Post-processing Differentially Private Counts to Satisfy Additive Constraints

Ziang Wang[*], **Jerome P. Reiter**[*]

[*]Box 90251, Duke University, Durham, NC 27708, USA.

E-mail: `ziang.wang@duke.edu`, `jreiter@duke.edu`

Received 16 May 2021; received in revised form 21 July 2021; accepted 3 August 2021

Abstract. To reduce disclosure risks, statistical agencies and other organizations can release noisy counts that satisfy differential privacy. In some contexts, the released counts satisfy additive constraints; for example, the released value of a total should equal the sum of the released values of its components. We present a simple post-processing procedure for satisfying such additive constraints. The basic idea is (i) compute approximate posterior modes of the true counts given the noisy counts, (ii) construct a multinomial distribution with trial size equal to the posterior mode of the total and probability vector equal to fractions derived from the posterior modes of the components, and (iii) find and release a mode of this multinomial distribution. We also present an approach for making Bayesian inferences about the true counts given these post-processed, differentially private counts. We illustrate these methods using simulations.

Keywords. census; consistency; geometric; multinomial; summary.

1 Introduction

Government agencies and other data collectors, henceforth all called agencies, are obligated to release data in ways that protect the privacy of data subjects' identities and sensitive attributes. Simply stripping individuals' identifying information, like names and addresses, generally does not offer sufficient protection [1]. Even releasing summary statistics—which is the setting of this article—can result in unintended disclosures [2, 3]. Because of these threats, some agencies are investigating methods for releasing summary statistics that satisfy differential privacy [4, 5]. For example, the U.S. Bureau of the Census is releasing differentially private counts from the 2020 population census [6].

In many contexts, agencies seek to release counts that satisfy additive constraints. For example, the sum of the released population sizes in all fifty states must equal the released population size of the entire U.S. Such consistency requirements facilitate analyses for end users of the data. However, when the agency adds differentially private noise independently to both the sum and its components—for example, as done in initial steps of the TopDown Algorithm used by the Census Bureau [7, 8]—the resulting noisy counts may not obey the additivity constraint.

One way to ensure additivity is to add noise only to components, and generate noisy totals from the sum of the noisy components. This can result in high variances for the noisy totals, as the variance for any total accumulates over its components. An alternative approach, which we take here, is to enforce additivity after perturbing both the total and its components; that is, the agency modifies the differentially private counts to ensure additivity. This is an example of what is called post-processing [9]. Post-processing is commonly employed in practical implementations of differential privacy to enhance the interpretability or accuracy of differentially private data [10]. For example, it can be used to ensure all released counts are non-negative, which may be important in applications where data users would not understand how to interpret negative counts. It also has been used to increase accuracy of differentially private histograms [11].

In this article, we present a post-processing procedure for differentially private counts that ensures additivity of a set of released values for components and their total. The basic idea is to begin by applying a differentially private algorithm to add independent noise to each component count and the total count. Here, we add noise with the geometric mechanism [12]. We then compute a posterior mode of the true total given the noisy total. For the components, we posit a multinomial model for the true counts, using the posterior mode of the total as the trial size and approximate posterior modes of true counts to compute the multinomial probabilities. This guarantees that the released counts sum to the total, while also keeping the released counts close to the original noisy counts. We also present a corresponding procedure to help users obtain posterior inferences about the true counts given the released counts. We illustrate the post-processing and inference engines using simulation studies.

The remainder of this article is organized as follows. Section 2 reviews differential privacy, the geometric mechanism, and an algorithm for finding a mode of the multinomial distribution. Section 3 introduces the post-processing procedure, including some empirical illustrations of its properties. Section 4 presents the method for obtaining posterior inferences, including an illustrative example. Section 5 concludes with a summary of the findings and a discussion of future research.

2 Background

In Section 2.1, we review differential privacy and describe the geometric mechanism. In Section 2.2, we outline an algorithm for finding a mode of a multinomial distribution.

2.1 Differential privacy and geometric mechanism

To state the definition of differential privacy, we closely follow the notations and definitions in [9], tuned to count data. Let D be a database comprising observations from some universe $\mathcal{D}$. We measure the distance between two databases D and D′ using the ℓ_1 norm, denoted $||\mathrm{D}-\mathrm{D}'||_1$. Here, $||\mathrm{D}-\mathrm{D}'||$ is a measure of the number of records that differ between D and D′. When $\|\mathrm{D}-\mathrm{D}'\|_1 \leq 1$, we call D and D′ "neighboring databases."

Definition 1. (ϵ-differential privacy): A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{\mathcal{D}}$ is ϵ-differentially private if for all $\mathcal{S} \subset$ Range $(\mathcal{M})$ and for all $\mathrm{D}, \mathrm{D}' \in \mathbb{N}^{\mathcal{D}}$ such that $||\mathrm{D}-\mathrm{D}'||_1 \leq 1$,

$$Pr[\mathcal{M}(\mathrm{D}) \in \mathcal{S}] \leq \exp(\epsilon) Pr[\mathcal{M}(\mathrm{D}') \in \mathcal{S}], \tag{1}$$

where the probability space is defined by $\mathcal{M}$.

The value of ϵ, often called the privacy budget, controls the amount of privacy. Smaller values of ϵ offer greater privacy protection.

