental parameters.

Input Data. We select three datasets to have a range of distributional characteristics (see Figure 4).

- 10,000 synthetic draws from a Binomial distribution with size parameter 20 and probability parameter 1/2. This is a symmetric distribution that closely approximates a normal distribution.
- Real counts of homicides in each US County.[8] When conducting accuracy experiments, we top code these counts at 50. This dataset has a considerable right skew (5.83) and a substantial peak at zero (53% of counties report zero homicides). It is our smallest dataset with 3,136 total counties.
- Real counts of full-time teachers in public elementary and secondary education facilities in the US.[9] When conducting accuracy experiments, we top code these counts at 80. This data was published by the US Department of Education for the 2017-2018 school year. This dataset has a mild positive skewness (1.29). It is our largest dataset, representing 95,969 schools.

Privacy Parameters. We set a total privacy budget, ϵ_t, to values ranging from 0.1 to 5. As noted in Section 1.2, the two-stage counting framework requires this budget to be divided into two components: ϵ_1 for the distribution privatizer $\mathcal{D}$, and ϵ_2 for the count mechanism T, with $\epsilon_t = \epsilon_1 + \epsilon_2$. Let $f = \epsilon_1/\epsilon_t$ represent the fraction allocated to the distribution privatizer. Setting f is complicated by two stylized facts. First, dividing the privacy budget requires a value judgement, as it trades off between accuracy of counts and accuracy of distribution. Second, both types of accuracy generally depend on the underlying data, and existing methods of estimating these for candidate values of f require considerable privacy budget. For these reasons, we pursue the development of a rule of thumb that is independent of data, and that practitioners can use to set f.

We detail the development of our rule of thumb in Appendix F. To summarize our process, we begin with a *combined error objective* of the form $\ln(\text{count\_error}(f)) + \ln(\text{distribution\_error}(f))$. This

[7] The staircase mechanism includes a parameter $\gamma \in [0, 1]$, which dictates the width the center-most step. When $\gamma = 1/2$ the staircase mechanism coincides with the truncated geometric. We follow the authors' recommendation and set $\gamma = (1 + \exp(\epsilon/2))^{-1}$ to minimize the expected noise amplitude, resulting in a distinct mechanism. However, in our experiments, we found that the performance of the staircase mechanism was visually indistinguishable from the truncated geometric.

[8] Downloaded from https://www.kaggle.com/datasets/mikejohnsonjr/united-states-crime-rates-by-county

[9] Downloaded from https://public.opendatasoft.com/explore/dataset/us-public-schools/

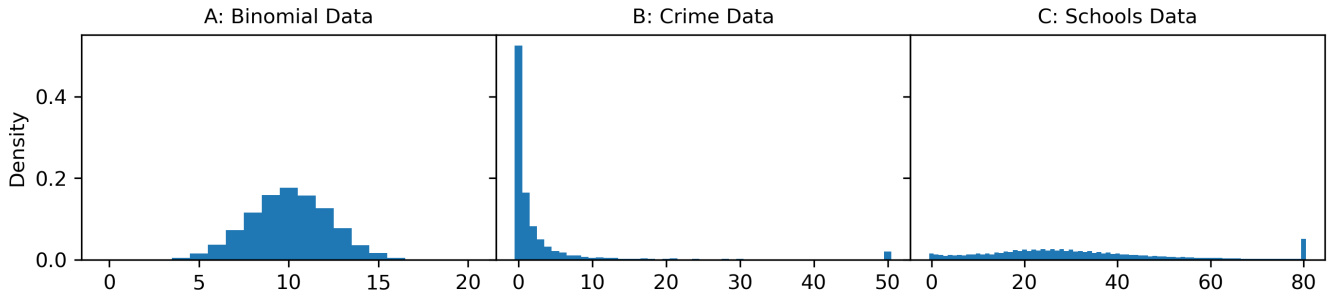

Figure 4: Distributions of the three datasets used in our experiments: (A) synthetic draws from a binomial distribution, (B) observed counts of homicides in US counties, and (C) observed counts of teachers in public schools.

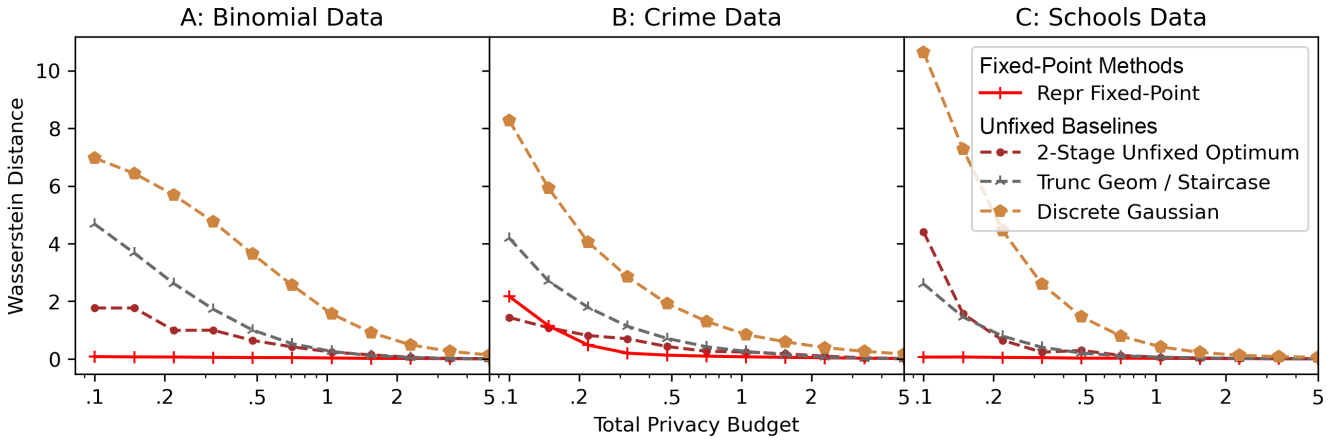

Figure 5: Distribution error under different total privacy budgets on logarithmic scale. All fixed-point constructors were visually overlapping, so we include only the heuristic with sandwich selector to represent all fixed-point constructors.

objective has the interpretation that, at the optimum, a multiplicative decrease in one type of error must be matched by an equal (or greater) multiplicative increase in the other. We perform a set of simulations on six new synthetic datasets, varying the total privacy budget, and computing the value of f that minimizes the combined error objective. We observe an inverse relationship between ϵ_t and the optimal f that appears in all datasets tested. We fit a model on these simulation results, yielding the following rule of thumb.

$$\hat{f}(\epsilon_t) = 0.106 + 0.533 \exp(-2.87\epsilon_t)$$

We employ this rule of thumb whenever we simulate the two-stage framework. Because the truncated geometric, staircase, and discrete Gaussian mechanisms do not require a two-stage framework, we grant them the entire privacy budget ϵ_t.

6.1 Accuracy of Distribution

In our first experiment, we investigate how the choice of privatization method affects accuracy of distribution. We run a set of simulations, varying the privatization method, dataset, and privacy parameters as described above. To operationalize accuracy of distribution, we use total variation distance, Kolmogorov–Smirnov (KS) distance, and Wasserstein-1 distance. Figure 5 shows the Wasserstein distance between the output distribution of counts and the true input distribution from a set of simulations; the corresponding plots for KS distance and total variation distance are similar and we omit them. We found that all fixed-point methods were closely overlapping, so we represent them with a single curve. Similarly, the truncated geometric and staircase mechanisms were closely overlapping, so we also represent them with a single curve.

As seen in Figure 5, the introduction of a fixed-point constraint often leads to a substantial improvement in accuracy of distribution. For example, for the binomial dataset at $\epsilon_t = 0.48$, switching from the best performing baseline (two-stage unfixed optimum) to a fixed-point constructor reduces Wasserstein distance from 0.64 to 0.04, a 94% decrease in distribution error. For the crime and schools dataset, the corresponding decreases in distribution error are 74% and 87% respectively. There are exceptions in which an unfixed baseline has slightly better accuracy of distribution than a fixed-point constructor. In particular, for the crime data at $\epsilon = 0.1$, Wasserstein distance is 1.44 for the two-stage unfixed optimum, but rises to 2.17 for the fixed-point methods. Aside from the crime data, the fixed point constraint seem to offer the greatest advantage when ϵ_t is small. In particular, at $\epsilon_t = 0.1$, for the schools dataset, the fixed-point methods feature a Wasserstein distance of 0.06. The closest baseline is the truncated geometric mechanism, with a Wasserstein distance of 2.60 – more than 40-times larger.

Overall, we believe this first experiment presents promising evidence in favor of imposing a fixed-point constraint. While such

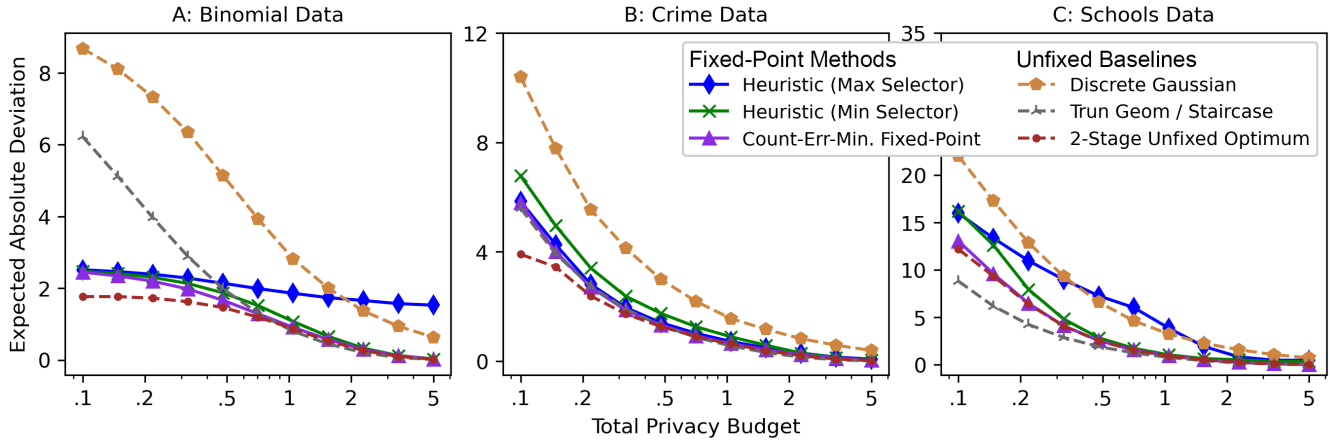

Figure 6: Count error under different total privacy budgets on logarithmic scale. As expected, an unfixed baseline yields lower count error than the fixed-point methods.

a constraint does not lead to perfect equality between the output distribution of counts and the true input, it represents a substantial improvement over current practice.

6.2 Accuracy of Counts

Our second experiment investigates how our fixed-point methods affect accuracy of counts. As noted earlier, we expect our techniques to involve some performance loss in this dimension. One reason is that they select a count mechanism from the polytope F, which is a strict subset of the unfixed polytope U. A more intuitive reason is that fixed-point techniques split attention between accuracy of distribution and accuracy of counts, whereas prior techniques focus exclusively on accuracy of counts. While some performance loss is expected, we believe that its magnitude is important. If users are to adopt a fixed-point technique, the loss in accuracy of counts cannot be too large, relative to the gain in accuracy of distribution.

To measure how much fixed-point methods degrade accuracy of counts, we run a set of simulations, varying the privatization method, dataset, and privacy parameters as before. To operationalize count error, we use both expected absolute deviation and mean squared error. Figure 6 shows the expected absolute deviation from a set of simulations; the corresponding plot for mean squared error is similar so we omit it. Additionally, we omit the curve for the sandwich selector, noting that it closely overlaps with the better of the max or min selector for all datasets.

We first observe that the two-stage unfixed optimum is consistently near the bottom of each plot, which we expect since this method outputs the best possible count mechanism in the polytope U_{ϵ_2}. Somewhat surprisingly, there are situations in which the truncated geometric and staircase mechanisms outperform the two-stage unfixed optimum. The reason this is possible is that these mechanisms don't require a privatized estimate of z, and hence the entire privacy budget ϵ_t can be used to privatize the table of counts. The truncated geometric and staircase mechanisms have the most visible advantage in the schools data, with the advantage shrinking with higher ϵ_t. One possible explanation relates to the shape of the schools data, which is the closest to a uniform among our three datasets. Recall that the two-stage unfixed optimum can be viewed as a truncated geometric mechanism with post-processing [21]. When data is near uniform, we expect that the optimal transition matrix in U_{ϵ_2} closely resembles the truncated geometric, so the benefits of post-processing are small. On the other hand, the truncated geometric mechanism enjoys the benefit of choosing a point from the larger polytope U_{ϵ_t}, even though that point is not optimal.

Among the fixed-point methods, the count-error-minimizing fixed-point constructors (i.e. fixed-point constructors using simplex or the interior point method) have the lowest count error. These methods are never the lowest curve in the graph, and in particular, the two-staged unfixed optimum always has lower count error. One reason is that adding constraints reduces the set of feasible mechanisms (from U_{ϵ_2} to F_{ϵ_2}). Even though switching from the best baseline to a count-error minimizing fixed-point method does increase count error, the gap is often moderate. For example, for the crime data, at $\epsilon_t = 0.48$, switching from the best baseline (two-stage unfixed optimum) to a count-error-minimizing fixed-point method increases count error from 1.27 to 1.32, a 5.7% increase (recall from the prior section that for the same switch, distribution error decreased by 74%). The largest performance gaps tend to occur for small values of ϵ_t.

The heuristic fixed-point constructors appear as a set of curves above the count-error-minimizing fixed point methods. Since these constructors also output a point in $F_{\epsilon_2}[z]$, they must result in at least as much count error as the minimum value over F. An interesting finding is that no column selector consistently outperforms the alternatives. In fact, by manipulating experimental parameters, we found situations in which each of the column selectors we studied was the best choice. As a general observation, the sandwich selector tends to have either the lowest count error, or close to the lowest for all datasets and parameter choices we tested. The max selector performed well for the highly skewed crime dataset, but poorly for the other two. Alternatively, the min selector performed well for the binomial and schools data, both of which exhibit more symmetry than the crime data.

When using a heuristic constructor is necessary (because it is computationally impractical to find the fixed-point optimum), the cost of imposing a fixed-point constraint, relative to the best unfixed

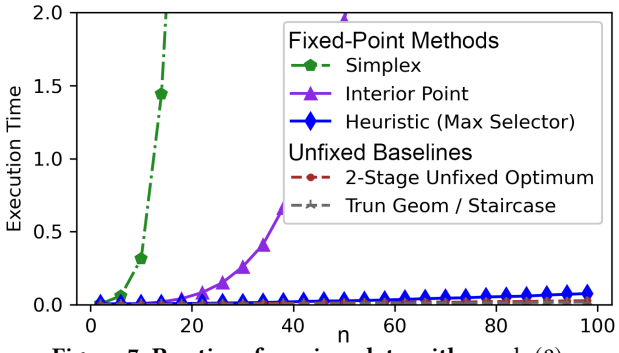

Figure 7: Runtime for crime data with $\epsilon_t = \ln(2)$.

baseline, is larger. However, the additional cost of switching from a count-error-minimizing fixed-point constructor to a heuristic constructor is usually less than the cost of switching from the best unfixed baseline to a count-error-minimizing fixed-point constructor. For example, in the crime dataset, when $\epsilon_t = 1$, the expected absolute deviation is 0.55 for the truncated geometric, 0.64 for the count-error-minimizing fixed-point constructors, and 0.66 for the best-performing heuristic (sandwich selector).

