Final Document: Improving Utility of Differentially Private Confidence Intervals

Benjamin Glass

August 12, 2016

1 Overview

A differentially private randomized algorithm, M, is one meeting the requirement that given two *neighboring* datasets d and d', that is datasets that differ in no more than one row, and a set of outcomes S, the following condition that $\Pr[M(d) \in S] \leq e^{\epsilon} \Pr[M(d') \in S]$ holds for some $\epsilon \geq 0$. Differentially private algorithms run on datasets can provide the guarantee that the information of any one contributor to that dataset is obscured by adding noise to statistical estimates. [1] Social scientists are currently interested in being able to run regressions on data in a differentially private fashion in order to learn about relationships within a dataset without exposing the information of individuals in the dataset. If given a dataset the covariance matrix could be estimated in a differentially private fashion, differentially private regressions could be run using the entries of the matrix. [2] To date, however, techniques used to construct differentially private covariance matrices have yielded results with large amounts of noise and little utility. Efforts are being made now to write algorithms for constructing differentially private confidence intervals for means assuming Normal population data as a first step toward solving the question of how to run differentially private regressions.

1.1 Existing Algorithms

"Optimal Estimation of Differentially Private Confidence Intervals" by Karwa & Vadhan presents seven algorithms with the following outputs:

- Algorithm 1: differentially private range estimate with known population variance.
- Algorithm 2: differentially private scale estimate for unknown variance case.
- Algorithm 3: differentially private range estimate with unknown population variance.
- Algorithm 4: differentially private confidence interval for a mean; known variance.
- Algorithm 5: differentially private confidence interval for a mean; unknown variance.
- Algorithm 6: differentially private confidence interval for a difference of means test.
- Algorithm 7: differentially private confidence interval for a paired t-test.

Algorithms 1-3 are necessary for differentially private estimation of bounds on data and scale for when population variance is unknown. Algorithms 4-7 construct four types of differentially private confidence intervals based on Laplace mechanism mean estimates. [3] All of the algorithms in the paper assume independent and identically distributed Normal population data.

[1] See D'Orazio, Honaker, & King § 2.1.
[2] Ibid. 25.
[3] On the Laplace mechanism see Roth & Dwork § 3.3.

1.2 Summer Agenda and Assessment Criteria

As of the start of the summer, no code to run and test the above algorithms had been written. Once this code was written the performance of the algorithms could be assessed. If possible, it would then be necessary to improve the performance of the algorithms to achieve desired performance levels.

Performance of confidence intervals can be assessed by considering coverage probability and interval length. Coverage probability is the probability that an interval covers the population value of the parameter for which the interval estimate is being calculated. By convention, coverage probability is expressed as 1 - α, taking α to be the percent of Type I error permitted and 1 - α to be the *confidence level*. The parameter α is often taken to be 0.05 in order to provide 95% confidence.

Length of a confidence interval is taken to be the distance between the upper and lower bounds of the interval. The narrower a confidence interval, the more informative the interval is as to the value of the population value of a parameter. Confidence intervals are constructed such that for larger sample sizes, the width of the intervals narrows. Asymptotically, the length of the confidence interval converges to 0 as the upper and lower bounds converge to the population parameter value.

A high-performing confidence interval is one that provides a high coverage probability and narrowly spaced upper and lower bounds for the parameter value. A differentially private confidence interval is necessarily wider than a non-private confidence interval since the differentially private estimates the intervals are constructed around include additional noise via the Laplace mechanism and the width of the interval must increase to account for this added noise. Confidence levels of differentially private confidence intervals, however, can be equal to those achieved by non-private confidence intervals. For Algorithms 4-7 above, it is desired that 1 - α coverage can be achieved and that the ratio between the lengths of the differentially private confidence intervals to the lengths of non-private confidence intervals be less than or equal to 1.5 for a sample size of 10,000 observations.

1.3 Simulations and Improvements

To date, code for all seven of the algorithms in “Optimal Estimation of Differentially Private Confidence Intervals” has been drafted and is shown in the appendix. Simulations and improvements to date have focused on Algorithm 1 and Algorithm 4, the two algorithms used to construct a differentially private confidence interval for a mean under the assumption that population variance is known. Assessment of the performance of Algorithm 5, for constructing differentially private confidence intervals for means when the population variance is unknown, has been started.

2 Differentially Private Confidence Intervals for Means with Known Population Variance

Algorithm 1 and Algorithm 4 in “Optimal Estimation of Differentially Private Confidence Intervals” are the algorithms necessary to run to construct differentially private confidence intervals for means with known population variance. Throughout section 2, references to “the algorithms” or the “confidence intervals” refer respectively to Algorithm 1 and Algorithm 4 and confidence intervals for means with known population variance.