In Definition 1, and throughout, we consider neighboring databases where D has one more (or less) record than D′. One also can define neighboring databases as having the same number of records where one record differs in D and D′. The latter definition of neighboring databases is useful when the overall number of records in D is known to the public, which we do not assume here. Our post-processing algorithm also can be applied with this alternate definition of neighboring databases.

For count data, one method of satisfying differential privacy is the geometric mechanism [12]. This mechanism relies on the double (two-sided) geometric distribution. This distribution has a probability mass function defined on all integers such that

$$Pr[\Delta = \delta] = \frac{1-\alpha}{1+\alpha}\alpha^{|\delta|}. \tag{2}$$

Here, Δ is a random variable and δ is an integer-valued realization. We say that $\Delta \sim \mathrm{DG}(\alpha)$. The differentially private count d corresponding to the true count t is

$$d = t + \delta, \tag{3}$$

where $\delta \sim \mathrm{DG}(\alpha)$. It has been shown [12] that (3) offers ϵ-differential privacy with $\alpha = 1/\exp(\epsilon)$. We refer to counts generated from (3) as the noisy counts.

2.2 Mode of the multinomial distribution

Our post-processing algorithm relies on obtaining a mode of the multinomial distribution. We do so using Finucan's algorithm [13, 14], which we now summarize.

Suppose we have a multinomial distribution with m trials and probability vector $\mathbf{p} = (p_1, \ldots, p_S)$. We abbreviate this distribution as $\mathrm{MD}(m, \mathbf{p})$. Let $\lfloor . \rfloor$ denote the floor function that finds the integer part of a number. Consider initial values $k_i = \lfloor (m + S/2)p_i \rfloor$ and $m_0 = \sum_{i=1}^{S} k_i$. Set $f_i = (m + S/2)p_i - k_i$. Finucan's algorithm for finding the mode of $\mathrm{MD}(m, \mathbf{p})$ proceeds as follows.

1. If $m_0 < m$: Define $q_i = \frac{1-f_i}{k_i+1}$ for $i = 1, \ldots, S$.

   While $m_0 < m$, set

   $a = \arg\min_i q_i$
   $k_a = k_a + 1$
   $f_a = f_a - 1$
   $q_a = \frac{1-f_a}{k_a+1}$
   $m_0 = m_0 + 1.$

   Continue updating all parameters until $m_0 = m$.

2. If $m_0 > m$: Define $q_i = \frac{f_i}{k_i}$ for $i = 1, \ldots, S$.

   While $m_0 > m$, set

   $a = \arg\min_i q_i$
   $k_a = k_a - 1$
   $f_a = f_a + 1$
   $q_a = \frac{f_a}{k_a}$

$m_0 = m_0 - 1.$

Continue updating all parameters until $m_0 = m$.

3. $(k_1, \ldots, k_S)$ is a mode for $\mathrm{MD}(m, \mathbf{p})$.

Given m and $\mathbf{p}$, Finucan's algorithm finds one mode $(k_1, \ldots, k_S)$, even when multiple modes exist [15]. In our post-processing operations, finding any mode is sufficient for our purposes.

3 Description of the Post-processing Procedure

We now present the post-processing algorithm that ensures additivity as well as non-negativity. To do so, we make use of approximations to the posterior distributions of the true counts given the noisy counts, which we describe in Section 3.1. We describe the algorithm itself in Section 3.2. We present illustrative simulations in Section 3.3.

3.1 Approximations to posterior distribution of the true count

Suppose we have S true counts, $\mathbf{N} = (N_1, \ldots, N_S)$. Let $N = \sum_{s=1}^{S} N_s$ be the total of the S counts. We apply the geometric mechanism independently to both N and each component of $\mathbf{N}$. Let $n = N + \delta$, where δ is noise sampled via the geometric mechanism. For $s = 1, \ldots, S$, let $n_s = N_s + \delta_s$, where again each δ_s is sampled via a geometric mechanism. From (2) and (3), we have

$$p(n \mid N) = p(\delta = n - N) = \alpha^{|n-N|}\left(\frac{1-\alpha}{1+\alpha}\right) \tag{4}$$

$$p(n_s \mid N_s) = p(\delta_s = n_s - N_s) = \alpha_s^{|n_s-N_s|}\left(\frac{1-\alpha_s}{1+\alpha_s}\right). \tag{5}$$

We note that n does not necessarily equal $\sum_{s=1}^{S} n_s$. Our goal is to apply a post-processing procedure to the noisy counts $(n, n_1, \ldots, n_S)$ to ensure that the released values of the components sum to the released value of the total (and are non-negative), while keeping reasonably close to the values of n and $\mathbf{n} = (n_1, \ldots, n_S)$.

To implement the post-processing algorithms, we make use of the posterior distributions of the true counts given the noisy counts, which we find by Bayesian inference. In doing so, we treat the noisy counts $(n, n_1, \ldots, n_S)$ as the data and the true counts $(N_1, \ldots, N_S)$ as random variables. By Bayes' theorem, we find the posterior distribution,

$$p(N_1, \ldots, N_S | n_1, \ldots, n_S, n) \propto p(n_1, \ldots, n_S, n | N_1, \ldots, N_S) p(N_1, \ldots, N_S). \tag{6}$$

When S is large, computing this posterior distribution can be computationally challenging. Thus, we consider two approximations to the posterior distribution for purposes of the post-processing (not inference), described below.