Overall, we find these results promising. As expected, count error does increase when adopting a fixed-point constraint. However, the increase is often a matter of a few percentage points and we expect that users will find this acceptable in some scenarios.

6.3 Runtime

In our final experiment, we measure how the choice of privatization method affects runtime. This is important because an algorithm must finish running in a reasonable time to be deployable.

We measure runtime using the system clock on a computer as it executes each privatization algorithm. Since each constructor algorithm must choose an $n \times n$ matrix, we expect that the bound on the number of counts, $n-1$, would have a strong effect on runtime. We therefore adjust the datasets used in this experiment so that we can vary n. For the binomial dataset, we continue taking draws with probability parameter $1/2$ but with size parameter set to $n-1$. For the crime and school datasets, we use the same raw data, but top code counts at $n-1$.

Measurements of runtime for different values of n for the crime dataset are shown in Figure 7. Plots for the binomial and schools dataset are similar and we omit them. Additionally, we found that the selector function had no visual impact on the runtime of our heuristic algorithm; we therefore include just a single selector function in the plot. Omitted from the plot is the Discrete Gaussian, whose runtime is consistently off the plot due to its use of rejection sampling [8]. The truncated geometric and staircase mechanisms were closely overlapping, so they are represented by a single curve.

For all parameter choices, we find that the truncated geometric and staircase baselines provide the fastest runtime. These are followed closely by the two-stage unfixed optimum. Among the fixed-point constructors, we find that the one that uses the interior point method, and especially the one that uses simplex, exhibit rapid growth rates. This suggests that their use must be limited to smaller values of n. Our heuristic constructor, on the other hand, performs more efficiently, with runtimes closer to that of the unfixed baselines. To explore this further, we ran additional simulations with even larger values of n. We were able to execute our heuristic constructor with $n = 2,000$ in under 10 seconds on a single laptop.

Overall, these results provide further evidence supporting the practicality of a fixed-point constraint. While the fixed point leads to higher runtimes relative to the unfixed baselines, the use of heuristic constructors serves as an effective technique to ensure that runtimes remain reasonable, even for high values of n.

7 Discussion

This study proposes a framework for computing tables of privatized counts that balances three performance criteria: accuracy of distribution, accuracy of counts, and runtime. We support the design of such a framework with a mathematical theory of transition matrices that obey differential privacy and fixed-point constraints. We provide advances for two framework components: the distribution privatizer and the constructor algorithm. Our experiments provide evidence that a fixed-point constraint can be practical, often yielding a favorable tradeoff among our three performance criteria.

Our study leaves open several questions. We would like to know if, under a reasonable class of loss functions, we can provide a compact description of the global optimum of F, analogous to the one found by Ghosh et al. for the unfixed case [21]. A little more broadly, we are interested in whether an algorithm for finding the global optimum exists that is more efficient than today's general-purpose solvers. Furthermore, numerous variants of differential privacy have been proposed [11], some of which may be well-suited for a polytope-style analysis. In particular, the set of (ϵ, δ)-differentially private count mechanisms forms a polytope in $\mathbb{R}^{n \times n}$. We would therefore like to investigate whether an analogue of ϵ-scales can be applied to this setting.

We are also interested in whether the upper bound n can be dropped from our framework, alleviating the need for truncating counts. We believe this would require radically different techniques, including infinite-dimensional analogs of the mathematical tools developed in this paper.

Another interesting direction involves preserving distributions for other types of data beyond simple counts. For example, many applications rely on the joint distribution of two count variables. Mortality rates and disease rates are typically computed as a fraction, meaning that systematic distortions in the joint distribution of incidence counts and population counts can interfere with our understanding of health trends [23, 34, 41]. Similarly, measures of segregation are based on the joint distribution of minority counts and majority counts, and can be biased upwards or downwards by privatization [2]. We believe the techniques developed in this study provide a promising starting point for developing privatization algorithms that can support applications like these.

Acknowledgments

This work was greatly improved by comments from our anonymous reviewers, our revision editor, Joshua Blumenstock, and members of UC Berkeley's Global Opportunity Lab. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

[1] John M Abowd. 2018. The US Census Bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2867–2867.

[2] Brian Asquith, Brad Hershbein, Tracy Kugler, Shane Reed, Steven Ruggles, Jonathan Schroeder, Steve Yesiltepe, and David Van Riper. 2022. Assessing the impact of differential privacy on measures of population and racial residential segregation. *Harvard Data Science Review* Special Issue 2 (2022).

[3] David Avis and Komei Fukuda. 1991. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. In *Proceedings of the Seventh Annual Symposium on Computational Geometry*. 98–104.

[4] Borja Balle and Yu-Xiang Wang. 2018. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International conference on machine learning*. PMLR, 394–403.

[5] Wayne D. Blizard et al. 1989. Multiset theory. *Notre Dame Journal of formal logic* 30, 1 (1989), 36–66.

[6] Fabrizio Boninsegna and Francesco Silvestri. 2025. Differential Privacy Releasing of Hierarchical Origin/Destination Data with a TopDown Approach. *Proceedings on Privacy Enhancing Technologies* (2025).

[7] US Census Bureau. 2021. Disclosure avoidance for the 2020 census: An introduction.

[8] Clément L Canonne, Gautam Kamath, and Thomas Steinke. 2020. The discrete gaussian for differential privacy. *Advances in Neural Information Processing Systems* 33 (2020), 15676–15688.

[9] United States Congress. 2024. No FAMS at the Border Act of 2024.

[10] Ryan Cumings-Menon, Robert Ashmead, Daniel Kifer, Philip Leclerc, Matthew Spence, Pavel Zhuravlev, and John M Abowd. 2023. Disclosure Avoidance for the 2020 Census Demographic and Housing Characteristics File. *arXiv preprint arXiv:2312.10863* (2023).

[11] Damien Desfontaines and Balázs Pejó. 2020. SoK: Differential privacies. *Proceedings on Privacy Enhancing Technologies* (2020).

[12] Fanny Dufossé, Kamer Kaya, Ioannis Panagiotas, and Bora Uçar. 2018. Further notes on Birkhoff–von Neumann decomposition of doubly stochastic matrices. *Linear Algebra Appl.* 554 (2018), 68–78.

[13] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28-June 1, 2006. Proceedings 25*. Springer, 486–503.

[14] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*. Springer, 265–284.

[15] Cynthia Dwork and Aaron Roth. 2014. The algorithmic foundations of differential privacy. *Foundations and trends® in theoretical computer science* 9, 3–4 (2014), 211–407.

[16] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 1054–1067.

[17] Natasha Fernandes, Kacem Lefki, and Catuscia Palamidessi. 2019. Utility-preserving privacy mechanisms for counting queries. *Models, Languages, and Tools for Concurrent and Distributed Programming: Essays Dedicated to Rocco De Nicola on the Occasion of His 65th Birthday* (2019), 487–495.

[18] Quan Geng, Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2015. The staircase mechanism in differential privacy. *IEEE Journal of Selected Topics in Signal Processing* 9, 7 (2015), 1176–1184.

[19] Quan Geng and Pramod Viswanath. 2013. The optimal mechanism in differential privacy: Multidimensional setting. *arXiv preprint arXiv:1312.0655* (2013).

[20] Quan Geng and Pramod Viswanath. 2014. The optimal mechanism in differential privacy. In *2014 IEEE international symposium on information theory*. IEEE, 2371–2375.

[21] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. 2009. Universally utility-maximizing privacy mechanisms. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 351–360.

[22] Darald Hartfiel. 1974. A study of convex sets of stochastic matrices induced by probability vectors. *Pacific J. Math.* 52, 2 (1974), 405–418.

[23] Mathew E Hauer and Alexis R Santos-Lozada. 2021. Differential privacy in the 2020 census will distort COVID-19 rates. *Socius* 7 (2021), 2378023121994014.

[24] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. 2010. Boosting the Accuracy of Differentially Private Histograms Through Consistency. *Proceedings of the VLDB Endowment* 3, 1 (2010).

[25] Ashley Hinson. 2024. Hinson Bill Forces Biden Administration to Keep Air Marshals on Flights Instead. https://hinson.house.gov/media/press-releases/hinson-bill-forces-biden-administration-keep-air-marshals-flights-instead.

[26] Shlomi Hod and Ran Canetti. 2025. Differentially private release of Israel's national registry of live births. In *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 3912–3930.

[27] Naoise Holohan, Douglas J Leith, and Oliver Mason. 2017. Extreme points of the local differential privacy polytope. *Linear Algebra Appl.* 534 (2017), 78–96.

[28] JASON 2020. 2022. *Consistency of Data Products and Formal Privacy Methods for the 2020 Census*. Technical Report. The MITRE Corporation.

[29] Wolfgang B Jurkat and Herbert John Ryser. 1967. Term ranks and permanents of nonnegative matrices. *Journal of Algebra* 5, 3 (1967), 342–357.

[30] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2016. Extremal mechanisms for local differential privacy. *The Journal of Machine Learning Research* 17, 1 (2016), 492–542.

[31] Nitin Kohli, Emily Aiken, and Joshua E Blumenstock. 2024. Privacy guarantees for personal mobility data in humanitarian response. *Scientific Reports* 14, 1 (2024), 28565.

[32] Vicki W Kramer. 2006. *Critical mass on corporate boards: Why three or more women enhance governance*. Wellesley Centers for Women.

[33] Chao Li, Gerome Miklau, Michael Hay, Andrew McGregor, and Vibhor Rastogi. 2015. The matrix mechanism: optimizing linear counting queries under differential privacy. *The VLDB journal* 24 (2015), 757–781.

[34] Yanran Li, Brent A Coull, Nancy Krieger, Emily Peterson, Lance A Waller, Jarvis T Chen, and Rachel C Nethery. 2023. Impacts of census differential privacy for small-area disease mapping to monitor health inequities. *Science Advances* 9, 33 (2023), eade8888.

[35] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Škoric. 2020. Estimating numerical distributions under local differential privacy. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 621–635.

[36] National Association for College Admission Counseling (NACAC). 2025. School Counseling: Caseloads and Responsibilities. https://www.nacacnet.org/school-counseling/.

[37] Hector Page, Charlie Cabot, and Kobbi Nissim. 2018. Differential privacy an introduction for statistical agencies. *NSQR. Government Statistical Service* (2018), 1–53.

[38] PricewaterhouseCoopers LLP. 2017. *The Governance Divide: Boards and Investors in a Shifting World – PwC's 2017 Annual Corporate Directors Survey*. Technical Report. Governance Insights Center, PricewaterhouseCoopers LLP. https://www.pwc.com/us/en/governance-insights-center/annual-corporate-directors-survey/assets/pwc-2017-annual-corporate--directors--survey.pdf

[39] Wahbeh Qardaji, Weining Yang, and Ninghui Li. 2013. Understanding hierarchical methods for differentially private histograms. *Proceedings of the VLDB Endowment* 6, 14 (2013), 1954–1965.

[40] Parastoo Sadeghi, Shahab Asoodeh, and Flavio du Pin Calmon. 2020. Differentially private mechanisms for count queries. *arXiv preprint arXiv:2007.09374* (2020).

[41] Alexis R Santos-Lozada, Jeffrey T Howard, and Ashton M Verdery. 2020. How differential privacy will affect our understanding of health disparities in the United States. *Proceedings of the National Academy of Sciences* 117, 24 (2020), 13405–13412.

[42] Merveille Koissi Savi, Akash Yadav, Wanrong Zhang, Navin Vembar, Andrew Schroeder, Satchit Balsari, Caroline O Buckee, Salil Vadhan, and Nishant Kishore. 2023. A standardised differential privacy framework for epidemiological modelling with mobile phone data. *medRxiv* (2023), 2023–03.

[43] Allan R Willms. 2008. Analytic results for the eigenvalues of certain tridiagonal matrices. *SIAM journal on matrix analysis and applications* 30, 2 (2008), 639–656.

[44] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. 2010. Differential privacy via wavelet transforms. *IEEE Transactions on knowledge and data engineering* 23, 8 (2010), 1200–1214.

[45] Yingtai Xiao, Zeyu Ding, Yuxin Wang, Danfeng Zhang, and Daniel Kifer. 2021. Optimizing fitness-for-use of differentially private linear queries. *Proceedings of the VLDB Endowment* 14, 10 (2021), 1730–1742.

[46] Mengmeng Yang, Taolin Guo, Tianqing Zhu, Ivan Tjuawinata, Jun Zhao, and Kwok-Yan Lam. 2023. Local differential privacy and its applications: A comprehensive survey. *Computer Standards & Interfaces* (2023), 103827.

Appendix

A Proof and Other Details from Section 3

Proof of Theorem 1. The proof proceeds in 5 steps.

Step 1: *Relate the cyclic Laplace $V(\zeta)$ to a new mechanism $\hat{V}(\zeta)$ to facilitate analysis.* We introduce modified mechanism $\hat{V}(\zeta)$, which is identical to the cyclic Laplace mechanism $V(\zeta)$, except that it only outputs the first $n-1$ dimensions of the output, $\hat{V} = (V_0, ..., V_{n-2}) \in \mathbb{R}^{n-1}$. By Observation 1, the sum of the cyclic Laplace outputs must be 1. It is therefore possible to post-process the output of $\hat{V}(\zeta)$ by computing the dropped component, $V_{n-1} = 1 - \sum_{i=0}^{n-2} V_i$, thereby recovering the original mechanism. By the postprocessing guarantee, it is therefore sufficient to prove that $\hat{V}$ satisfied ϵ-differential privacy. This reduction allows us to use the convenient language of probability density functions in the remainder of the proof.

Step 2: *Compute the conditional probability density function of $\hat{V}$ given $L_{n-1} = l_{n-1}$.* The modified cyclic Laplace outputs $V_0, ..., V_{n-2}$ where $V_i = \zeta_i + L_i - L_{i+1}$ for all $i \in \{0, ..., n-2\}$. Repeatedly substituting $L_{j+1} - \zeta_j + V_j$ for L_j yields the following $n-1$ equations.

$$L_i = L_{n-1} + \sum_{j=i}^{n-2} V_j - \zeta_j \text{ for all } i \in \{0, ..., n-2\}$$

Given $l_{n-1} \in \mathbb{R}$, we condition on the event $L_{n-1} = l_{n-1}$. We also define the random vector $\hat{L} = (L_0, ..., L_{n-2})$. For any $\hat{l} = (\hat{l}_0, ..., \hat{l}_{n-2}) \in \mathbb{R}^{n-1}$, let $\phi(\hat{l})$ denote the joint density of $\hat{L}$. By independence of the L_i's, $\phi(\hat{l}) = \prod_{i=1}^{n-2} \varphi(\hat{l}_i; 1/(N\epsilon))$ where $\varphi(\cdot; 1/(N\epsilon))$ is the probability distribution function of the single-variable Laplace with scale parameter $1/(N\epsilon)$ (as described in Footnote 2 of Section 3). Since the scale parameter will not change throughout this proof, we omit it and write $\varphi(\hat{l}_i)$ in place of $\varphi(\hat{l}_i; 1/(N\epsilon))$ for brevity. Also by independence of the L_i's, the conditional density of $\hat{L}$ given $L_{n-1} = l_{n-1}$ is also ϕ.