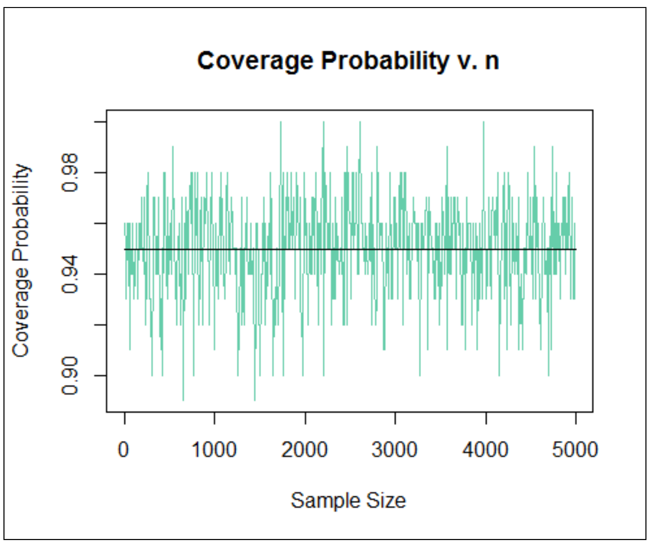
Algorithm 1, shown in the appendix generates a differentially private range for data, assuming the bounds of the data are unknown. Algorithm 1 requires inputs of (i) randomly generated data, (ii) an ϵ value to specify allowed privacy loss, (iii) an α value to specify the confidence level of the confidence intervals making use of the range algorithm, (iv) an assumed upper bound R on the value of the population mean, and (v) a variance parameter σ^2. The output is a differentially private range estimate for the data.

Algorithm 4, also shown in the appendix, computes the confidence interval for a mean, assuming variance is known. Algorithm 4 requires inputs of (i) randomly generated data, (ii) three parameters α_0, α_1, α_2 that sum to the desired α-level of the interval, (iii) two privacy parameters ϵ_1 and ϵ_2, (iv) a variance parameter σ^2, and (v) an assumed upper bound R on the population mean. Algorithm 4 runs Algorithm 1, truncates the data to fit within the range produced by Algorithm 1, calculates a differentially private mean estimate via the Laplace mechanism, then places bounds around that estimate to construct the desired interval.

The non-private confidence interval for a mean assuming known population variance σ^2 for sample size n and Normal quantile z^* is $\left(\bar{X} - z^* \frac{\sigma}{\sqrt{n}}, \bar{X} + z^* \frac{\sigma}{\sqrt{n}}\right)$ for which the width is $2z^* \frac{\sigma}{\sqrt{n}}$. The probabilistic confidence guarantee for some permitted level of error α is found via: $\Pr\left[\mu \in \left(\bar{X} \pm z^* \frac{\sigma}{\sqrt{n}}\right)\right] = 1-\alpha$. The coverage probabilities and lengths of the differentially private confidence interval constructed via Algorithm 1 and Algorithm 4 were compared to these non-private results.

2.1 Initial Performance

Initially, the Algorithm 1 and Algorithm 4 were producing differentially private confidence intervals that were conservative with higher average coverage probabilities than non-private confidence intervals. The lengths of the differentially private confidence intervals, however, were notably larger than those of the non-private confidence intervals.



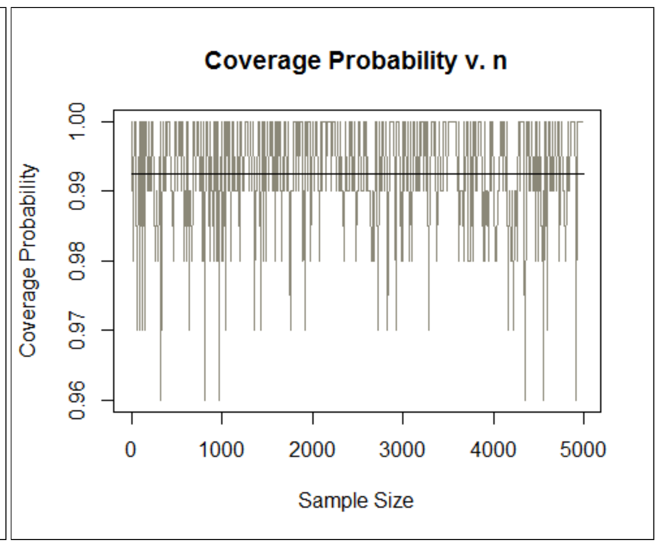


Figure 1: Left: Coverage probabilities for changing sample size for non-private confidence intervals for means with known population variance. Right: Coverage probabilities for changing sample size for differentially private confidence intervals for means with known population variance. Black horizontal lines mark average coverage probability values.