3.1.1 Approximation based on independence assumptions

In this approximation, we assume that the total count and component counts are all independent in the posterior distribution, even though in reality they are not. Thus, we have

$$p(N_1, \ldots, N_S | n_1, \ldots, n_S) \approx p(N_1|n_1) \cdots p(N_S|n_S). \tag{7}$$

As convenient prior distributions, we assume that N and each N_s are independently and uniformly distributed on some sufficiently large, integer-valued sample space. Then, we approximate the posterior distributions as

$$p(N \mid n) \propto p(N)p(n \mid N) \propto p(n \mid N) \quad = \quad \alpha^{|n-N|}\left(\frac{1-\alpha}{1+\alpha}\right) \tag{8}$$

$$p(N_s \mid n_s) \propto p(N_s)p(n_s \mid N_s) \propto p(n_s \mid N_s) \quad = \quad \alpha_s^{|n_s-N_s|}\left(\frac{1-\alpha_s}{1+\alpha_s}\right). \tag{9}$$

As we compute the posterior distributions independently for each count, we disregard additivity constraints. Doing so makes it much easier to compute the approximate posterior distribution for the purpose of post-processing.

3.1.2 Approximation based on summed noisy total

In addition to n, $\mathbf{n} = (n_1, \ldots, n_S)$ can provide information about N. We may be able to improve inferences about N by using both n and $\mathbf{n}$. We now describe an approximate posterior distribution that takes advantage of this information.

Let the sum of the noisy counts be $n^* = \sum_s n_s$. We define η such that $\eta = n^* - N$. We note that n^* and n are independent conditional on N.

For convenience, we assume that N follows a uniform distribution on some sufficiently large integer-valued sample space. Then, the approximate posterior distribution of N given (n^*, n) is

$$p(N \mid n, n^*) \propto p(n, n^* \mid N)p(N) = p(n \mid N)p(n^* \mid N)p(N). \tag{10}$$

Here, $p(n \mid N)$ is the probability of sampling $\delta = n - N$ based on (4). And, $p(n^* \mid N)$ is the probability of sampling η from the sum of S independent double geometric distributions. We approximate $p(n^* \mid N)$ as follows.

1. Generate 10000 sets of $(\delta_1, \ldots, \delta_S)$ from the double geometric distributions used to make $(n_1, \ldots, n_S)$.

2. For each set of $(\delta_1, \ldots, \delta_S)$, calculate the corresponding $\eta = \sum_s \delta_s$.

3. The approximate probability of sampling any η is its corresponding frequency divided by 10000.

We use $p(N|n, n^*)$ in lieu of (8), and otherwise use (9) to approximate the posterior distribution of each N_s.

3.2 The algorithm for generating the released counts

We present two flavors of the post-processing algorithm, one that uses the independence assumption of Section 3.1.1 and one that uses the summed noisy counts of Section 3.1.2. We first present a version of the algorithm that uses the independence assumption.

1. Add independent noise via the geometric mechanism to N. Call the noisy count n.

2. Add independent noise via geometric mechanisms to each N_s. Call each noisy count n_s.

3. Under the approximation based on independence, compute the posterior mode of N based on (8). Call the modal value $\hat{N}$.

4. For each $s = 1, \ldots, S$, compute the posterior mode of N_s from (9). Call each modal value $\hat{N}_s$.

5. Compute the probabilities $\theta_s = \hat{N}_s / \sum_{s=1}^{S} \hat{N}_s$ for each s.

6. Using $\mathbf{p} = (\theta_1, \ldots, \theta_S)$ and setting $m = \hat{N}$, apply Finucan's algorithm to find $\tilde{\mathbf{N}} = (\tilde{N}_1, \ldots, \tilde{N}_S)$. These counts, along with $\tilde{N} = \hat{N}$, are released.

By design, this algorithm generates released counts such that $\sum_{s=1}^{S} \tilde{N}_s = \tilde{N}$, that is, the released counts satisfy the additive constraint. With uniform prior distributions, the posterior modes in step 3 and step 4 should equal the noisy counts, except for any $n_s \leq 0$ (or $n \leq 0$) in which case $\hat{N}_s = 0$ (or $\hat{N} = 0$). Thus, because we base each θ_s on the posterior mode of N_s, and we also use a mode-finding algorithm for post-processing, we expect that $\tilde{\mathbf{N}}$ generally should be reasonably close to $\mathbf{n}$.

For the approximation based on the summed noisy counts, we replace step 3 with step 3′.

3′. Under the approximation based on the summed noisy total, compute the posterior mode of N based on (10). Call the model value $\hat{N}$.

We emphasize that the post-processing steps in the algorithm, i.e., those after step 2, are based only on the differentially private counts $(n_1, \ldots, n_S, n)$. These steps do not use the confidential counts. Hence, as long as the noisy counts generated in step 1 and step 2 are differentially private, the post-processing property of differential privacy guarantees that the post-processed counts $(\tilde{N}_1, \ldots, \tilde{N}_S, \tilde{N})$ remain differentially private.