Then we can write, $\hat{L} = g_\zeta(\hat{V}; l_{n-1})$ for function $g_\zeta(\,\cdot\,; l_{n-1}) : \mathbb{R}^{n-1} \to \mathbb{R}^{n-1}$,

$$g_\zeta(\hat{v}; l_{n-1})_i = l_{n-1} + \sum_{j=i}^{n-2} v_j - \zeta_j \text{ for all } i \in \{0, ..., n-2\}$$

Note that $g_\zeta(\,\cdot\,; l_{n-1})$ is a one-to-one function that relates random vector $\hat{V}$ with random vector $\hat{L}$. For any $\hat{v} = (v_0, ..., v_{n-2}) \in \mathbb{R}^{n-1}$, let $\rho_\zeta(\hat{v}|l_{n-1})$ denote the conditional probability density of $\hat{V}$ at $\hat{v}$, given $L_{n-1} = l_{n-1}$. By the standard change of coordinates for probability densities, we have:

$$\rho_\zeta(\hat{v}|l_{n-1}) = |\det(J(\hat{v}))|\ \phi(g_\zeta(\hat{v}; l_{n-1}))$$

where $J(\hat{v})$ is the Jacobian of $g_\zeta(\,\cdot\,; l_{n-1})$ evaluated at $\hat{v}$. It can be verified that $J(\hat{v})$ is upper triangular with ones on the diagonal, so $|\det(J(\hat{v}))| = 1$. Hence $\rho_\zeta(\hat{v}|l_{n-1}) = \phi(g_\zeta(\hat{v}; l_{n-1}))$.

Step 3: *Bound the ratio of conditional probability densities of $\hat{V}$ on neighboring inputs.* Consider two neighboring tables of counts d and d', with corresponding histograms η and η', and with corresponding distributions of counts ζ and ζ', respectively. Without loss of generality, suppose d' is formed from d by adding an individual to row i. Then $d'_i = d_i + 1$.

Without loss of generality, suppose d has count f in row i, d' has count $f+1$ in row i and d and d' are otherwise identical. Let $\mathbb{I}(\cdot)$ be the indicator function. Then

$$\eta'_f = \sum_{j=0}^{N-1} \mathbb{I}(d'_j = f) = \sum_{j \neq i} \mathbb{I}(d'_j = f)$$

because $\mathbb{I}(d'_i = f) = 0$. Also,

$$\eta_f = \sum_{j=0}^{N-1} \mathbb{I}(d_j = f) = \sum_{j \neq i} \mathbb{I}(d_j = f) + 1 = \sum_{j \neq i} \mathbb{I}(d'_j = f) + 1$$

Hence, $\eta'_f = \eta_f - 1$ and $\zeta'_{d_i} = \zeta_{d_i} - \frac{1}{N}$. By a similar argument, $\eta'_{f+1} = \eta_{f+1} + 1$ and $\zeta'_{f+1} = \zeta_{f+1} + \frac{1}{N}$. For all other $\bar{f} \notin \{f, f+1\}$, $\eta_{\bar{f}} = \eta'_{\bar{f}}$, so $\zeta_{\bar{f}} = \zeta'_{\bar{f}}$.

Next, consider the following ratio of conditional probability density functions.

$$\frac{\rho_\zeta(\hat{v}|l_{n-1})}{\rho_{\zeta'}(\hat{v}|l_{n-1})} = \frac{\phi(g_\zeta(\hat{v}; l_{n-1}))}{\phi(g_{\zeta'}(\hat{v}; l_{n-1}))} = \frac{\prod_{i=0}^{n-2} \varphi(g_\zeta(\hat{v}; l_{n-1})_i)}{\prod_{i=0}^{n-2} \varphi(g_{\zeta'}(\hat{v}; l_{n-1})_i)}$$

Plugging in the values for $g_\zeta(\hat{v}; l_{n-1})_i)$ and $g_{\zeta'}(\hat{v}; l_{n-1})_i)$, we have

$$\frac{\rho_\zeta(\hat{v}|l_{n-1})}{\rho_{\zeta'}(\hat{v}|l_{n-1})} = \frac{\prod_{i=0}^{n-2} \varphi(l_{n-1} + \sum_{j=i}^{n-2} v_j - \zeta_j)}{\prod_{i=0}^{n-2} \varphi(l_{n-1} + \sum_{j=i}^{n-2} v_j - \zeta'_j)}$$

Notice that $\sum_{j \leq i} \zeta'_j = \sum_{j \leq i} \zeta_j$ whenever $i \neq f$. Canceling equal terms,

$$\frac{\rho_\zeta(\hat{v}|l_{n-1})}{\rho_{\zeta'}(\hat{v}|l_{n-1})} = \frac{\varphi(l_0 + \sum_{j \leq f} \zeta_j - v_j)}{\varphi(l_0 + \sum_{j \leq f} \zeta'_j - v_j)}$$

Since $\sum_{j \leq f} \zeta'_j = \sum_{j \leq f} \zeta_j - \frac{1}{N}$, we have

$$\begin{aligned}
\frac{\rho_\zeta(\hat{v}|l_{n-1})}{\rho_{\zeta'}(\hat{v}|l_{n-1})} &= \frac{\varphi(l_0 + \sum_{j \leq f} \zeta_j - v_j)}{\varphi(l_0 + \sum_{j \leq f} \zeta_j - \frac{1}{N} - v_j)} \\
&= \frac{\exp(N\epsilon|l_0 + \sum_{j \leq f} \zeta_j - v_j|)}{\exp(N\epsilon|l_0 + \sum_{j \leq f} \zeta_j - \frac{1}{N} - v_j|)} \\
&\leq \exp(N\epsilon|1/N|) \\
&= \exp(\epsilon)
\end{aligned}$$

where the inequality follows by the triangle inequality. By a symmetric argument, $\frac{\rho_\zeta(\hat{v}|l_{n-1})}{\rho_{\zeta'}(\hat{v}|l_{n-1})} \geq \exp(-\epsilon)$.

Step 4: *Bound the ratio of (unconditional) probability densities of $\hat{V}$ on neighboring inputs ζ and ζ'.* Define $\rho_\zeta(\hat{v})$ as the unconditional density of $\hat{V}$. Then,

$$\frac{\rho_\zeta(\hat{v})}{\rho_{\zeta'}(\hat{v})} = \frac{\int_{\mathbb{R}} \rho_\zeta(\hat{v}|l_{n-1})\varphi(l_{n-1})dl_{n-1}}{\int_{\mathbb{R}} \rho_{\zeta'}(\hat{v}|l_{n-1})\varphi(l_{n-1})dl_{n-1}}$$

Upper-bounding the numerator using Step 3,

$$\frac{\rho_\zeta(\hat{v})}{\rho_{\zeta'}(\hat{v})} \leq \frac{\int_{\mathbb{R}} e^\epsilon \rho_{\zeta'}(\hat{v}|l_{n-1})\varphi(l_{n-1})dl_{n-1}}{\int_{\mathbb{R}} \rho_{\zeta'}(\hat{v}|l_{n-1})\varphi(l_{n-1})dl_{n-1}} = e^\epsilon$$

By a symmetric argument using the other bound from Step 3, we similarly deduce $\frac{\rho_\zeta(\hat{v})}{\rho_{\zeta'}(\hat{v})} \geq e^{-\epsilon}$.

Step 5: *Deduce that the original cyclic Laplace mechanism satisfies differential privacy.* By Step 4, the ratio of probability density functions is between $e^{-\epsilon}$ and e^ϵ at every point $\hat{v} \in \mathbb{R}^{n-1}$. Hence, by standard integration, for any measurable set $\hat{E} \subseteq \mathbb{R}^{n-1}$, we have

$e^{-\epsilon}\mathbb{P}(\hat{V}(\zeta') \in \hat{E}) \leq \mathbb{P}(\hat{V}(\zeta) \in \hat{E}) \leq e^{\epsilon}\mathbb{P}(\hat{V}(\zeta') \in \hat{E})$. By the reduction argument from Step 1, this implies that the cyclic Laplace mechanism also satisfies ϵ-differential privacy. □

Additional Applications of the Cyclic Approach. Our cyclic approach for the Laplace mechanism can also be used with other differentially private mechanisms, such as the Gaussian mechanism [4, 13].

Definition 13. We define the *cyclic Gaussian mechanism for distributions of counts* as a randomized algorithm that accepts a parameter σ and table of counts d as inputs. It then computes its distribution of counts ζ and independently samples $G_0, ..., G_{n-2}$ from Normal$(0, \sigma^2)$, defines $G_n = G_0$, and outputs $V_i = \zeta_i + G_i - G_{i+1}$ for all $i \in \{0, ..., n-1\}$.

This mechanism satisfies approximate differential privacy [13], for appropriately chosen values of σ. To sketch a proof, we first rewrite the cyclic Gaussian mechanism as $V = \zeta + H$, where $H_i = G_i - G_{i+1}$. We perform the technique in Step 1 of Theorem 1, and construct a new mechanism $\hat{V} = (V_0, ..., V_{n-2})$. This new mechanism $\hat{V}$ is equivalent to the Exact Correlated Gaussian Mechanism of Xiao et al. [45], with an $(n-1)\times(n-1)$ covariance matrix whose diagonal entries are $2\sigma^2$, whose superdiagonal and subdiagonal entries are both $-\sigma^2$, and all other entries are 0. This covariance matrix is invertible, as it is a symmetric tridiagonal Toeplitz matrix, so its eigenvalues are given by $\lambda_m = 2\sigma^2 + 2\sigma^2 \cos\left(\frac{m\pi}{n}\right)$ for $m \in \{1, ..., n-1\}$ [43], all of which are positive. Hence $\hat{V}$ satisfies approximate differential privacy by Theorem 3 of Xiao et al. [45] for appropriately chosen values of σ. By the post-processing argument presented in Step 1 of Theorem 1, the cyclic Gaussian mechanism for distributions of counts V also satisfies approximate differential privacy.

B Proofs and Other Details from Section 4

B.1 Using Ψ to Represent F

Proof of Lemma 1. To prove Lemma 1, the following technical lemma will be helpful; it establishes the linear independence of the differential privacy constraints and the row sum constraints on a transition matrix.

Lemma 3. *For any $\pi_i \in \{\pi_i^-, \pi_i^+\}$ for $i \in \{0, ...n-2\}$, the set of row vectors $\{\pi_0, ..., \pi_{n-2}, \mathbb{1}_n^\top\}$ are linearly independent.*

Proof. (Of Lemma 3.) Let W be the $n \times n$ matrix whose rows are $\pi_0, ..., \pi_{n-2}, \mathbb{1}_n^\top$ in this order. Suppose $v \in \mathbb{R}^n$ is in the kernel of W. If $v_0 > 0$, then row 0 of $Wv = 0_n$ specifies $\pi_0 v = 0$. If $\pi_0 = \pi_0^+$, then $\pi_0 v = v_0 - e^{\epsilon} v_1 = 0$, yielding $v_1 = e^{-\epsilon} v_0 > 0$. By similar argument, $\pi_0 = \pi_0^-$, also yields $v_1 > 0$. Repeating the process $n-1$ times, we know $v_i > 0$ for all $i \in \{0, .., n-1\}$. The last row of $Wv = 0_n$ specifies $\mathbb{1}_n^\top v = 0$. However, the left hand side is a sum of all positive entries, yielding a contradiction. By similar argument, v_0 cannot be negative. Therefore, $v_0 = 0$. By similar procedure as above, this implies $v_i = 0$ for all $i \in \{0, ..., n-1\}$. Hence the kernel of W only contains the zero vector, so W is full rank, implying the rows are linearly independent, as required. □

Proof. (Of Lemma 1) Let Q denote the set of probability vectors that satisfy ϵ-neighbor indistinguishability.

$$Q = \{t \in \Delta_n : \pi_i^+ t \leq 0, \pi_i^- t \leq 0 \text{ for all } 0 \leq i \leq n-2\}$$

Then Q is a convex polytope, and every scale $\Psi_j \in Q$. To prove the result, we will first show that the extreme points of Q are the scales, $ex(Q) = \{\Psi_0, ..., \Psi_{k-1}\}$.

A point in a convex polytope in $\mathbb{R}^n$ is an extreme point if and only if there are n linearly independent binding constraints. The privacy constraints guarantee that any $t \in Q$ has all positive entries. So for each $i \in \{0, ..., n-1\}$ only one of π_i^+ or π_i^- can bind. Therefore, all binding constraints are linearly independent by Lemma 3. This implies that a point in Q is extreme if and only if there are n binding constraints. In turn, this is equivalent to the statement that t is an ϵ-scale. This establishes $ex(Q) = \{\Psi_0, ..., \Psi_{k-1}\}$.

We now complete the proof of the lemma. Suppose $T \in U$ and consider the j^{th} column. We can write $T_j = \alpha_j q$ for some $\alpha_j \geq 0$ and $q \in Q$. Since the extreme points of Q are the ϵ-scales, we can write $q = \Psi\beta$ for $\beta \in \Delta_n$. Hence $T_j = \alpha_j \Psi\beta$, which is a conic combination of ϵ-scales. □

Proof of Theorem 2. ($\implies$) Suppose $T \in ex(U)$. A well known fact from geometry is that a point in a convex polytope in $\mathbb{R}^{n\times n}$ is an extreme point if and only if there are n^2 linearly independent binding constraints. Since U includes n row constraints, T must satisfy $n^2 - n = n(n-1)$ tight privacy constraints. Since each column is non-zero, only a single differential privacy constraint can bind for every pair of adjacent entries, meaning that each column is associated with a maximum of $n-1$ tight privacy constraints. It follows that every column must have exactly $n-1$ tight privacy constraints, hence every column of T is a multiple of an ϵ-scale.

The proof that the columns of T are linearly independent follows by a similar argument to Theorem 2 of Holohan et al. [27], which we include below for completeness. Assume for contradiction that there exists a linear dependence among the columns of T. Then there is a non-zero vector of weights, $\theta \in \mathbb{R}^n$ such that $\sum_{j=0}^{n-1} \theta_j T_j = 0_n$. Let $W = T\,\mathrm{diag}(\theta)$. Then $W \neq 0_{n\times n}$ as $\theta \neq 0_n$. Also, $W\mathbb{1}_n = 0_n$ by the linear dependency.

Choose any $\delta \in (0, 1/\max_j\{|\theta_j|\})$. Define $X = T - \delta W$ and $Y = T + \delta W$. By construction, $(T \pm \delta W)\mathbb{1}_n = T\mathbb{1}_n \pm \delta W\mathbb{1}_n = T\mathbb{1}_n = \mathbb{1}_n$, so the row sums of both X and Y are 1. Next, we show that all elements of X are positive. For $i, j \in \{0, ..., n-1\}$,

$$\begin{aligned} x_{i,j} &= t_{i,j} - \delta w_{i,j} \\ &= t_{i,j} - \delta \sum_{u=0}^{n-1} t_{i,u}\mathrm{diag}(\theta)_{u,j} \\ &= t_{i,j}(1 - \delta\theta_j) \\ &> t_{i,j}\left(1 - \frac{\theta_j}{\max_v\{|\theta_v|\}}\right) \\ &\geq 0 \end{aligned}$$

By similar argument, all elements of Y are positive. Next, we note that any column j of X is a positive multiple of the corresponding column of T, as $X_j = T_j(1 - \delta\theta_j)$. Since T_j is ϵ-neighbor indistinguishable, so is X_j. Therefore, $X \in U$. By a similar argument, $Y \in U$ as well. But then $T = \frac{1}{2}X + \frac{1}{2}Y$, implying that $T \notin ex(U)$, yielding a contradiction.

