Technical Appendix for Protection System Post-Secondary Employment Outcomes (PSEO) (Beta)

This appendix details how we protect the released PSEO data. First, we discuss the Laplace noise infusion, which is used to protect the count queries for Graduate Earnings and Pipeline Flows. Second, we detail how we use Laplace noise in the histogram approach to calculate percentiles of earnings within a cell.

1 Protecting Count Data: Laplacian Noise Infusion

Consider a dataset d, and a neighboring dataset d' which differs by one observation. Furthermore, consider a count query $q_c(.)$ on a dataset, which returns the number of observations with certain attributes, which we will refer to as X. Now consider the cases below:

$$|q_c(d) - q_c(d')| = \begin{cases} 1, \text{if the differing observation has the attributes } X \\ 0, otherwise. \end{cases} \tag{1}$$

The sensitivity $S(q_c)$ is then the smallest number such that for any neighboring datasets,

$$|q_c(d) - q_c(d')| \leq S(q_c)$$

In the case of the count query, $S(q_c) = 1$. Therefore, for any count query $q_c(d)$, if we draw $\zeta \sim Lap(1/\epsilon)$, then $q_c(d) + \zeta$ is ϵ-differentially private. For these tables, all the counts are integers, and therefore we will draw Laplace noise using the geometric noise mechanism.[1]

[1]The geometric noise mechanism distribution is the difference between two geometric random variables, such that $\zeta \sim X - Y$, where $X, Y \sim Geometric(p)$ where $p = 1 - \frac{1}{e^\epsilon}$.

2 Protecting Percentiles: Histogram Approach

To protect percentiles, we use an approach called the histogram approach.

Consider a dataset d, which has sorted values $e_1, e_2, ..., e_N$ and a function $H(.)$ that assigns each value e_i into a bin, grouping them together. Formally, this function is defined as:

$$H(e_i) = \begin{cases} 1, & \text{if } e_i \in [b_1, b_2) \\ & ... \\ j, & \text{if } e_i \in [b_j, b_{j+1}) \\ & ... \\ M-1, & \text{if } e_i \in [b_{M-1}, b_M) \\ M, & \text{if } e_i \geq b_M \end{cases} \tag{2}$$

Where the borders of the histogram bins, b_i, are public information. The key decision in implementing this protection method is determining how to set the bin definitions, which we describe in the next subsection.

Choosing Bin Definitions

We choose the cutoffs as follows. The bottom cutoff is \$10,000, which is very close to the minimum value in the data by construction, given that we restrict the sample based on earnings. For the next 19 b_is, we choose every 5th percentile of the log normal distribution with mean 11.003 and standard deviation 0.753.[2] Additionally, for b_M, we use the 97.5th percentile value of the distribution, which is about \$260,000. Finally, for any earnings greater than that value, we count it in the final bin, M. Together, we have 21 bins.

For reference, these histogram values are in the appendix.

[2]The log normal distribution is a good approximation of the overall earnings distribution. The mean and standard deviation were calculated using the 5-year ACS Public-Use Microsample. We calculated the mean and standard deviation of wage and salary income for employed individuals with a BA or above.

Queries to Protect

From the definition of the histogram function above, the set of queries we protect are of the form q_j^c, which returns the count of the observations in a given bin j. Additionally, these queries imply the corresponding empirical CDF:

$$F(j) = \frac{\sum_{i=1}^{j} q_i^c}{\sum_{i=1}^{M} q_i^c} \tag{3}$$

The sensitivity of each of these queries is 1, and therefore we can protect each of these queries with privacy loss ϵ by adding Laplace noise as described above in Section 1. Therefore, our protected counts are:

$$\tilde{q}_j^c = q_j^c + \zeta$$

Where $\zeta \sim Laplace(1/\epsilon)$.[3] The resulting histogram list of counts is ϵ-differentially private (Proposition 1 in [1]), and any function of these counts is also ϵ-differentially private because of the composition properties of differential privacy.[4]

Calculating Protected Percentiles

We use these fuzzed values to create a fuzzed CDF,

$$\tilde{F}(j) = \frac{\sum_{i=1}^{j} \tilde{q}_i^c}{\sum_{i=1}^{M} \tilde{q}_i^c}$$

If we assume that earnings are distributed uniformly within a bin, we can use $\tilde{F}(j)$ to extract protected percentiles.[5]

To calculate a percentile Y, suppose that it is in bin J such that

$$\frac{\sum_{i=1}^{J-1} \tilde{q}_i^c}{\sum_{i=1}^{M} \tilde{q}_i^c} < Y/100 \leq \frac{\sum_{i=1}^{J} \tilde{q}_i^c}{\sum_{i=1}^{M} \tilde{q}_i^c} \tag{4}$$

[3] As noted above, because these counts are integers, we use the geometric noise function, which is the integer analog of the Laplace noise distribution.

[4] This result from Hay et al. (2009) allows the list of values $(\tilde{q}_1, \tilde{q}_2, ..., \tilde{q}_M)$ to also be considered releasable.

[5] Note that $\tilde{F}(j)$ will not necessarily be a true CDF, because there may be cases when $\tilde{q}_j < 0$.

Then, the Yth percentile is $b_J + (b_{J+1} - b_J) \times \frac{(Y/100 \times \sum^J \tilde{q}_i^c) - \sum^{J-1} \tilde{q}_i^c}{\tilde{q}_J^c}$.[6] In the case where a percentile is in the largest bin, we define b_{M+1} to be the 99.9th percentile of earnings from the log normal distribution, which is 614597.[7]

We use this technique to calculate the 25th, 50th, and 75th percentile values.

Calculating Protected Counts

We use the fuzzed counts from the histogram approach to calculate the total cell count. In this application, this is just a sum of all the histogram counts: $cellcount_c = \sum_{i=1}^{M} \tilde{q}_i^c$.

We determine whether or not to release values for a cell based on the total protected cell count, because values are noisier for smaller cells. Specifically, we do not release any data for cells with protected counts below 30. When we release tables, we will simply indicate that the cell count is below 30 and publish missing values.

References

[1] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially-private queries through consistency. *CoRR*, abs/0904.0942, 2009.

[6] In words, if a bin J includes the Yth percentile, and the Yth percentile is W of the way through the interval defined by bin J, then the Yth percentile is the lower-bound value of bin J, b_J, plus $W \times width$.

[7] We calculate the percentile for the smallest bin such that Equation 4 is satisfied. This addresses the issue of negative counts in a subsequent bin, since it is possible for Equation 4 to be fulfilled in two distinct bins if the intervening bins have negative counts.

A Tables Appendix

Table 1: Histogram bin values

Bin	Lower Bound	Upper Bound
1	10000	17403
2	17403	22876
3	22876	27512
4	27512	31857
5	31857	36128
6	36128	40449
7	40449	44914
8	44914	49605
9	49605	54609
10	54609	60027
11	60027	65982
12	65982	72639
13	72639	80226
14	80226	89080
15	89080	99735
16	99735	113106
17	113106	130970
18	130970	157509
19	157509	207050
20	207050	262475
21	262475	614597

Notes: Except for the lowest value, these are all percentiles from a log normal distribution with mean 11.003 and standard deviation 0.753. Any observation will be classified into the final bin (21) if it has a value above 262475. For purposes of calculating the percentiles, we use the upper bound value for bin 21 of 614597, which is the 99.9th percentile of the log normal distribution.