3.3 Illustrative simulations of the post-processing algorithm

To illustrate the performance of the post-processing algorithm, we use repeated sampling simulations. Specifically, for a fixed set of true counts, we iterate the procedure from Section 3.2 for 10000 times. We consider two sets of fixed true counts: a scenario with several large and small counts (Scenario 1) and a scenario with mostly small counts and one relatively large count (Scenario 4). For these two simulation scenarios, we use $\epsilon = 1$ for all counts. We also consider a Scenario 2 where we use $\epsilon = 0.1$ for N and $\epsilon = 5$ for each N_s, and a Scenario 3 where we use $\epsilon = 5$ for N and $\epsilon = 0.1$ for each N_s. We use the true counts from Scenario 1 in these two scenarios.

In Scenario 2 and Scenario 3, the differences in ϵ for the total and components are more extreme than what might be chosen in practice. They represent scenarios where using the summed noisy counts could be especially beneficial or mostly pointless, and hence facilitate comparisons of the post-processing algorithm based on Section 3.1.1 and Section 3.1.2. To see this, note that the theoretical variances of the summed noisy totals, $Var(n^*)$, before post-processing are $(92.1, 0.7, 9991.7, 92.1)$ in Scenarios 1 – 4, respectively. Thus, in Scenario 2, n^* is likely to estimate N far more accurately than n does, but in Scenario 3, n^* is a highly variable estimate of N. The large variance of n^* in Scenario 3 also illustrates why it can be beneficial to add noise to both the total and components.

For Scenarios 1 – 3, we simulate counts of 50 components including three values of 0; one value each of 1, 2 and 3; five values between 15 and 60; and the remainder of values scattered uniformly between 60 and 1000. The counts are displayed in Figure 1. We present

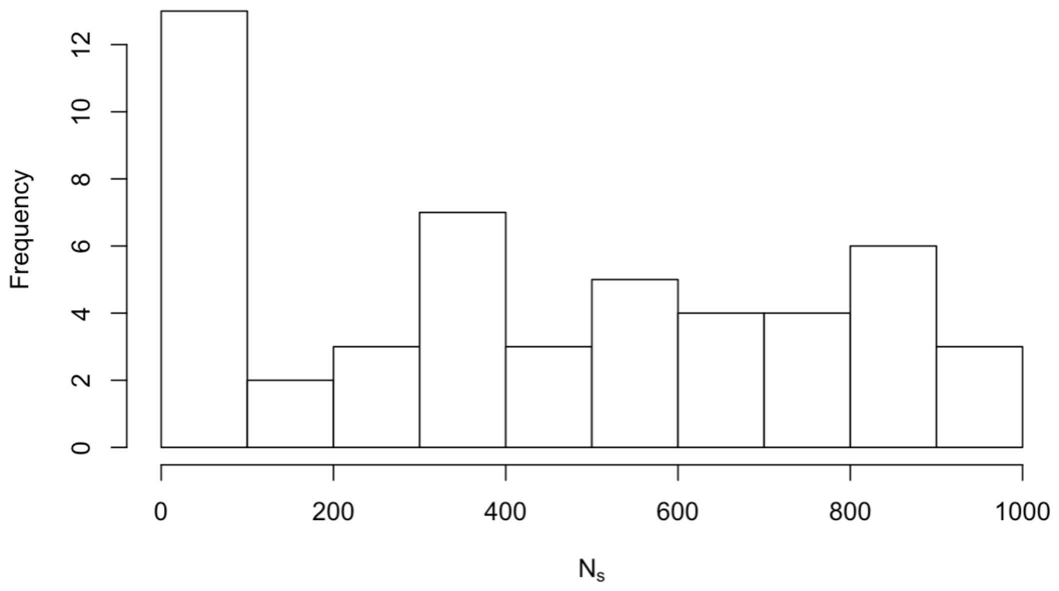


Figure 1: Distribution of true counts N_s in simulations in Scenarios 1 – 3.

Count	1	2	3	4	5	6	7	8	9	10
Instances	7	5	3	6	3	2	2	4	4	1
Count	11	12	13	14	16	18	27	36	53	435
Instances	1	1	1	2	1	2	2	1	1	1

Table 1: True counts for the simulation in Scenario 4.

results of the post-processing algorithm for six counts: the minimum $N_{min} = 0$, the first quantile $N_{q1} = 60$, the median $N_{med} = 382$, the third quantile $N_{q3} = 736$, the maximum $N_{max} = 977$, and the total $N = 21249$. For Scenario 4, we generate counts as shown in Table 1, resulting in $N_{min} = 1$, $N_{q1} = 2$, $N_{med} = 6$, $N_{q3} = 11$, $N_{max} = 435$, and $N = 863$.

In all simulations, we use a computationally convenient approximation to the double geometric distribution. Specifically, we truncate the sample space of Δ to values that have non-negligible mass under the appropriate $\mathrm{DG}(\alpha)$. For example, for $\epsilon = 1$, the chance of generating a $|\delta| \geq 50$ is very small (about 1×10^{-22}). Thus, for the simulation we restrict the support of the noise distribution to $[-50, 50]$. When necessary, we increase to an appropriate range that captures nearly all the probability mass.