($\Longleftarrow$) Suppose $T \in U$ and the columns of T are linearly independent multiples of ϵ-scales. Assume for contradiction that T is not an extreme point. Then, there exists $X, Y \in U$, distinct, and $\alpha \in (0,1)$ such that $T = \alpha X + (1-\alpha)Y$.

For each column j we show that T_j is a constant multiple of X_j. This is equivalent to showing $x_{i,j}/t_{i,j} = x_{i+1,j}/t_{i+1,j}$ for all $i \in \{0, ..., n-2\}$. Assume for contradiction that there exists some $i \in \{0, ..., n-2\}$ such that $x_{i,j}/t_{i,j} \neq x_{i+1,j}/t_{i+1,j}$. We proceed by cases. If $t_{i,j} = e^{\epsilon} t_{i+1,j}$, then $x_{i,j} < e^{\epsilon} x_{i+1,j}$. Hence,

$$\begin{aligned} e^{\epsilon} t_{i+1,j} &= t_{i,j} \\ &= \alpha x_{i,j} + (1-\alpha) y_{i,j} \\ &< \alpha e^{\epsilon} x_{i+1,j} + (1-\alpha) y_{i,j} \\ &\leq \alpha e^{\epsilon} x_{i+1,j} + (1-\alpha) e^{\epsilon} y_{i+1,j} \\ &= e^{\epsilon} t_{i+1,j} \end{aligned}$$

producing a contradiction. An analogous argument holds when $t_{i,j} = e^{-\epsilon} t_{i+1,j}$, implying that X_j must be proportional to T_j. Similarly, Y_j must be proportional to T_j. Hence, there must be weight vectors $\theta, \nu \in \mathbb{R}^n$ such that $X = T \operatorname{diag}(\theta)$ and $Y = T \operatorname{diag}(\nu)$. Define $\beta = \theta - \nu$ and note that $\beta \neq 0_n$ since $X \neq Y$. Then

$$\begin{aligned} 0_n &= (X - Y)\mathbb{1}_n \\ &= (T \operatorname{diag}(\theta) - T \operatorname{diag}(\nu))\, \mathbb{1}_n \\ &= T \operatorname{diag}(\theta - \nu)\mathbb{1}_n \\ &= T \operatorname{diag}(\beta)\mathbb{1}_n \end{aligned}$$

which is a linear dependency on the columns of T, yielding a contradiction. □

Proof of Theorem 3. We first show that $F \subseteq \Psi(R_F)$. Given $T \in F$, it is also in U, so every column of T can be expressed a conic combination of ϵ-scales by Lemma 1. So for every column j there exists a column vector $\beta(j)$ such that $T_j = \Psi\beta(j)$. Let $B \in \mathbb{R}^{k \times n}_{\geq 0}$ be the matrix whose j^{th} column is $\beta(j)$. Then $T = \Psi B$. To complete the proof, we show $B \in R_F$. To do so, we show $z\Psi B = z$ and $\Psi B \mathbb{1}_n = \mathbb{1}_n$.

We establish the first condition $z\Psi B = z$ via contradiction. Suppose there exists some $j \in \{0, ..., n-1\}$ such that $z_j \neq (z\Psi B)_j = z\Psi B_j$. By construction, $T_j = \Psi B_j$, so $z_j \neq zT_j$, so $z \neq zT$, contradicting $T \in F$. Hence $z\Psi B = z$.

The second condition holds as well, as $\Psi B \mathbb{1}_n = T\mathbb{1}_n = \mathbb{1}_n$ since $T \in F$. Hence, given $T \in F$ there exists $B \in R_F$ such that $T = \Psi B$. This establishes $T \in \Psi(R_F)$.

Next, we show that $F \supseteq \Psi(R_F)$. Let $T \in \Psi(R_F)$. Then there exists some $B \in R_F$ such that $T = \Psi B$. By definition of R_F, $zT = z\Psi B = z$, and $T\mathbb{1}_n = \Psi B \mathbb{1}_n = \mathbb{1}_n$. Column j of T is given by ΨB_j. This is a conic combination of ϵ-neighbor indistinguishable vectors, and is hence ϵ-neighbor indistinguishable. Since every column of T is ϵ-neighbor indistinguishable, T is ϵ-differentially private, establishing $T \in F$. Hence, $\Psi(R_F) \supseteq F$, producing $\Psi(R_F) = F$. □

Proof of Proposition 1. We provide two proofs of this result.

Proof. (Approach 1) For two matrices X and Y of the same size, we write the Frobenius inner product as $\langle X, Y \rangle = tr[X^\top Y]$. A standard result from linear algebra is that a point X in a convex polytope P is an extreme point if and only if it is the unique maximizer of some linear function from P to $\mathbb{R}$. We restate this result in our context as follows.

Lemma 4. *Given convex polytope P in some vector space of real matrices V, a matrix $T \in P$ is an extreme point if and only if there exists $d \in V$ such that T is the unique maximizer of $\phi(\cdot) = \langle \cdot, d \rangle$ in P.*

Now, we prove Proposition 1. Suppose $T \in ex(F)$. Then by Lemma 4 there exists a $c \in \mathbb{R}^{n \times n}$ such that T is the unique maximizer of $\langle c, T' \rangle$ for all $T' \in F$. Consider $d = \Psi^\top c$ and note for any $B' \in R_F$, $\langle d, B' \rangle = tr[d^\top B'] = tr[c^\top \Psi B'] = \langle c, \Psi B' \rangle$. Let B be any extreme point of R_F that maximizes $\langle d, B' \rangle = \langle c, \Psi B' \rangle$ over all $B' \in R_F$. We must have $\Psi B = T$ because if $\Psi B = T' \neq T$, then for any $\bar{B}$ in the preimage of T through Ψ, we have $\langle d, B \rangle = \langle c, \Psi B \rangle = \langle c, T' \rangle < \langle c, T \rangle = \langle c, \Psi \bar{B} \rangle = \langle d, \bar{B} \rangle$ where the inequality follows from the maximality of T. This contradicts the maximality of B. Hence B is an extreme point in R_F such that $T = \Psi B$. □

Proof. (Approach 2) Suppose $T \in ex(F)$. Since Ψ is surjective, there exists some $W \in R_F$ such that $T = \Psi W$. Since R_F is convex, $W = \theta_1 B^{(1)} + ... + \theta_l B^{(l)}$ for some $B^{(1)}, ..., B^{(l)} \in ex(R_F)$ and positive θ_i's that sum to 1. Then

$$T = \Psi W = \sum_{i=1}^{l} \theta_l (\Psi B^{(i)})$$

Since T is extreme, the expression on the right-hand side must be the trivial convex combination. Hence, $\Psi B^{(i)} = T$ for all $i \in \{1, ..., l\}$. Let $B = B^{(1)}$. Then there is some $B \in ex(R_F)$ such that $T = \Psi B$. □

Example 1. Ψ is not a 1-1 mapping between $ex(R_F)$ and $ex(F)$. Consider the target distribution $z = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ with $\epsilon = \ln(2)$ and $n = 3$. There are 4 possible ϵ-scales, displayed in the matrix Ψ below.

$$\Psi = \begin{bmatrix} 4/7 & 2/5 & 1/4 & 1/7 \\ 2/7 & 1/5 & 1/2 & 2/7 \\ 1/7 & 2/5 & 1/4 & 4/7 \end{bmatrix}$$

Using the Avis-Fukuda vertex enumeration algorithm [3], we generate all extreme points of F and R_F. We find $|ex(F)| = 36$ and $|ex(R_F)| = 78$.

The matrices $B^{(1)}$ and $B^{(2)}$ below are extreme points of R_F, yet they both map to the same extreme point in F under Ψ.

$$B^{(1)} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 2/3 \\ 0 & 1 & 1/3 \\ 0 & 0 & 0 \end{bmatrix} \in ex(R_F)$$

$$B^{(2)} = \begin{bmatrix} 0 & 0 & 14/30 \\ 1 & 0 & 0 \\ 0 & 1 & 2/30 \\ 0 & 0 & 14/30 \end{bmatrix} \in ex(R_F)$$

$$\Psi B^{(1)} = \Psi B^{(2)} = \begin{bmatrix} 2/5 & 1/4 & 7/20 \\ 1/5 & 1/2 & 3/10 \\ 2/5 & 1/4 & 7/20 \end{bmatrix} \in ex(F)$$

Additionally, matrix $B^{(3)}$ below is an extreme point of R_F, yet $\Psi B^{(3)} \notin ex(F)$, as the matrix $\Psi B^{(3)}$ is not an output of the Avis-Fukuda algorithm over F.

$$B^{(3)} = \begin{bmatrix} 5/6 & 0 & 1/3 \\ 0 & 0 & 0 \\ 0 & 0 & 2/3 \\ 1/6 & 1 & 0 \end{bmatrix} \in ex(R_F)$$

△

Proof of Lemma 2. ($\implies$) We proceed by the contrapositive. Suppose there exists a non-zero movement matrix $\mu \in M_F(\Psi)$ such that $\mu_{i,j}$ is zero whenever $b_{i,j}$ is zero. Since $\mu \neq 0_{k\times n}$, there must be a column with at least one non-zero entry. Note that this column cannot have exactly one non-zero entry, because if $\mu_{i,j} \neq 0$ yet $\mu_{i',j} = 0$ for all $i' \neq i$, then $(z\Psi\mu)_j = (z\Psi)\mu_j = z\Psi_i\mu_{i,j} \neq 0$, implying $z\Psi\mu \neq 0_n^\top$, contradicting $\mu \in M_F(\Psi)$. Hence, there is some column of μ with at least two non-zero entries.

For column j, let $l(j)$ be the smallest row index i such that $b_{i,j} > 0$, and let $D(j)$ be the set of all other indices i such that $b_{i,j} > 0$. Then there exists some $j \in \{0, ..., n-1\}$ such that $D(j) \neq \emptyset$ because μ has a column with at least 2 non-zero entries and B must be non-zero at those positions. We can then write,

$$\Psi\mu_j = \Psi_{l(j)}\mu_{l(j),j} + \sum_{i\in D(j)} \Psi_i\mu_{i,j}$$

For all $i \in D(j)$, define h_i^j as the vector

$$h_i^j = (z\Psi_i)^{-1}\Psi_i - (z\Psi_{l(j)})^{-1}\Psi_{l(j)}$$

Note that the set of vectors $\{h_i^j : i \in D(j)\}$ is the set $H(B_j)$. Rearranging gives,

$$\Psi_i = z\Psi_i\left((z\Psi_{l(j)})^{-1}\Psi_{l(j)} + h_i^j\right)$$

Substituting into the above equation for $\Psi\mu_j$ yields,

$$\Psi\mu_j = \Psi_{l(j)}\mu_{l(j),j} + \sum_{i\in D(j)} z\Psi_i\left((z\Psi_{l(j)})^{-1}\Psi_{l(j)} + h_i^j\right)\mu_{i,j}$$

Next, we rewrite each term on the right-hand side of the above equation. The first term can be expressed as $z\Psi_{l(j)}(z\Psi_{l(j)})^{-1}\Psi_{l(j)}\mu_{l(j),j}$ and the sum can be rewritten as

$$\sum_{i\in D(j)} z\Psi_i(z\Psi_{l(j)})^{-1}\Psi_{l(j)}\mu_{i,j} + \sum_{i\in D(j)} z\Psi_i\mu_{i,j}h_i^j$$

Hence,

$$\begin{aligned}\Psi\mu_j = {} & z\Psi_{l(j)}(z\Psi_{l(j)})^{-1}\Psi_{l(j)}\mu_{l(j),j} \\ & + \sum_{i\in D(j)} z\Psi_i(z\Psi_{l(j)})^{-1}\Psi_{l(j)}\mu_{i,j} \\ & + \sum_{i\in D(j)} z\Psi_i\mu_{i,j}h_i^j\end{aligned}$$

which simplifies to

$$\Psi\mu_j = (z\Psi_{l(j)})^{-1}\Psi_{l(j)} \sum_{i\in D(j)\cup\{l(j)\}} z\Psi_i\mu_{i,j} + \sum_{i\in D(j)} z\Psi_i\mu_{i,j}h_i^j$$

By the definition of $M(\Psi)$, $z\Psi\mu = 0_n^\top$, meaning the first sum is 0. Therefore, we have

$$\Psi\mu_j = \sum_{i\in D(j)} z\Psi_i\mu_{i,j}h_i^j$$

So then

$$0_n = \Psi\mu\mathbb{1}_n = \sum_{j=0}^{n-1}\Psi\mu_j = \sum_{j=0}^{n-1}\sum_{i\in D(j)} z\Psi_i\mu_{i,j}h_i^j$$

The above is a linear dependency among the vectors of the multiset $\uplus_{j=0}^{n-1}H(B_j)$, implying that B is not Ψ-affinely simplified, as required.

($\impliedby$) Again by the contrapositive, suppose B is not Ψ-affinely simplified. Then $\uplus_{j=0}^{m-1}H(B_j)$ is linearly dependent. As above, let $l(j)$ be the smallest row i for which $b_{i,j} > 0$ and $D(j)$ be the set of all other indices i such that $b_{i,j} > 0$. The linear dependency can be written as,

$$\sum_{j=0}^{n-1}\sum_{x\in D(j)} \alpha_{x,j}\left((z\Psi_i)^{-1}\Psi_x - (z\Psi_{l(j)})^{-1}\Psi_{l(j)}\right) = 0_n$$

for a collection of $\alpha_{x,j}$'s, not all zero. We can rewrite the left-hand side of the above equation as

$$\sum_{j=0}^{n-1}\left(-\sum_{x\in D(j)}\alpha_{x,j}(z\Psi_{l(j)})^{-1}\right)\Psi_{l(j)} + \sum_{j=0}^{n-1}\sum_{x\in D(j)}\left(\alpha_{i,j}(z\Psi_x)^{-1}\right)\Psi_x$$

Let μ be the matrix such that $\mu_{i,j}$ equals $-\sum_{x\in D(j)}(z\Psi_{l(j)})^{-1}\alpha_{x,j}$ when $i = l(j)$, equals $\alpha_{i,j}(z\Psi_i)^{-1}$ whenever $i \in D(j)$, and equals 0 otherwise. Since the $\alpha_{x,j}$'s are not all zero, the matrix $\mu \neq 0_{k\times n}$. Also, the sum above can be written in matrix notation as $\Psi\mu\mathbb{1}_n = 0_n$. Additionally, the j^{th} column constraint of μ can be written,

$$\begin{aligned}(z\Psi\mu)_j &= z\sum_{i=0}^{n-1}\Psi_i\mu_{i,j} \\ &= z\Psi_{l(j)}\mu_{l(j),j} + \sum_{x\in D(j)} z\Psi_x\mu_{x,j} \\ &= -z\Psi_{l(j)}\sum_{x\in D(j)}(z\Psi_{l(j)})^{-1}\alpha_{x,j} + \sum_{x\in D(j)} z\Psi_x(z\Psi_x)^{-1}\alpha_{x,j} \\ &= -\sum_{x\in D(j)}\alpha_{x,j} + \sum_{x\in D(j)}\alpha_{x,j} \\ &= 0\end{aligned}$$

implying $z\Psi\mu = 0_n^\top$. Therefore $\mu \in M_F(\Psi)$. □

Proof of Theorem 4. ($\implies$) We proceed by the contrapositive. Suppose B is not Ψ-simplified. By the equivalence of Ψ-simplified and movement matrices established in Lemma 2, there exists a movement matrix $\mu \in M(\Psi)$ such that $\mu_{i,j} = 0$ whenever $b_{i,j} = 0$. Choose any positive

$$\delta < \min_{\{(i,j)\mid\mu_{i,j}\neq 0\}}\left\{\frac{b_{i,j}}{|\mu_{i,j}|}\right\}$$

Then $B \pm \delta\mu \in R_F$, as

(1) For any i, j entry, $\pm\delta\mu_{i,j} \leq b_{i.j}$ with equality if and only if $b_{i,j} = 0$. Hence $(B \pm \delta\mu)_{i,j} \geq 0$.
(2) $\Psi(B \pm \delta\mu)\mathbb{1}_n = \Psi B\mathbb{1}_n \pm \delta\Psi\mu\mathbb{1}_n = \mathbb{1}_n \pm 0_n = \mathbb{1}_n$
(3) $z\Psi(B \pm \delta\mu) = z\Psi B \pm \delta z\Psi\mu = z \pm 0_n^\top = z$

Hence, B can be represented as

$$B = \frac{1}{2}(B + \delta\mu) + \frac{1}{2}(B - \delta\mu)$$

Therefore $B \notin ex(R_F)$.