Table 2 displays the average and variance of each element in $(\tilde{N}_{min}$, $\tilde{N}_{q1}$, $\tilde{N}_{med}$, $\tilde{N}_{q3}$, $\tilde{N}_{max}, N)$ over the 10000 simulation runs for the algorithms based on both the independence approximation and the summed noisy total approximation. In general, across all scenarios and both algorithms, the averages are close to the true counts. The only instance of non-negligible relative bias in the post-processed counts is for N_{min} in Scenario 3. As we use $\epsilon = 0.1$ in this scenario, the generated value of n_{min} can be far from zero. For any $n_{min} \leq 0$, the posterior mode $\hat{N}_{min} = 0$, and for any $n_{min} > 0$ we have $\hat{N}_{min} = n_{min}$. Thus, $\hat{N}_{min}$, and hence $\tilde{N}_{min}$, has an upward bias. The bias is more substantial in Scenario 3 compared to others where $\tilde{N}_{min}$ is based on $\epsilon \geq 1$ because (i) the number of simula-

Count	N_{min}	N_{q1}	N_{med}	N_{q3}	N_{max}	N
Scenario 1						
True Count	0	60	382	736	977	21249
Indep.	0.4 (0.7)	60 (2.0)	382 (2.0)	736 (2.0)	977 (2.0)	21249 (1.8)
Summed.	0.4 (0.7)	60 (2.0)	382 (2.0)	736 (2.0)	977 (2.0)	21249 (1.8)
Scenario 2						
True Count	0	60	382	736	977	21249
Indep.	0.01 (0.0)	60 (0.1)	382 (0.1)	736 (0.3)	977 (0.5)	21249 (191)
Summed	0.01 (0.0)	60 (0.0)	382 (0.0)	736 (0.0)	977 (0.1)	21249 (0.7)
Scenario 3						
True Count	0	60	382	736	977	21249
Indep.	5 (71)	60 (191)	381 (187)	735 (184)	976 (193)	21249 (0.0)
Summed	5 (71)	60 (191)	381 (187)	735 (184)	976 (193)	21249 (0.0)
Scenario 4						
True Count	1	2	6	11	435	863
Indep.	1.0 (1.1)	2.0 (1.5)	5.9 (1.8)	11.0 (1.8)	438 (19)	863 (1.8)
Summed	1.0 (1.1)	2.0 (1.5)	5.9 (1.8)	11.0 (1.8)	438 (19)	863 (1.8)

Table 2: Averages and variances of post-processed, differentially private counts over 10000 simulation runs for each scenario. Here, "Indep." refers to algorithms based on independence approximation, and "Summed" refers to the algorithms based on the summed noisy total approximation.

tion runs where $n_{min} < 0$ is largest in Scenario 3—these numbers are $(2714, 72, 4765, 1045)$ in Scenario 1 through Scenario 4, respectively—and (ii) when positive, n_{min} has a higher chance of being far from N_{min}. Unsurprisingly, the variances for the component counts are largest in Scenario 3. In the other scenarios, the variability in the post-processed counts is generally small compared to the corresponding true counts.

The results in Scenario 1 and Scenario 4 offer additional insight into the effects of the post-processing algorithms. In particular, since the post-processing operations do not affect $\tilde{N}$, we know that the variance associated with N (1.8 in both settings) approximates the variance from applying the geometric mechanism with $\epsilon = 1$, without post-processing, regardless of the true count. Using this fact, we see that the post-processing operations generally do not add much to the variances for the counts that are not large; in fact, they actually can decrease the variance. The post-processing does increase the variance for N_{max} in Scenario 4. The distribution of N_{max} in this scenario is right-skewed, with around a 2% chance that $\tilde{N}_{max} \geq 450$ and almost no chance that $\tilde{N}_{max} < 430$. Evidently, when the posterior modes equal zero for many of the small counts, the probability $\theta_{N_{max}}$ associated with the largest count can increase. As a result, Finucan's algorithm finds a multinomial mode with $\tilde{N}_{max}$ greater than its true value of 435.

Comparing the algorithms based on the independence approximation and the summed noisy total approximation, we see few practical differences for each of the five values of N_s and N across Scenarios 1, 3, and 4. In these scenarios, the variance of n^*, which accumulates over $S = 50$ draws of the geometric mechanism, is large compared to the variance of n. Hence, in these scenarios the summed noisy total adds little information about N. However, we see a marked difference in Scenario 2. For $\tilde{N}$, the variance for the algorithm based on the summed noisy total is orders of magnitude smaller than the variance for the algorithm based on the independence assumption. We also see reductions in the variances for some of the component counts. Because we use $\epsilon = 5$ for the components, n^* typically is much closer to N than is n, which is based on $\epsilon = 0.1$. Since (10) takes advantage of both sources of information about N, the algorithm based on the summed noisy total offers more accurate post-processed counts in this scenario.

4 Posterior Inferences for the True Counts

Given the post-processed, differentially private counts, $(\tilde{N}_1, \ldots, \tilde{N}_S, \tilde{N})$, analysts need to estimate the values of the confidential counts, $(N_1, \ldots, N_S, N)$. Here we outline a Bayesian inference procedure for point and interval estimation of $(N_1, \ldots, N_S, N)$. In doing so, we explicitly seek to preserve the additivity constraint. This was not essential in Section 3 when we constructed the post-processing algorithm, but it is necessary to have the right posterior distribution for inference.

To begin, we assume that the analyst specifies prior distributions for each $N_s \in \mathbf{N}$. Here, we use uniform distributions over a sufficiently large domain, requiring $N = N_1 + \cdots + N_S$. Marginally, this implies a prior distribution on N that is not a uniform distribution; rather, it follows a sum of uniform distributions. Alternatively, the analyst could set a prior distribution on N directly, such as a uniform distribution on some suitable range. Then, $p(N_1, \ldots, N_S, N) = p(N)p(N_1, \ldots, N_s \mid N)$, and the analyst must specify $p(N_1, \ldots, N_S \mid N)$. For example, this could be a uniform distribution over all combinations of $(N_1, \ldots, N_S)$ that sum to N or a multinomial distribution with *a priori* probabilities for the S components. We do not investigate such prior specifications here.