($\impliedby$) Proceed by the contrapositive yet again. Suppose $B \notin ex(R_F)$. Then there exists distinct $X, Y \in R_F$ and $\theta \in (0, 1)$ such that $B = \theta X + (1-\theta)Y$. By non-negativity, the convex combination above implies $x_{i,j} = 0 = y_{i,j}$ whenever $b_{i,j} = 0$. Let $\mu = X - Y$. Then $\mu_{i,j} = 0$ whenever $b_{i,j} = 0$. Also, $z\Psi\mu = z\Psi X - z\Psi Y = z - z = 0_n^\top$ and $\Psi\mu\mathbb{1}_n = \Psi X\mathbb{1}_n - \psi Y\mathbb{1}_n = \mathbb{1}_n - \mathbb{1}_n = 0_n$. So $\mu \in M_F(\Psi)$, implying B is not Ψ-simplified. □

Proof of Corollary 2. Let $B \in ex(R_F)$ and $\mathcal{P}$ be the set of positive indices of z. First, observe that for each $j \in \mathcal{P}$, $B_j \neq 0_k$ (as $z\Psi B_j = z_j \neq 0$), so there must be at least one positive entry in B_j, so B must contain at least $|\mathcal{P}|$ positive entries. To show the upper bound on the number of positive entries is $|\mathcal{P}|+n-1$, note that every element $h \in \uplus_{j=0}^{n-1} H(B_j)$ belongs to an $n-1$ dimensional subspace of $\mathbb{R}^n$ because of the constraint $zh = 0$. Since the elements are linearly independent, there can be at most $n-1$ of them. The number of positive entries in each column $j \in \mathcal{P}$ of B is at most one more than the number of difference vectors in $H(B_j)$ (because $H(B_j)$ is constructed to include one vector corresponding to each index i such that $b_{i,j} > 0$ except the smallest). Hence the number of positive elements of B is at most $|\mathcal{P}|$ plus the number of elements of $\uplus_{j=0}^{n-1} H(B_j)$, yielding a maximum of $|\mathcal{P}|+n-1$ positive entries. □

B.2 Characterizing extreme points that are preserved under Ψ.

In this section, we provide a necessary and sufficient condition that allows us to identify which extreme points in R_F are mapped to extreme points of F. Our theorem is more general than required, referring to the extreme points of a convex polytope P in an arbitrary $\hat{k} \times \hat{n}$ space of real matrices under affine surjection. We place hats over the dimensions as the following results hold for any dimensions $\hat{k} \geq \hat{n}$, and not only the specific values of k and n used in this paper. As before, for matrices X and Y of the same size, we write the Frobenius inner product as $\langle X, Y\rangle = tr[X^\top Y]$. We will make use of the following two definitions.

Definition 14. Given a convex polytope $P \subset \mathbb{R}^{\hat{k}\times\hat{n}}$ and a matrix $D \in \mathbb{R}^{\hat{k}\times\hat{n}}$ we define the maximizing set of the function $\langle \cdot, D\rangle$ as $\text{opt}_{P,D} = \text{argmax}_{B\in P}\langle B, D\rangle$.

Definition 15. The kernel of a linear surjection $\Phi : \mathbb{R}^{\hat{k}\times\hat{n}} \to \mathbb{R}^{\hat{n}\times\hat{n}}$ is denoted as $K[\Phi] = \{B \in \mathbb{R}^{\hat{k}\times\hat{n}} : \Phi B = 0_{\hat{n}\times\hat{n}} \in \mathbb{R}^{\hat{n}\times\hat{n}}\}$. We denote the orthogonal complement of the kernel of Φ as the set $K^\perp[\Phi] = \{B \in \mathbb{R}^{\hat{k}\times\hat{n}} : \langle B, B'\rangle = 0 \text{ for all } B' \in K[\Phi]\}$.

Theorem 9. *Let $\Phi : \mathbb{R}^{\hat{k}\times\hat{n}} \to \mathbb{R}^{\hat{n}\times\hat{n}}$ be a linear surjection between convex polytopes $P \subset \mathbb{R}^{\hat{k}\times\hat{n}}$ and $\tilde{P} \subset \mathbb{R}^{\hat{n}\times\hat{n}}$. Suppose $B \in P$. Then $\Phi B \in ex(\tilde{P})$ if and only if there exists $D \in K^\perp[\Phi]$ such that $B \in opt_{P,D}$ and Φ maps all the matrices in $opt_{P,D}$ to the same point in $\tilde{P}$.*

Proof. ($\implies$) Since ΦB is an extreme point of $\tilde{P}$, by Lemma 4 there exists $W \in \mathbb{R}^{\hat{n}\times\hat{n}}$ such that $\langle W, \Phi B\rangle > \langle W, T\rangle$ for all $T \in \hat{P} - \{\Phi B\}$. Consider $D = \Phi^\top W \in \mathbb{R}^{\hat{k}\times\hat{n}}$. Then $D \in K^\perp[\Phi]$ because for any $\hat{B} \in K[\Phi]$, $\langle D, \hat{B}\rangle = tr[D^\top \hat{B}] = tr[W^\top(\Phi\hat{B})] = 0$. Also, for any $B' \in P$,

$$\langle D, B'\rangle = tr[D^\top B'] = tr[(\Phi^\top W)^\top B'] = tr[W^\top(\Phi B')] = \langle W, \Phi B'\rangle$$

Then for any $\tilde{B} \in P$, if $\Phi\tilde{B} = \Phi B$, then

$$\langle D, B\rangle = \langle W, \Phi B\rangle = \langle W, \Phi\tilde{B}\rangle = \langle D, \tilde{B}\rangle$$

Alternatively, if $\Phi\tilde{B} \neq \Phi B$, then

$$\langle D, B\rangle = \langle W, \Phi B\rangle > \langle W, \Phi\tilde{B}\rangle = \langle D, \tilde{B}\rangle$$

where the strict inequality follows from the definition of W. Thus $B \in \text{opt}_{P,D}$ and Φ sends all of $\text{opt}_{P,D}$ to ΦB.

($\impliedby$) Suppose there exists $D \in K^\perp[\Phi]$ such that $B \in \text{opt}_{P,D}$ and Φ maps all of $\text{opt}_{P,D}$ to the same point. Define $W \in \mathbb{R}^{\hat{n}\times\hat{n}}$ as $W = (\Phi^\top)^\dagger D$ where $\dagger$ indicates the Moore–Penrose pseudoinverse. For any $B' \in P$,

$$\langle W, \Phi B'\rangle = tr[(\Phi B')^\top W] = tr[(\Phi B')^\top(\Phi^\top)^\dagger D] = tr[B'^\top\Phi^\top(\Phi^\top)^\dagger D]$$

A standard result from linear algebra holds that $\Phi^\top(\Phi^\top)^\dagger$ is a projection matrix onto $K^\perp[\Phi]$. Since D is in $K^\perp[\Phi]$, $\Phi^\top(\Phi^\top)^\dagger D = D$. Hence,

$$\langle W, \Phi B'\rangle = tr[B'^\top\Phi^\top(\Phi^\top)^\dagger D] = tr[B'^\top D] = \langle D, B'\rangle$$

Given some $T \in \tilde{P}$, the surjectivity of Φ guarantees the existence of a $B' \in P$ with $T = \Phi B'$. Either B' is in $\text{opt}_{P,D}$ or it is not. If $B' \in \text{opt}_{P,D}$, since Φ maps all of $\text{opt}_{P,D}$ to the same point, we must have $T = \Phi B$. If $B' \notin \text{opt}_{P,D}$, then $T \neq \Phi B$ so

$$\langle W, T\rangle = \langle D, B'\rangle < \langle D, B\rangle = \langle W, \Phi B\rangle$$

Therefore, there exists a $W \in \mathbb{R}^{\hat{n}\times\hat{n}}$ such that ΦB is the unique maximizer, implying $\Phi B \in ex(\tilde{P})$ by Lemma 4. □

Using Theorem 9, we provide a geometric characterization of extreme points of R_F that are preserved under Ψ by setting $\Phi = \Psi$, $\hat{k} = k$, $\hat{n} = n$, $P = R_F$, and $\tilde{P} = F$.

Corollary 3. *Suppose $B \in R_F$. Then $\Psi B \in ex(F)$ if and only if there exists $D \in K^\perp[\Psi]$ such that $B \in opt_{R_F,D}$ and Ψ maps all the matrices in $opt_{R_F,D}$ to the same point.*

Since the above result holds for any $B \in R_F$, it also holds in particular when $B \in ex(R_F)$.

C Heuristic Constructor Analysis of Section 5

C.1 Runtime Analysis of Algorithm 1

The following proposition shows how Step 8 can be computed efficiently and will help us establish the runtime of Algorithm 1.

Proposition 2. *The value of q in Step 8 of the Algorithm 1 can be computed as $q = \min\left\{q_0, ..., q_{n-1}, \frac{c_j}{zs}\right\}$ where,*

$$q_i = \begin{cases} \frac{e^\epsilon r_{i+1} - r_i}{e^\epsilon s_{i+1} - s_i}, & p_i = 1 \\ \frac{r_i - e^{-\epsilon} r_{i+1}}{s_i - e^{-\epsilon} s_{i+1}}, & p_i = -1 \end{cases}$$

Proof. In Step 8 of the algorithm, for any possible value of γ, $r - \gamma s$ is ϵ-neighbor indistinguishable if and only if for every $0 \leq i \leq n-2$ the following two constraints hold.

$$e^{-\epsilon}(r_{i+1} - \gamma s_{i+1}) \leq r_i - \gamma s_i$$
$$\text{and}$$
$$r_i - \gamma s_i \leq e^\epsilon(r_{i+1} - \gamma s_{i+1})$$

Let $\mathbb{L}_i$ be the set of values for γ for which the first constraint holds and let $\mathbb{U}_i$ be the set of values for γ for which the second constraint holds.

$$\mathbb{L}_i = \left\{\gamma : e^{-\epsilon}(r_{i+1} - \gamma s_{i+1}) \leq r_i - \gamma s_i\right\}$$
$$\mathbb{U}_i = \left\{\gamma : r_i - \gamma s_i \leq e^\epsilon(r_{i+1} - \gamma s_{i+1})\right\}$$

Additionally, let

$$\mathbb{V} = \{\gamma : \gamma z s \leq c_j\} = \left(-\infty, \frac{c_j}{zs}\right)$$

Then the value chosen for q in Step 8 can be written as the maximum of the intersection of all of these sets.

$$q = \max\left\{\mathbb{V} \cap \bigcap_{0\le i\le n-2} \mathbb{L}_i \cap \mathbb{U}_i\right\}$$

The sets $\mathbb{L}_i$ and $\mathbb{U}_i$ can be rewritten as,

$$\mathbb{L}_i = \left\{\gamma : \gamma\left(s_i - e^{-\epsilon}s_{i+1}\right) \le r_i - e^{-\epsilon}r_{i+1}\right\}$$

$$\mathbb{U}_i = \left\{\gamma : \gamma\left(e^{\epsilon}s_{i+1} - s_i\right) \le e^{\epsilon}r_{i+1} - r_i\right\}$$

The right hand side of each inequality above is non-negative by ϵ-neighbor indistinguishability of r. We consider two cases separately: when $p_i = 1$ and when $p_i = -1$.

In the case when $p_i = +1$, the left hand side of the inequality of $\mathbb{L}_i$ is zero, so $\mathbb{L}_i = \mathbb{R}$. Meanwhile, the left hand side of $\mathbb{U}_i$ is γ times a positive quantity, so the inequality can be written as

$$\gamma \le \frac{e^{\epsilon}r_{i+1} - r_i}{e^{\epsilon}s_{i+1} - s_i} = q_i$$

where q_i is defined as in the statement of the lemma. Therefore $\mathbb{U}_i = (-\infty, q_i)$ so $\mathbb{L}_i \cap \mathbb{U}_i = (-\infty, q_i)$.

Alternatively, in the case when $p_i = -1$, the left hand side of the inequality of $\mathbb{U}_i$ is zero, so $\mathbb{U}_i = \mathbb{R}$. Meanwhile, the left hand side of the inequality of $\mathbb{L}_i$ is γ times a positive quantity, so the inequality can be written,

$$\gamma \le \frac{r_i - e^{-\epsilon}r_{i+1}}{s_i - e^{-\epsilon}s_{i+1}} = q_i$$

where q_i is defined as in the statement of the lemma. Therefore $\mathbb{L}_i = (-\infty, q_i)$ so $\mathbb{L}_i \cap \mathbb{U}_i = (-\infty, q_i)$. Therefore, in either case,

$$q = \max\left\{\left(-\infty, \frac{c_j}{zs}\right) \cap \bigcap_{0\le i\le n-2} (-\infty, q_i)\right\} = \min\left\{q_0, ..., q_{n-1}, \frac{c_j}{zs}\right\}$$

completing the proof. □

Observe that each q_i can be computed in constant time, so the value of q in Step 8 can be computed in $O(n)$ time. Using this result, we can now establish the runtime of Algorithm 1 in Theorem 5.

Proof of Theorem 5. First, we show that the inner loop executes a maximum of $2n-1$ times during the entire program execution (and not per outer-loop execution). Our proof proceeds by demonstrating a map from the program state to the non-negative integers that decrements with each inner-loop iteration.

At any point in the execution of the algorithm, define Ω to be the number of columns j with $c_j > 0$, plus the number of rows $0 \le i \le n-2$ such that $r_i \notin \{e^{-\epsilon}r_{i+1}, e^{\epsilon}r_{i+1}\}$. We show that Ω decreases with each pass through the inner loop. Since Ω can be no more than the number of columns plus the number of rows minus 1, this establishes that the algorithm's inner loop executes at most $2n-1$ times.

Once $c_j = 0$ for some $0 \le j \le n-2$, column j will never be selected again by the algorithm, so the bound $c_j = 0$ will continue to hold through the remaining execution. Similarly, once there is a row i with $r_{i+1} = e^{\epsilon}r_i$, pattern p will be set in Step 6 so that $p_i = +1$. Let r' be the next value of variable r assigned in Step 9. Then we have,

$$r'_{i+1} = r_{i+1} - qs_{i+1} = e^{\epsilon}r_i - qe^{\epsilon}s_i = e^{\epsilon}r'_i$$

An analogous result follows when $r_{i+1} = e^{-\epsilon}r_i$, establishing that once a row i meets the ϵ-neighbor-indistinguishability bound, the bound continues to hold through the remaining execution.