With the uniform prior distribution, the posterior distribution can be written as

$$\begin{aligned} p(N_1, \ldots, N_S \mid \tilde{N}_1, \ldots, \tilde{N}_S) &\propto p(\tilde{N}_1, \ldots, \tilde{N}_S \mid N_1, \ldots, N_S) p(N_1, \ldots, N_S) \\ &\propto \sum_{n_1} \cdots \sum_{n_S} \sum_{n} p(\tilde{N}_1 \ldots \tilde{N}_S, n_1, \ldots, n_S, n \mid N_1, \ldots, N_S) \\ &= \sum_{n_1}, \ldots, \sum_{n_S} \sum_{n} p(\tilde{N}_1, \ldots, \tilde{N}_S \mid n_1, \ldots, n_S, n, N_1, \ldots, N_S) \\ &\times \; p(n_1, \ldots, n_S, n \mid N_1, \ldots, N_S). \end{aligned} \tag{11}$$

We note that $(n_1, \ldots, n_s, n)$ are unobserved. Thus, we need to average over $p(n_1, \ldots, n_s, n \mid N_1, \ldots, N_s)$.

We present an illustration of the use of (11) for $S = 2$ components. We do so for post-processed counts that derive from the algorithm based on the independence assumption. To compute $p(\tilde{N}_1, \tilde{N}_2, n_1, n_2, n \mid N_1, N_2)$, we proceed as follows.

1. For a given (N_1, N_2), enumerate all possible combinations of (n_1, n_2) from a suitably large region around (N_1, N_2). For example, with $\epsilon = 1$, a reasonable range spans $(N_1 \pm 30) \times (N_2 \pm 30)$. The size of each interval depends on the value of ϵ for that count. One should use a wider range when $\epsilon < 1$. Under the post-processing algorithm, generally $\tilde{N} = \hat{N} = n$, so we do not need to consider any other plausible values of n.

2. For a particular combination of (n_1, n_2, n), compute $p(n_1, n_2, n \mid N_1, N_2)$ using (4) and (5).

3. For the same particular combination of $(n_1, n_2 \mid N_1, N_2)$, find the posterior modes $\hat{N}_1$ and $\hat{N}_2$ using step 4 in Section 3.2. Compute $\theta = (\theta_1, \theta_2)$ per step 5 in Section 3.2.

N_1	N_2	N	Probability
250	357	607	0.21
251	356	607	0.13
251	357	608	0.08
250	356	606	0.08
249	358	607	0.07
250	358	608	0.05
249	357	606	0.05
251	358	609	0.03
249	356	605	0.03
252	355	607	0.03

Table 3: Ten highest posterior probabilities of true counts. The post-processed, differentially private counts are $\tilde{N}_1 = 250$, $\tilde{N}_2 = 357$, and $\tilde{N} = 607$.

4. Apply Finucan's algorithm to find a mode, using $m = n$ as the trial size and θ as the probabilities. When the mode from Finucan's algorithm equals $(\tilde{N}_1, \tilde{N}_2)$, set $p(\tilde{N}_1, \tilde{N}_2 \mid n_1, n_2, n) = 1$, otherwise set $p(\tilde{N}_1, \tilde{N}_2 \mid n_1, n_2, n) = 0$.

5. Multiply the probabilities in step 2 and step 4, and save the product.

6. Repeat steps 2 – 5 for all combinations identified in step 1.

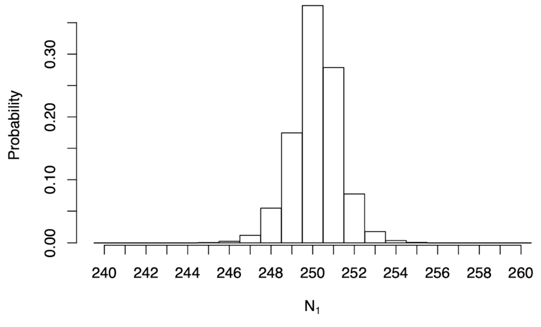
We repeat steps 1 – 6 over all possible combinations of (N_1, N_2) from a suitably large region around $(\tilde{N}_1, \tilde{N}_2)$; for example, $(\tilde{N}_1 \pm 30) \times (\tilde{N}_2 \pm 30)$. After cycling through the full space of (N_1, N_2), we normalize by summing over the products from all the step 6s. The result is an estimate of the posterior distribution in (11).

We illustrate the posterior inferences using $\epsilon = 1$ for both the total and components. We suppose that $\tilde{N}_1 = 250$, $\tilde{N}_2 = 357$, and $\tilde{N} = 607$. Table 3 displays the combinations of (N_1, N_2) with the ten highest posterior probabilities. The modal combination equals the post-processed, differentially private counts. It is clearly separated from the next highest combination. Nonetheless, the top ten probabilities reveal posterior mass away from the mode, indicating the importance of characterizing the uncertainty.

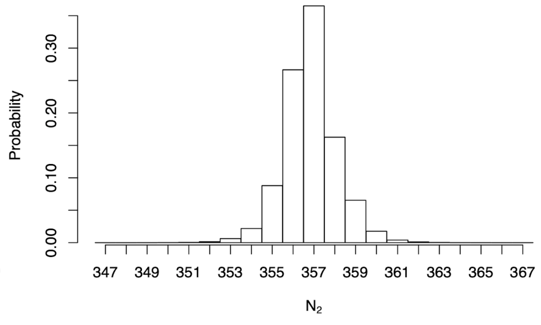
We also can view the marginal posterior distributions of N_1, N_2, and N, which are shown in Figure 2a, Figure 2b, and Figure 2c, respectively. We obtain the marginal distribution for any of the variables by accumulating the posterior probability for each value in the sample space of that variable. We see that the posterior distributions are centered around the post-processed counts with most posterior mass within ± 4 of the these counts.

5 Discussion

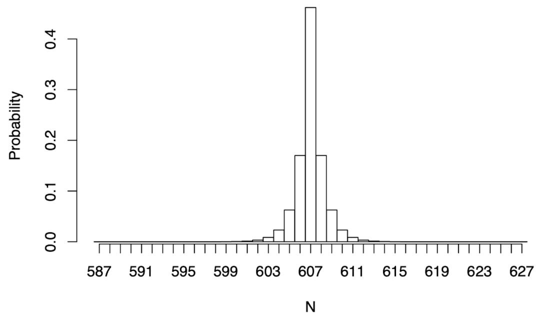
To summarize, the post-processing algorithm can generate released counts that are close to the true counts, at least for the values of ϵ examined here. Additionally, we find no practical differences between the algorithms based on the independence approximation and the summed noisy total approximation, except for the case where the ϵ values for the components are large compared to the ϵ value for the total. On the downside, when a set of true counts has many small values, e.g., they equal zero, and only a few large values,



(a) Marginal distribution of N_1.



(b) Marginal distribution of N_2.



(c) Marginal distribution of N.

Figure 2: Marginal posterior distributions of (N_1, N_2, N) given $(\tilde{N}_1, \tilde{N}_2) = (250, 357)$ and $\tilde{N} = 607$, computed using the independence assumption.

the post-processing algorithm can result in bias as well as increased variability for large counts. This is the price one pays to enforce the additivity and non-negativity constraints.

Using the properties of the geometric mechanism and Finucan's algorithm, analysts can obtain posterior inferences about the true counts using a Bayesian approach. We illustrated the algorithm with $S = 2$ counts. These computations took about 3.5 minutes on a standard desktop with an implementation in the software package R with minimal efforts at making run times efficient. Nonetheless, computations following this "brute force" approach get cumbersome with larger S. For example, if one considers a ± 30 grid of values for each additional N_s, a naive accounting of the number of additional computations increases by a factor of roughly 61^2, which could translate to a huge increase in computational time. However, there are ways to speed up the computations; for example, one easily could spread computations over many processors, which would dramatically reduce the run time. Efficient coding aside, with larger S, most combinations will have very low probabilities that practically do not need to be computed. Thus, a next step of the research is to consider alternatives to the "brute force" approach that explore regions of high probability, such as those based on Markov chain Monte Carlo sampling.

The post-processing algorithm here is particularly suited to settings where the agency adds noise to both a total and its components. In such settings, agencies should account for the aggregate privacy loss when selecting values of ϵ. For example, when adding noise with $\epsilon = 1$ to each element of (N, N_1, N_2, N_3), where (N_1, N_2, N_3) are counts from disjoint groups, the aggregate privacy loss is $\epsilon = 2$, not $\epsilon = 1$. Agencies may be able to use sophisticated methods like the matrix mechanism [16] to generate $(n, n_1, \ldots, n_S)$ with lower error than those generated by the independent geometric mechanisms used here. We expect that the post-processing algorithm (from step 3 and 3′ onward) can be applied to such counts with satisfactory performance.

Our methods are targeted at the setting where an agency releases a set of differentially private counts one time. As noted by a reviewer, in the interactive setting, analysts can query a system repeatedly for differentially private counts. In this case, an analyst could request a noisy version of the same count, say N_k, multiple times, and get a different value of n_k each time (presuming the total privacy budget allows, as the privacy loss may grow by ϵ each time a new noisy version of the count is released). If the analyst wanted to use the post-processing strategy presented here, the analyst would need to compute the posterior mode of N_k using the appropriate posterior distribution, e.g., $p(N_k \mid n_k^1, n_k^2)$ where (n_k^1, n_k^2) are two differentially private versions. After finding all modes, the post-processing algorithm could be used as illustrated in the simulations.

Other post-processing algorithms can be used to ensure additivity (and non-negativity) constraints are satisfied, such as those based on optimization routines [10]. As our post-processing algorithm is based on posterior modes, it inherits some benefits of Bayesian modeling. In particular, our approach naturally allows agencies to leverage prior information about the true counts in post-processing. Related, it offers a principled way to utilize the information in the summed noisy total potentially to improve the accuracy of the post-processed total. Typical implementations of optimization algorithms, such as minimizing the squared distance between the post-processed and noisy counts subject to the additivity (and non-negativity) constraints, may not do so. However, unlike other post-processing algorithms, our algorithm applies only when the set of counts can be expressed as a saturated multinomial model. This is not the case, for example, in the U. S. decennial census application, in which an optimization routine is run on noisy counts from k-way marginal tables.