Next, we show that in each inner loop iteration, Ω decreases. In one pass of Step 8, either $q = c_j$, in which case, c_j is set to 0 in Step 9, decreasing Ω. Alternately, if $q \ne c_j$, then $r - \gamma s$ is ϵ-neighbor-indistinguishable for $\gamma = q$ but not ϵ-neighbor-indistinguishable for any $\gamma > q$. There must therefore be some position i such that the ratio $\frac{(r-\gamma s)_{i+1}}{(r-\gamma s)_i}$ is within $[e^{-\epsilon}, e^{+\epsilon}]$ for $\gamma \le q$, and outside the interval for $\gamma > q$. Note that the numerator and denominator are both positive as long as $c_j > 0$, because the algorithm always updates each column so that it remains ϵ-neighbor indistinguishable. By the previous argument, position i cannot be one for which the ϵ-neighbor indistinguishability bound already holds. Since the ratio is continuous in γ as long as the ratio is finite, we must have $\frac{(r-\gamma s)_{i+1}}{(r-\gamma s)_i} \in \{e^{-\epsilon}, e^{+\epsilon}\}$ when $\gamma = q$. This establishes that the ϵ-neighbor indistinguishability bound goes from not binding to binding during Step 9, decreasing Ω.

Since Ω decreases with each inner loop iteration and cannot be negative, the algorithm must halt. The number of inner loop iterations is bounded by the maximum possible value of Ω, which is at most $2n-1$ when the algorithm begins.

Next, we show that the outer loop executes a maximum of n times. To see this, note that once a column j is selected in Step 3, the inner loop will continue executing until $c_j = 0$ (by the previous argument, we know the inner loop will halt). At that point, column j will never be selected again, so $c_j = 0$ will continue to hold through the remaining execution. Since each of the n columns can only be selected once, the outer loop executes a maximum of n times.

Applying the above analysis, Table 2 lists the total number of times each step may execute, along with a big-O bound for each time it executes. Note in particular the max repetitions are total for an entire program execution, and not per iteration of any loop. Totaling the values in the table, the runtime of the entire algorithm is $O\left(\max\{\tau n, n^2\}\right)$. □

Step	Max Repetitions	Runtime per Repetition
1	1	$O(n^2)$
2	n	$O(1)$
3	n	$O(\tau)$
4	$2n-1$	$O(1)$
5	$2n-1$	$O(n)$
6	$2n-1$	$O(n)$
7	$2n-1$	$O(n)$
8	$2n-1$	$O(n)$
9	$2n-1$	$O(n)$
10	$2n-1$	$O(1)$
11	n	$O(1)$
12	1	$O(1)$

Table 2: Total execution time attributable to each step of Algorithm 1.

C.2 Extreme Point Analysis of Algorithm 1

Given inputs z, ϵ, and κ to Algorithm 1, we denote the output as $O[z, \epsilon, \kappa] \in \mathbb{R}^{n\times n}$. For readability, we will omit the inputs and

simply refer to this matrix as O. In Theorem 6 below, we prove that O is an extreme point of F.

To do so, we first show in Lemma 5 that the probability placed on the diagonal by Algorithm 1 is maximal in a particular sense. Next, we consider the set of all possible vectors that could be valid choices for a column at a particular point in algorithm execution. We show in Lemma 6 that this set is a polytope and that the vector selected by the algorithm is one of its extreme points. Finally, we show that if an algorithm sequentially fills each column with a vector that is extreme in this sense, then O is an extreme point of F, thereby finishing the proof of Theorem 6.

In the execution of the algorithm, the function κ will select columns in a particular sequence. For a column j, let $\text{prev}(j)$ denote the set of all columns chosen before j, and $\text{post}(j)$ denote the set of all columns chosen after j.

Definition 16. Given column $0 \le j \le n-1$, let F_j be the set of matrices $O' \in F$ such that $O'_l = O_l$ for all $l \in \text{prev}(j)$.

By construction, when j is the first column selected, $F_j = F$, and when j is the last column selected, $F_j = \{O\}$. Moreover, for any columns j and $j' \in \text{post}(j)$, $F_j \supseteq F_{j'}$.

Notice that each F_j is formed by adding linear constraints to F, so it is also a convex polytope. The following lemma shows that each time the algorithm fills in a column, it puts the maximum possible probability on the diagonal, given all the previously filled-in columns.

Lemma 5. *For any $0 \le j \le n-1$ and for any $O' \in F_j$ such that $O'_j \ne O_j$, we have $o_{j,j} > o'_{j,j}$.*

Proof. Assume for contradiction that $o_{j,j} \le o'_{j,j}$. Since $O'_j \ne O_j$, there must be some row $0 \le i \le n-1$ such that $o'_{i,j} < o_{i,j}$ (otherwise, we would have $o'_{i,j} \ge o_{i,j}$ for all rows i, with strict inequality in at least one row; this implies $zO'_j > zO_j = z_j$, so $O' \notin F$, contradicting $O' \in F_j \subseteq F$).

We consider three possible cases for i.

Case 1: Consider $i = j$. Then $o_{j,j} > o'_{j,j}$, yielding an immediate contradiction.

Case 2: Consider $i < j$. Since the condition $o'_{i',j} < o_{i',j}$ holds for $i' = i$ but not for $i' = j$, there must be some $i \le \hat{i} < j$ such that the condition holds for $i' = \hat{i}$ but not for $i' = \hat{i}+1$. That is,

$$o'_{\hat{i},j} < o_{\hat{i},j} \text{ and } o'_{\hat{i}+1,j} \ge o_{\hat{i}+1,j}$$

There are two subcases to consider.

Subcase 1: Suppose $o_{\hat{i},j} = e^{-\epsilon} o_{\hat{i}+1,j}$. Then $o'_{\hat{i},j} < o_{\hat{i},j} = e^{-\epsilon} o_{\hat{i}+1,j} \le e^{-\epsilon} o'_{\hat{i}+1,j}$, contradicting the ϵ-neighbor indistinguishability of O'_j.

Subcase 2: Suppose $o_{\hat{i},j} > e^{-\epsilon} o_{\hat{i}+1,j}$. Then the scale variable s that is added to column j during Step 9, cannot always have $s_{\hat{i}} = e^{-\epsilon} s_{\hat{i}+1}$. This means that at some point during the outer loop iteration for column j, the value of pattern variable p at the end of Step 6 had $p_{\hat{i}} = -1$, which can only happen if the variable r fulfills $r_{\hat{i}} = e^{\epsilon} r_{\hat{i}+1}$. The proof of Theorem 5 establishes that once the remainder fulfills this bound, it always fulfills the bound.

Let $\hat{r}$ be the value of variable r at the start of the outer loop iteration for column j. Then, the value at the end of the outer loop iteration would be given by $\hat{r} - O_j$, and the bound above can be written as

$$\hat{r}_{\hat{i}} - o_{\hat{i},j} = e^{\epsilon}(\hat{r}_{\hat{i}+1} - o_{\hat{i}+1,j})$$

Since O is an output of our algorithm, we also have

$$\hat{r} = \mathbb{1}_n - \sum_{\bar{j} \in \text{prev}(j)} O_{\bar{j}}$$

Furthermore, since the rows of O' must sum to 1, we can decompose O' as follows.

$$\begin{aligned}\mathbb{1}_n &= \sum_{\bar{j} \in \text{prev}(j)} O'_{\bar{j}} + O'_j + \sum_{\tilde{j} \in \text{post}(j)} O'_{\tilde{j}} \\ &= \sum_{\bar{j} \in \text{prev}(j)} O_{\bar{j}} + O'_j + \sum_{\tilde{j} \in \text{post}(j)} O'_{\tilde{j}}\end{aligned}$$

where the second equality follows from $O' \in F_j$. Combining the prior two equalities yields

$$\hat{r} - O'_j = \sum_{\tilde{j} \in \text{post}(j)} O'_{\tilde{j}}$$

The right-hand side is a sum of $\epsilon-$neighbor indistinguishable columns, so must be $\epsilon-$neighbor indistinguishable. However, the left-hand side is not ϵ-neighbor indistinguishable because,

$$\begin{aligned}(\hat{r} - O'_j)_{\hat{i}} &= \hat{r}_{\hat{i}} - o'_{\hat{i},j} > \hat{r}_{\hat{i}} - o_{\hat{i},j} \\ &= e^{\epsilon}(\hat{r}_{\hat{i}+1} - o_{\hat{i}+1,j}) \\ &\ge e^{\epsilon}(\hat{r}_{\hat{i}+1} - o'_{\hat{i}+1,j}) \\ &= e^{\epsilon}(\hat{r} - O'_j)_{\hat{i}+1}\end{aligned}$$

This yields a contradiction.

Since all subcases yield a contradiction, we deduce a contradiction for the entire case.

Case 3: Consider $i > j$. This case is also yields a contradiction. It is analogous to the prior case, and is hence omitted.

Since each case yields a contradiction, we conclude $o_{j,j} > o'_{j,j}$. □

Next, we consider the set of all possible vectors that could be inserted into column j when the algorithm reaches the column. This is defined formally below.

Definition 17. Given column $0 \le j \le n-1$, let Γ_j be the set of entries for column j in F_j; $\Gamma_j = \{O'_j : O' \in F_j\}$.

Notice that Γ_j is a linear projection of F_j so it is also a convex polytope. We show the column O_j chosen by the algorithm is an extreme point of Γ_j.

Lemma 6. *Given column $0 \le j \le n-1$, $O_j \in ex(\Gamma_j)$.*

Proof. Consider any $v \in \Gamma_j$, where $v \ne O_j$. Then $v = O'_j$ for some $O' \in F_j$, and since $v \ne O_j$, $O'_j \ne O_j$, By the Lemma 5, we have $o_{j,j} > v_j$. Let $u \in \mathbb{R}^n$ be the column vector with $u_j = 1$, $u_{j'} = 0$ for all $j' \ne j$. Then $u^\top O_j = o_{j,j} > v_j = u^\top v$. Then O_j is the unique maximizer of the linear objective u in Γ_j, so it is an extreme point. □

Finally, the last step is to show that if an algorithm always fills each column j with a vector that is extreme in the sense above, the overall matrix will also be an extreme point of F.

Proof of Theorem 6. First, we show $O \in F$. First, note that local variables A, r, and c are only updated in Step 9. This step ensures that the conditions $A\mathbb{1}_n + r = \mathbb{1}_n$ and $zA + c = z$ are maintained. Since the algorithm continues until $r = 0_n$ and $c = 0_n^\top$, the final value of A fulfills the conditions $A\mathbb{1}_n = \mathbb{1}_n$ and $zA = z$. Additionally, the update in Step 9 proceeds by adding a positive multiple of an ϵ-scale to a column of A. At every point of execution, every column of A is a conic combination of ϵ-scales, and is thus ϵ-neighbor indistinguishable. Since O is the final value of A, we have $O \in F$.

Next, we show $O \in ex(F)$. Assume for contradiction that there exists $0 < \alpha < 1$ and distinct matrices $X, Y \in F$ such that $\alpha X + (1-\alpha)Y = O$. Then there must be at least one column in which X and Y differ. Let j be the first column chosen by the algorithm for which $X_j \neq Y_j$. Then $X, Y \in F_j$ so $X_j, Y_j \in \Gamma_j$. Moreover $\alpha X_j + (1-\alpha)Y_j = O_j$, contradicting $O_j \in ex(\Gamma_j)$. □

D Characterization of Unfixed Polytope U

In this Appendix section, we turn our attention to the general problem of describing extreme points of U (including extreme points that include zero columns). Similar to $ex(F)$, it is possible to characterize $ex(U)$ using a framework that is based on ϵ-scales. We will see that extreme points may be built from conic combinations of scales that are linearly independent of each other in a particular sense (Definition 19). To make this explicit, we will define a new polytope R_U which encodes how much of each scale contributes to every column of $T \in U$.

D.1 Representing U with ϵ-Scales

We now define the unfixed representation polytope R_U. Analogous to R_F, for a representation matrix $B \in R_U$, $b_{i,j}$ quantifies the amount of scale i that enters column j of T.

Definition 18. We define the *unfixed representation polytope* R_U as

$$R_U = \left\{B \in \mathbb{R}_{\geq 0}^{k \times n} : \Psi B \mathbb{1}_n = \mathbb{1}_n\right\}$$

The following theorem will show that every matrix in U can be reached from a representation matrix in R_U under this linear map.

Theorem 10. *Ψ is an affine surjection from R_U to U.*

Proof. Since Ψ is a linear map, it is affine, and hence it only remains to show $\Psi(R_U) = U$.

Suppose $T \in \Psi(R_U)$. Then there is some $B \in R_U$ such that $\Psi B = T$. By definition of R_U, $T\mathbb{1}_n = \Psi B \mathbb{1}_n = \mathbb{1}_n$. Column j of T is given by ΨB_j. This is a conic combination of ϵ-neighbor indistinguishable vectors, and is hence ϵ-neighbor indistinguishable. Since every column of T is ϵ-neighbor indistinguishable, T is ϵ-differentially private, establishing $T \in U$.

Conversely, suppose $T \in U$. By Lemma 1, every column j of T can be written as a conic combination of ϵ-scales $T_j = \Psi\beta^j$ for some $\beta^j \in \mathbb{R}_{\geq 0}^k$. Construct the $n \times k$ matrix B where the j^{th} column of B is β^j. Then $T = \Psi B$. Furthermore, $\Psi B \mathbb{1}_n = T\mathbb{1}_n = \mathbb{1}_n$, so $B \in R_U$. Therefore, $T \in \Psi(R_U)$. Hence, $\Psi(R_U) \supseteq U$, producing $\Psi(R_U) = U$. □

In Proposition 3, we show the linear map Ψ also provides a relation between the extreme points of U and the extreme points of R_U.

Proposition 3. *For all $T \in ex(U)$ there exists $B \in ex(R_U)$ such that $T = \Psi B$.*

The proof of Proposition 3 is very similar to that of Proposition 1, as is thus omitted. Hence, every extreme point in U can be written as an extreme point in R_U under the mapping Ψ.

However, as we discovered when Ψ represented matrices in F using R_F, this representation is not unique for matrices in U. As demonstrated in Example 2 below, it is possible for multiple extreme points of R_U to map to the same extreme point in U, and it is possible for an extreme point of R_U to map to a non-extreme point of U under Ψ.