Our post-processing algorithm tries to find the (approximately) highest probability set of counts, which could differ from a solution found by another post-processing algorithm. For a simple example, suppose we have $S = 2$ counts with $(n_1, n_2, n) = (1, 9, 11)$. Using the independence approximation, our algorithm returns a highest probability solution of $(\tilde{N}_1, \tilde{N}_2, \tilde{N}) = (1, 10, 11)$. The post-processing algorithm that minimizes squared distances between released and noisy counts could generate this solution, but it also could generate $(2, 9, 11)$ or $(1, 9, 10)$, depending on where the optimization algorithm starts. Of course, it is difficult to say whether one solution is "better" than the others, as it depends on the definition of "better."

Finally, a potential alternative to post-processing entirely is to generate differentially private counts that automatically satisfy the additivity constraints. This would follow the general strategy of simultaneous editing and disclosure limitation [17, 18, 19, 20, 21, 22]. To date, we are not aware of approaches for doing so with differentially private counts, although it may be possible to do so via a congenial differential privacy approach like those proposed in [23]. This represents an important area of research.

References

[1] A. F. Karr and J. P. Reiter. Using statistics to protect privacy. In J. Lane, V. Stodden, S. Bender, and H. Nissenbaum, editors, *Privacy, Big Data, and the Public Good: Frameworks for Engagement*, pages 276–295. Cambridge University Press, 2014.

[2] Institute of Medicine. *Sharing Clinical Trial Data: Maximizing Benefits, Minimizing Risk*. The National Academies Press, Washington DC, 2015.

[3] C. Dwork, A. Smith, T. Steinke, and J. Ullman. Exposed! A survey of attacks on private data. *Annual Review of Statistics and Its Application*, 4:12.1–12.24, 2017.

[4] Y. Rinott, C. M. O'Keefe, N. Shlomo, and C. Skinner. Confidentiality and differential privacy in the dissemination of frequency tables. *Statistical Science*, 33:358–385, 2018.

[5] J. P. Reiter. Differential privacy and federal data releases. *Annual Review of Statistics and Its Application*, 6:85–101, 2019.

[6] J. Abowd. The U.S. Census Bureau adopts differential privacy. *KDD'18 Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 2867, 2018.

[7] J. Abowd, R. Ashmead, G. Simson, D. Kifer, P. Leclerc, A. Machanavajjhala, B. Moran, and W. Sexton. Census TopDown: Differentially private data, incremental schemas, and consistency with public knowledge. Technical report, US Census Bureau, 2019.

[8] S. Petti and A. D. Flaxman. Differential privacy in the 2020 US census: What will it do? Quantifying the accuracy / privacy tradeoff. *Gates Open Research*, 3:1722, 2020.

[9] C. Dwork and A. Roth. *The Algorithmic Foundations of Differential Privacy*. Foundations and Trends in Theoretical Computer Science, 2014.

[10] K. Zhu, P. van Hentenryck, and F. Fioretto. Bias and variance of post-processing in differential privacy, 2020. Available at https://arxiv.org/pdf/2010.04327.pdf.

[11] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. In *Proceedings of the VLDB Endowment*, volume 3(1-2), page 1021–1032, 2010.

[12] A. Ghosh, T. Toughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. *Society for Industrial and Applied Mathematics*, 41(6):1673–1693, 1928.

[13] F. Le Gall. Determination of the modes of a multinomial distribution. *Statistics and Probability Letters*, 62:325–333, 2005.

[14] H. M. Finucan. The mode of a multinomial distribution. *Biometrika*, 51:513–517, 1964.

[15] W. White and M. Hendy. A fast and simple algorithm for finding the modes of a multinomial distribution. *Statistics and Probability Letters*, 80:63–68, 2010.

[16] C. Li, G. Miklau, M. Hay, A. McGregor, and V. Rastogi. The matrix mechanism: optimizing linear counting queries under differential privacy. *The VLDB Journal*, 24:757–781, 2015.

[17] H. J. Kim, A. F. Karr, and J. P. Reiter. Statistical disclosure limitation in the presence of edit rules. *Journal of Official Statistics*, 31:121–138, 2015.

[18] H. J. Kim, J. P. Reiter, and A. F. Karr. Simultaneous edit-imputation and disclosure limitation for business establishment data. *Journal of Applied Statistics*, 45:63–82, 2018.

[19] N. Shlomo and T. De Waal. Protection of microdata subject to edit constraints against statistical disclosure. *Journal of Official Statistics*, 24:1–26, 2008.

[20] N. Shlomo and T. De Waal. Preserving edits when perturbing microdata for statistical disclosure control. In *Conference of European Statisticians*, page WP11, 2005.

[21] V. Torra. Constrained microaggregation: adding constraints for data editing. *Transactions on Data Privacy*, 1:86–104, 2008.

[22] L. Wei and J. P. Reiter. Releasing synthetic magnitude microdata constrained to fixed marginal totals. *Statistical Journal of the IAOS*, 32:93–108, 2016.

[23] R. Guong and X. L. Meng. Congenial differential privacy under mandated disclosure. In *FODS 2020 – Proceedings of the 2020 ACM-IMS Foundations of Data Science Conference*, pages 59–70. Association for Computing Machinery, Inc, 2020.