Example 2. Ψ is not a 1-1 mapping between $ex(R_U)$ and $ex(U)$. Similar to Example 1, consider the case when $\epsilon = \ln(2)$ and $n = 3$. There are 4 possible ϵ-scales, presented as

$$\Psi = \begin{bmatrix} 4/7 & 2/5 & 1/4 & 1/7 \\ 2/7 & 1/5 & 1/2 & 2/7 \\ 1/7 & 2/5 & 1/4 & 4/7 \end{bmatrix}$$

Using the Avis-Fukuda vertex enumeration algorithm, we generate all extreme points of U and R_U. We find $|ex(U)| = 27$ and $|ex(R_U)| = 36$. In particular, both matrices $B^{(4)}$ and $B^{(5)}$ are extreme points of R_U, yet they both map to the same extreme point in U under Ψ.

$$B^{(4)} = \begin{bmatrix} 7/6 & 0 & 0 \\ 0 & 0 & 0 \\ 2/3 & 0 & 0 \\ 7/6 & 0 & 0 \end{bmatrix} \in ex(R_U)$$

$$B^{(5)} = \begin{bmatrix} 0 & 0 & 0 \\ 5/3 & 0 & 0 \\ 4/3 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \in ex(R_U)$$

$$\Psi B^{(4)} = \Psi B^{(5)} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \in ex(U)$$

Additionally, the mapping Ψ may send extreme points in R_U to non-extreme points in U. For example,

$$B^{(6)} = \begin{bmatrix} 7/6 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2/3 \\ 7/6 & 0 & 0 \end{bmatrix} \in ex(R_U)$$

yet $\Psi B^{(6)} \notin ex(U)$, as

$$\Psi B^{(6)} = \begin{bmatrix} 5/6 & 0 & 1/6 \\ 4/6 & 0 & 2/6 \\ 5/6 & 0 & 1/6 \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} 11/12 & 0 & 1/12 \\ 10/12 & 0 & 2/12 \\ 11/12 & 0 & 1/12 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 9/12 & 0 & 3/12 \\ 6/12 & 0 & 6/12 \\ 9/12 & 0 & 3/12 \end{bmatrix}$$

△

However, we can apply Theorem 9 to characterize the necessary and sufficient conditions for which an extreme point of R_U is also an extreme point of U under Ψ (as we did in Corollary 3).

Corollary 4. *Suppose $B \in R_U$. Then $\Psi B \in ex(U)$ if and only if there exists $D \in K^{\perp}[\Psi]$ such that $B \in opt_{R_U,D}$ and Ψ maps all the matrices in $opt_{R_U,D}$ to the same point.*

Since the above result holds for any $B \in R_U$, it holds in particular when $B \in ex(R_U)$.

D.2 Characterizing the Extreme Points of R_U

Next, we establish necessary and sufficient conditions for points in R_U to be extreme, as as we did for R_F in Theorem 4. As we will see, an analogous theorem does hold, but the removal of a fixed-point constraint alters the affinely simplified condition. Whereas the condition of affinely simplified involved linear independence of weighted differences of scales, we will see that the unfixed case involves linear independence of scales directly.

Given a column vector $x \in \mathbb{R}^k$, let $G(x)$ be the set of scales with positive coefficients in the corresponding positions of x, $G(x) = \{\Psi_u : x_u > 0\}$.

Definition 19. We call a $k \times n$ matrix B $\Psi-$*linearly simplified* if the additive union $\uplus_{j=0}^{n-1} G(B_j)$ is linearly independent.

Let $M_U(\Psi)$ be the set of $k \times n$ matrices μ such that $\Psi\mu\mathbb{1}_n = 0_n$. By construction of $M_U(\Psi)$, if $X, Y \in R_U$, then $X - Y \in M_U(\Psi)$. This suggests that M_U represents "movement matrices" that can potentially be added to matrices in R_U to move through the affine space. The following lemma and theorem formalize the equivalence between the extreme points of R_U, the notion of Ψ-linearly simplified matrices, and these movement matrices.

Lemma 7. *Suppose $B \in R_U$. Then B is $\Psi-$linearly simplified if and only if there does not exist a non-zero matrix $\mu \in M_U(\Psi)$ such that $\mu_{i,j}$ is zero whenever $b_{i,j}$ is zero.*

Proof. ($\Longrightarrow$) We proceed by the contrapositive. Suppose there exists a non-zero movement matrix $\mu \in M_U(\psi)$ such that $\mu_{i,j}$ is zero whenever $b_{i,j}$ is zero.

For column j, let $P(j)$ be the set of row indices of B_j that are non-zero. Then $G(B_j) = \{\Psi_p : p \in P(j)\}$. We can then write,

$$\Psi\mu_j = \sum_{p \in P(j)} \Psi_p \mu_{p,j}$$

So then

$$0_n = \Psi\mu\mathbb{1}_n = \sum_{j=0}^{n-1} \Psi\mu_j = \sum_{j=0}^{n-1} \sum_{p \in P(j)} \Psi_p \mu_{p,j}$$

Since $\mu \neq 0_{k \times n}$, and the nonzero entries in μ_j can only occur in rows $P(j)$, some of the coefficients in the sum above must be nonzero. Thus, the above is a linear dependency among the vectors of the multiset $\uplus_{j=0}^{n-1} G(B_j)$, implying that B is not $\Psi-$linearly simplified, as required.

($\Longleftarrow$) Again by the contrapositive, suppose B is not Ψ-linearly simplified. Then $\uplus_{j=0}^{m-1} G(B_j)$ is linearly dependent. The linear dependency can be written as,

$$\sum_{i,j} \mu_{i,j} \Psi_i = 0_n$$

for a collection of $\mu_{i,j}$'s, not all zero, where $\mu_{i,j} \neq 0$ only when $b_{i,j} > 0$, and equals 0 otherwise. Let μ be the matrix with value $\mu_{i,j}$ in position (i, j). Then the sum above can be written in matrix notation as $\Psi\mu\mathbb{1}_n = 0_n$. Therefore $\mu \in M_U(\Psi)$. □

Theorem 11. *Suppose $B \in R_U$. Then $B \in ex(R_U)$ if and only if B is $\Psi-$linearly simplified.*

Proof. ($\Longrightarrow$) We proceed by the contrapositive. Suppose B is not Ψ-linearly simplified. By the equivalence of Ψ-linearly simplified and movement matrices established in Lemma 7, there exists a movement matrix $\mu \in M_U(\Psi)$ such that $\mu_{i,j} = 0$ whenever $b_{i,j} = 0$. Choose any positive

$$\delta < \min_{\{(i,j) \mid \mu_{i,j} \neq 0\}} \left\{ \frac{b_{i,j}}{|\mu_{i,j}|} \right\}$$

Then $B \pm \delta\mu \in R_U$, as

(1) For any i, j entry, $\pm\delta\mu_{i,j} \leq b_{i.j}$ with equality if and only if $b_{i,j} = 0$. Hence $(B \pm \delta\mu)_{i,j} \geq 0$.
(2) $\Psi(B \pm \delta\mu)\mathbb{1}_n = \Psi B\mathbb{1}_n \pm \delta\Psi\mu\mathbb{1}_n = \mathbb{1}_n \pm 0_n = \mathbb{1}_n$

But then

$$B = \frac{1}{2}(B + \delta\mu) + \frac{1}{2}(B - \delta\mu)$$

Therefore $B \notin ex(R_U)$.

($\Longleftarrow$) Proceed by the contrapositive yet again. Suppose $B \notin ex(R_U)$. Then there exists distinct $X, Y \in R_U$ and $\theta \in (0, 1)$ such that $B = \theta X + (1-\theta)Y$. By non-negativity, the convex combination above implies $x_{i,j} = 0 = y_{i,j}$ whenever $b_{i,j} = 0$. Let $\mu = X - Y$. Then $\mu_{i,j} = 0$ whenever $b_{i,j} = 0$. Also $\Psi\mu\mathbb{1}_n = \Psi X\mathbb{1}_n - \Psi Y\mathbb{1}_n = \mathbb{1}_n - \mathbb{1}_n = 0_n$. So $\mu \in M_U(\Psi)$, implying B is not Ψ-linearly simplified. □

Speaking intuitively, Theorem 11 says that an extreme point of U can be formed by summing multiples of linearly independent ϵ-scales, each one contributing to a unique column. It is useful to make two points of comparison. Since at most n ϵ-scales can be linearly independent, if we add the condition that all n columns are non-zero, by the pigeonhole principle each column must have a multiple of a single ϵ-scale, so we immediately recover Theorem 2.

In a related study, Ghosh et al. focus on finding an optimal point in U under a natural class of linear objective functions [21] that we call row-wise concentrating (Definition 11). In Appendix E, we explain that the result of Ghosh et al. can be seen as a further restriction on B in Theorem 11.

One combinatorial comparison can be made between the extreme points of R_F and the extreme points of R_U. In the case of an extreme point of U, the condition of Ψ-linearly simplified implies there must be at most n positive matrix entries. Additionally, any row of $B \in ex(R_U)$ cannot have more than 1 positive entry – otherwise, the same same scale would appear twice in the multiset $\uplus_{j=0}^{m-1} G(B_j)$, contradicting Ψ-linear-simplified. This analysis is collected in the following corollary.

Corollary 5. *Extreme matrices $B \in ex(R_U)$ contain at least 1 and at most n positive entries. Also, such matrices B have at most 1 positive entry in every row.*

E Two-Stage Unfixed Optimum Constructor Analysis of Section 5

In this section, we establish the key properties of the two-staged unfixed optimum constructor. In order to provide a fair comparison for fixed-point constructors, our design approach includes making reasonable assumptions where possible to enable efficient execution.

Our algorithm leverages a key result from Ghosh et al. [21], who study count mechanisms that optimize a natural class of count error measures which we call *row-wise concentrating* (Definition 11).

Ghosh et al. show that any count mechanism that optimizes a row-wise concentrating objective is equivalent to the truncated geometric mechanism with post-processing. We will restate their result in terms of our framework in this section.

Recall from the main text that Σ denotes the matrix containing all single-peaked scales in order. That is, Σ_l is the single-peaked scale at position l. Also recall from Appendix D.2 that given a column vector $x \in \mathbb{R}^k$, $G(x)$ is defined to be the set of scales with positive coefficients in the corresponding positions of x; that is, $G(x) = \{\Psi_u : x_u > 0\}$.

Definition 20. Let $\mathcal{T}$ be the set of count mechanisms T such that $T = \Psi B$ for $B \in R_U$ such that $\uplus_{j=0}^{n-1} G(B_j)$ is the set of all single-peaked scales, $\{\Sigma_l\}_{l=0}^{n-1}$.

Let ω_l be the entry of B that indicates the amount of scale Σ_l.

Observation 3. *The row constraint of R_U gives*

$$\Psi B \mathbb{1}_n = \sum_{l=0}^{n-1} \Psi \omega_l \Sigma_l = \mathbb{1}_n$$

It is straightforward to check that the single-peaked scales are linearly independent, and therefore there is a unique set of values for the ω_l, which can be computed as follows.

$$\omega_l = \begin{cases} \frac{1-e^{-\epsilon}}{1+e^{\epsilon}} \sum_{i=0}^{n-1} e^{-\epsilon|i-l|}, & 0 < l < n-1 \\ \frac{1-e^{-\epsilon}}{(1+e^{\epsilon})(1-e^{\epsilon})} \sum_{i=0}^{n-1} e^{-\epsilon|i-l|}, & l \in \{0, n-1\} \end{cases}$$

Note that the ω_l do not depend on the specific $T \in \mathcal{T}$. Given this notation, a central result from Ghosh et al. can be stated as follows.

Theorem 12. *(Adapted from Ghosh et al. [21]) If T is an optimal count mechanism for row-wise concentrating weight matrix W, then $T \in \mathcal{T}$.*

Given $T \in \mathcal{T}$, suppose $B, B' \in ex(R_U)$ with $\Psi B = \Psi B' = T$ and $\uplus_{j=0}^{n-1} G(B_j) = \uplus_{j=0}^{n-1} G(B'_j)$ is the set of all single-peaked scales. For each column j, $\Psi(B_j - B'_j) = T_j - T_j = 0_n$. The left hand side is a linear combination of the single-peaked scales, but these are linearly independent. Hence $B_j = B'_j$. We can therefore define $\text{col}_T(l)$ to be the column of B with a positive coefficient associated with Σ_l.

Using this notation, the count error associated with weight matrix W can be written,

$$\langle W, T \rangle = \langle W, \Psi B \rangle = \sum_{l=0}^{n-1} \omega_l (W_{col_T(l)})^\top \Sigma_l$$

We next observe that the scales that are part of the optimum are ordered according to where their peaks are. This is made precise in the following lemma.

Lemma 8. *Given row-wise concentrating weight matrix W, and $0 \le l < l' \le n-1$, and c is is the largest value of $\bar{c} \in \{0, ..., n-1\}$ that minimizes $(W_{\bar{c}})^\top \Sigma_l$, and c' is the largest value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma_{l'}$, then $c' \ge c$.*

Proof. We show that the claim holds when $l' = l+1$, as repeating the argument shows that the claim holds for any $l' > l$.

Suppose c is the largest value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma_l$. If $c = 0$, then the claim follows trivially. If $c > 0$, because it is a value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma_l$, setting $\bar{c}$ to $c-1$ cannot decrease this quantity, so

$$(W_{c-1})^\top \Sigma_l \ge (W_c)^\top \Sigma_l$$

Equivalently, $(W_{c-1} - W_c)^\top \Sigma_l \ge 0$. Let $\Delta w_i = w_{i,c-1} - w_{i,c}$. We can rewrite this inequality as

$$\sum_{i=1}^{l} (\Delta w_i)\sigma_{i,l} + \sum_{i=l+1}^{n-1} (\Delta w_i)\sigma_{i,l} \ge 0 \qquad (\star)$$

To prove the claim, we consider two cases: $c \le l$ or $c > l$.

First consider $c > l$. Then the Δw_i in the first sum are all non-positive (because $|c-i| = c-i > c-1-i = |(c-1)-i|$ and row-wise concentrating implies $w_{i,c} \ge w_{i,c-1}$), so the first sum is non-positive. Thus, the second sum must be non-negative for the inequality to hold.

Note that for $i \le l$, $\sigma_{i,l+1} = e^{i\epsilon}\sigma_{0,l+1} = \alpha e^{i\epsilon}\sigma_{0,l} = \alpha\sigma_{i,l}$, where $\alpha = \sigma_{0,l+1}/\sigma_{0,l} < 1$. Similarly, for $i \ge l+1$, $\sigma_{i,l+1} = \beta\sigma_{i,l}$, where $\beta = \sigma_{n-1,l+1}/\sigma_{n-1,l} > 1$. Then

$$\begin{aligned}(W_{c-1} - W_c)^\top \Sigma_{l+1} &= \sum_{i=1}^{n-1} (\Delta w_i)\sigma_{i,l+1} \\ &= \alpha \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} + \beta \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l}\end{aligned}$$

Since $\alpha < 1$, and the first sum is non-positive, we have $\alpha \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} \ge \sum_{i=1}^{l} \Delta w_i \sigma_{i,l}$. Since $\beta > 1$, and the second sum is non-negative, we have $\beta \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l} \ge \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l}$. Thus,

$$\begin{aligned}(W_{c-1} - W_c)^\top \Sigma_{l+1} &\ge \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} + \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l} \\ &= (W_{c+1} - W_c)^\top \Sigma_l \ge 0\end{aligned}$$

Thus, moving Σ_{l+1} from column c to column $c-1$ cannot decrease error; hence if c' is the rightmost value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma'_l$, we must have $c' \ge c$.

Next, consider the case when $c \le l$. Once again, consider the inequality in ($\star$). In the second sum the Δw_i are all non-negative (because $|c-i| = i-c < i-(c-1) = |(c-1)-i|$ and row-wise concentrating implies $w_{i,c} \le w_{i,c-1}$).

If the first sum is also non-negative, then both $\alpha \sum_{i=1}^{l} \Delta w_i \sigma_{i,l}$ and $\beta \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l}$ are non-negative so their sum is non-negative.

On the other hand, if the first sum is negative, then because $\alpha < 1$, $\alpha \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} > \sum_{i=1}^{l} \Delta w_i \sigma_{i,l}$. Because $\beta > 1$, $\beta \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l} \ge \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l}$. Thus $\alpha \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} + \beta \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l} > \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} + \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l} \ge 0$.

In either case, we have

$$(W_{c-1} - W_c)^\top \Sigma_{l+1} = \alpha \sum_{i=1}^{l} \Delta w_i \sigma_{i,l} + \beta \sum_{i=l+1}^{n-1} \Delta w_i \sigma_{i,l} \ge 0$$

So once again, moving Σ_{l+1} from column c to column $c-1$ cannot decrease error, implying $c' \geq c$. □

As noted in the main text, we further restrict attention to weight matrices that are also *row-wise convex*. The following shows that for such a weight matrix, the error corresponding to a single-peaked scale is convex in the index of the column in which it is placed.

Lemma 9. *Let W be a row-wise convex weight matrix. For any scale s, the inner product $(W_j)^\top s$ is convex in j.*

Proof. Note that $(W_{j+1})^\top s - (W_j)^\top s = \sum_{i=0}^{n-1}(w_{i,j+1} - w_{i,j})s_i$ is the sum of terms that are non-decreasing in j, so the total is non-decreasing in j. Hence $(W_j)^\top s$ is convex in j. □

Our two-staged unfixed optimum constructor (Algorithm 2) leverages the results of Lemma 8 and Theorem 12 above. It begins by considering the single-peaked scale at position 0. To find the appropriate column for this scale, it begins at column 0 and scans to the right. Lemma 9 tells us that the associated cost is convex in the column, so once we find a local optimum, we know that it is a global optimum. In particular, this is true for the rightmost local optimum. The algorithm then moves on to the next single-peaked scale. By Lemma 8, we know that the rightmost optimum position for this scale cannot be to the left of the previous one, so the algorithm does not need to return to column 0, but can continue scanning from the current column. The process repeats until all single-peaked scales are assigned to columns.

We are now ready to establish the runtime of Algorithm 2.

Proof of Theorem 7. For each iteration of the while loop, either l increases by 1 or j increases by 1. Both variables start at 0, and cannot be exceed $n-1$ at the start of a loop iteration – in particular, once l is assigned the value n, the algorithm will exit the while loop; if j ever has the value of $n-1$ at the start of the loop, then *next_error* will be assigned the value of ∞, guaranteeing that the algorithm does not enter the else clause in which j would be incremented. This guarantees that the while loop can execute no more than $2n$ times.

Within the loop, there are two inner products that require $O(n)$ to complete, and all other instructions are assumed atomic. Thus, the loop requires $O(n^2)$ steps. Initializing the matrices and $\{\omega_l\}_{l=0}^{n-1}$ in Steps 1 and 2 also requires $O(n^2)$. Hence, the total runtime is $O(n^2)$. □

Lastly, we prove that Algorithm 2 outputs a count mechanism that minimizes count error.

Proof of Theorem 8. First we show that $O \in \mathcal{T}$. To do so, we consider the theoretical B associated with each step of our algorithm. In Step 1, initialize $k \times n$ matrix B of all zeroes. In Step 11, set $B_{\hat{i},j} \leftarrow B_{\hat{i},j} + \omega_l$ where $\hat{i}$ is the column of Ψ equal to Σ_l. Then the algorithm maintains $A = \Psi B$ at every point in execution. Furthermore, B is formed by placing each single-peaked scale into a unique column, so the multiset $\uplus_{l=0}^{n-1} G(B_j) = \{\Sigma_l\}_{l=0}^{n-1}$. Thus $O \in \mathcal{T}$.

Next we show that O is optimal for W. Since $O \in \mathcal{T}$, we can write,

$$\langle W, O\rangle = \sum_{l=0}^{n-1} \omega_l (W_{\mathrm{col}_O(l)})^\top \Sigma_l$$

By Theorem 12, any optimal T is in $\mathcal{T}$, and so $\langle W, T\rangle$ can be similarly written as $\sum_{l=0}^{n-1} \omega_l (W_{\mathrm{col}_T(l)})^\top \Sigma_l$. Thus, it suffices to show that $(W_{\mathrm{col}_O(l)})^\top \Sigma_l \leq (W_{\mathrm{col}_T(l)})^\top \Sigma_l$ for all l; that is, $\mathrm{col}_O(l)$ is a value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma_l$.

Examining the while loop, variable l records which single-peaked scale is being placed, and variable j represents a potential column to place the current single-peaked scale. l is initialized to zero and does not change until after scale Σ_0 is placed into a column in Step 11. While l is zero, j begins at 0 and increments 1 at a time, scanning columns from left to right. At Step 4 of any loop iteration, *current_error* is assigned the value $(W_j)^\top \Sigma_0$. At Step 6, *next_error* is assigned the value $(W_{j+1})^\top \Sigma_0$ (or ∞ if j has reached the last column). The condition *next_error* > *current_error* indicates that incrementing j any further would increase $(W_j)^\top \Sigma_0$ (or that we have reached the last possible column). This implies that at this point in execution, j is the right-most value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma_0$.

The argument for single-peaked scales at position $l > 0$ is the same, except that the scan begins with j equal to the column in which scale Σ_{l-1} was placed. The scan will still locate the largest value of $\bar{c}$ that minimizes $(W_{\bar{c}})^\top \Sigma_l$ because Lemma 8 guarantees that this value cannot be less than that corresponding to the previous scale. □

F Development of Rule of Thumb for Privacy Budget Allocation in Section 6

In the two-stage fixed-point framework, the total privacy budget ϵ_t must be divided between ϵ_1 for the distribution privatizer, and ϵ_2 for the privatization of the table of counts. We let $f = \epsilon_1/\epsilon_t$. As noted in Section 6, to set f, we consider a combined error objective that combines accuracy of counts and accuracy of distribution.

$$\ln(\text{count\_error}(f)) + \ln(\text{distribution\_error}(f))$$

Let $f_\star$ be a value of f that minimizes the above objective. As noted in the main text, deviating from $f_\star$ cannot decrease either type of error without also causing an equal or great increase in the other type of error (in a multiplicative sense). To see why, consider an f' that decreases count error by a factor of a, and increases distribution error by a factor of b. By optimality,

$$\begin{aligned}&\ln(\text{count\_error}(f_\star)) + \ln(\text{distribution\_error}(f_\star))\\ &\leq \ln(\text{count\_error}(f')) + \ln(\text{distribution\_error}(f'))\end{aligned}$$

The right-hand side is

$$\begin{aligned}&\ln(a^{-1}\cdot \text{count\_error}(f_\star)) + \ln(b\cdot \text{distribution\_error}(f_\star))\\ &= \ln(\text{count\_error}(f_\star)) + \ln(\text{distribution\_error}(f_\star)) - a + b\end{aligned}$$

Canceling terms, we deduce $b \geq a$, as claimed.

One might hope to use $f_\star$ to split the total privacy budget in practice, but it turns out that $f_\star$ is data-dependent. To approximate $f_\star$, one could consider privately testing different values of f on the real data – however, this would require considerable privacy budget. While it may be possible to improve existing algorithms to select $f_\star$ with less privacy cost (e.g., adapting methods from differentially private budget selection [26]), we defer this possibility to future research. For the current work, we propose an alternate approach:

ϵ_t	Uniform	Left-skewed	Right-skewed	Bimodal symmetric	Zero-inflated	Top-inflated
0.10	0.8	0.4	0.5	0.7	0.3	0.4
0.15	0.8	0.3	0.4	0.8	0.3	0.3
0.22	0.4	0.3	0.3	0.5	0.3	0.3
0.32	0.3	0.3	0.2	0.4	0.2	0.3
0.48	0.3	0.2	0.2	0.3	0.2	0.3
0.71	0.2	0.2	0.2	0.3	0.2	0.2
1.05	0.2	0.1	0.2	0.1	0.1	0.2
1.55	0.1	0.1	0.1	0.1	0.1	0.1
2.29	0.1	0.1	0.1	0.1	0.1	0.1
3.38	0.1	0.1	0.1	0.1	0.1	0.1
5.00	0.1	0.1	0.1	0.1	0.1	0.1

Table 3: Optimal values of f for each of our synthetic datasets.

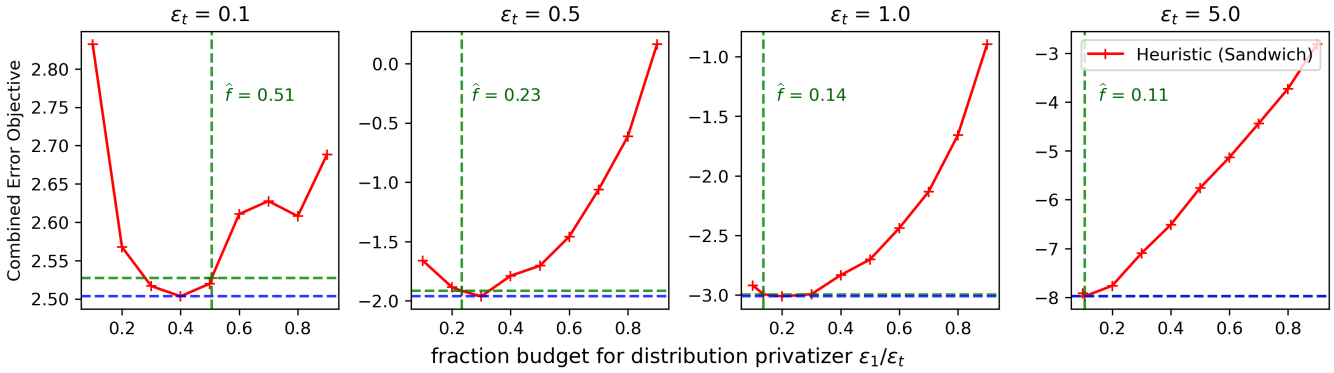

Figure 8: Combined error objective as a function of f for the crime dataset. The vertical green line displays the value of $\hat{f}$ selected by our rule of thumb. The horizontal green line depicts that value of the objective resulting from the rule of thumb. Finally, the horizontal blue line depicts the minimal objective value measured across all parameter settings.

a *rule of thumb* that practitioners can use to set f that does not depend on the data.

At a high level, our approach to attain a rule of thumb proceeds as follows. We begin by constructing six new synthetic datasets, chosen to have a wide range of distributional features. We then perform simulations for each dataset to determine the optimal f for a range of total privacy budgets. We fit a parametric model to this data, resulting in a function that provides a recommended f as a function of ϵ_t. Finally, we assess the performance of our rule of thumb by applying it to the three datasets used in the main text.

Our rule of thumb is fitted using six new synthetic datasets. One reason we created new data for this exercise is to capture a wide range of distributional features that may be observed in real-world applications. A second reason is that we would like our three canonical datasets to serve as a "holdout" to assess the performance of our rule of thumb. This mitigates the overfitting that would arise from training a model and evaluating it on the same data. The datasets we selected are as follows.

- **Uniform:** 3,000 draws from a discrete uniform distribution over the set $\{0, 1, ..., 29\}$.
- **Left-skewed:** 10,000 draws from the discrete distribution over the set $\{0, 1, ..., 69\}$ with mass function p such that $p(i) = \frac{139}{140}p(i+1)$ for $0 \le i \le 68$.
- **Right-skewed:** 10,000 draws from the discrete distribution over the set $\{0, 1, ..., 69\}$ with mass function p such that $p(i+1) = \frac{139}{140}p(i)$ for $0 \le i \le 68$.
- **Bimodal symmetric:** 10,000 draws from a binomial distribution over the set $\{0, 1, ..., 39\}$ with parameter 0.7 and 10,000 draws from a binomial distribution over the same set with parameter 0.3.
- **Zero-inflated:** 9,800 draws from a discrete uniform distribution over the set $\{0, 1, ..., 79\}$ and 200 additional datapoints set to 0.
- **Top-inflated:** 9,900 draws from a discrete uniform distribution over the set $\{0, 1, ..., 79\}$ and 100 additional datapoints set to 79.

For each synthetic dataset above, we perform a set of simulations to assess how f affects the combined error objective. We operationalize count error using expected absolute deviation, and operationalize distribution error using Wasserstein distance. For the distribution privatizer we use the cyclic Laplace mechanism, and for the constructor algorithm we use Algorithm 1 with the sandwich selector. The total privacy budget ϵ_t is varied from 0.1 to 5.0, taking on 11 values that are evenly spaced on a logarithmic scale. We test every choice for f in the set $\left\{\frac{1}{10}, \frac{2}{10}, ..., \frac{9}{10}\right\}$. For every combination of the above parameters, we perform 100 simulations

of our two-stage framework, and record the value of f with the lowest combined error.

The results of our simulation study are presented in Table 3. We observe an inverse relationship between ϵ_t and the optimal f – generally, as ϵ_t decreases, the optimal f increases (or remains constant). As a robustness check, we perform additional simulations using MSE for count error, as well as total deviation and KS distance for distribution error. A similar inverse relationship between ϵ_t and the optimal f is observed in all cases.

Based on the use case for our rule of thumb, we select a functional form with a small number of parameters, which outputs a recommended f between 0 and 1. The form we select is

$$\hat{f}(\epsilon_t) = B + (A - B)\exp(-K\epsilon_t)$$

Here, $A \in [0, 1]$ is the limit as $\epsilon_t \to 0^+$, $B \in [0, 1]$ is the limit as $\epsilon_t \to \infty$, and K is a parameter that controls the curvature. We fit these parameters to the data in Table 3, yielding values $A = 0.639$, $B = 0.106$, and $K = 2.87$. This results in our rule of thumb,

$$\hat{f}(\epsilon_t) = 0.106 + 0.533\exp\left(-2.87\epsilon_t\right)$$

We evaluate the performance of this rule of thumb for our three canonical datasets. For each dataset, we compute the combined error objective using the rule of thumb $\hat{f}$ and also the standard set of values $f \in \left\{\frac{1}{10}, \frac{2}{10}, ..., \frac{9}{10}\right\}$. This can be seen for the crime dataset in Figure 8. The corresponding plots for the binomial and schools datasets are similar and we omit them. In Figure 8, the vertical green line depicts the value of f chosen by the rule of thumb. The value of the corresponding combined error objective is depicted by the horizontal green line. The horizontal blue line displays the best value of the objective across all values of f we simulated. Thus, the distance between the two horizontal lines can serve as an approximate measure of the optimality gap induced by our rule of thumb. Since the combined error objective is on a natural log scale, as long as the optimality gap is small, it can be roughly interpreted as a percent change in accuracy.

As observed for the crime data in Figure 8, the optimality gap tends to decrease as we increase ϵ_t. Scanning the panels from left to right, as ϵ_t increases from 0.1 to 0.5, 1.0, and 5.0, the corresponding optimality gap changes from 0.024 to 0.046, 0.017, and 0, respectively. Note that a measured optimality gap of 0 is possible anytime the objective at $\hat{f}$ matches the minimum objective over all values of f we test. Similar decreasing patterns are also present for the binomial and schools datasets. The worst optimality gap we observe across all experiments is for the binomial data at $\epsilon_t = 0.1$, with a value of 0.42.

We use this rule of thumb throughout the experiments in Section 6. As we observe in that section, despite the optimality gaps that we observe in this appendix, our fixed-point techniques using the rule of thumb often provide a favorable tradeoff among our three main performance criteria: Large increases in accuracy of distribution can be attained with modest performance losses in both accuracy of counts and runtime.