Confidence interval lengths for varying sample size for both non-private and differentially private confidence intervals are shown below.

Sample sizes between 1,000,000-10,000,000 observations were necessary to achieve a ratio of differentially private to non-private confidence interval lengths at or below 1.5. A difference in lengths below 1 was achieved with sample sizes between 1000-5000.

From these initial results it was clear that the desired confidence levels were being achieved, but that the performance of the differentially private confidence intervals with regard to length required improvement.

The sensitivity term $\frac{4\sigma\sqrt{\log\frac{n}{\alpha_1}}}{\epsilon_2 n}$ in Algorithm 4 was causing the differentially private confidence intervals to be wider than desired. Subsequent improvement relied on reducing the sensitivity term.

n	NP_Length	DP_Length	Dif_Len	Rat_Len
1	3.92000000	733.33189330	729.411893299	187.07446
10	1.23961284	93.31356791	92.073955066	75.27638
50	0.55437172	21.26815062	20.713778902	38.36442
100	0.39200000	11.21713749	10.825137489	28.61515
500	0.17530773	2.55283361	2.377525882	14.56201
1000	0.12396128	1.35896995	1.235008670	10.96286
5000	0.05543717	0.32556818	0.270131012	5.87274
10000	0.03920000	0.17998406	0.140784063	4.59143
100000	0.01239613	0.02918242	0.016786288	2.35416
1000000	0.00392000	0.00620387	0.002283867	1.58262
10000000	0.00123961	0.00163620	0.000396582	1.31992

2.2 Optimal Division of α

The width term for the confidence interval for means with known population variance is:

$$w = \frac{\sigma}{\sqrt{n}} z_{1-\alpha_0/2} + \frac{4\sigma\sqrt{\log\frac{n}{\alpha_1}}}{\epsilon_2 n} \log\left(\frac{1}{\alpha_2}\right)$$

The computation of w requires the division of the error term α into three values α_0, α_1, and α_2 such that $\alpha_0 + \alpha_1 + \alpha_2 = \alpha$. The algorithms for producing confidence intervals yield varying results for different choices of α_0, α_1, and α_2. The first improvement to the algorithms was implemented by numerically solving for the optimal values of α_0, α_1, and α_2 that minimize the value of w.

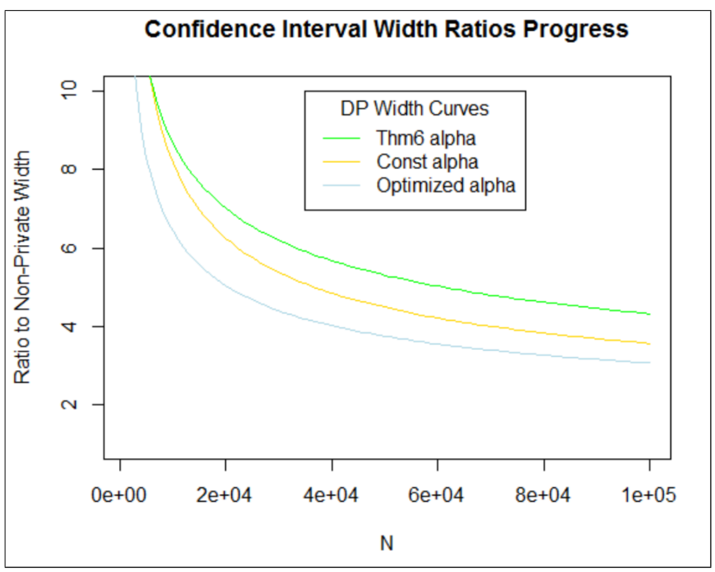


Figure 2: The curves in this plot show convergence to 1 of ratios of differentially private to non-private confidence interval lengths. The green and yellow curves represent the ratios as calculated for different α divisions. The green curve results use optimal values proposed in Theorem 6 in "Optimal Estimation of Differentially Private Confidence Intervals," and the yellow curve results use alternate values. The blue curve, achieving the lowest of the three ratios, plots ratios for optimal division of α.

2.3 Reducing Existing Bounds

The width term w discussed above is derived in part from the distance of any data point in a sample, X_i, from the differentially private mean, $\hat{l}\sigma$ calculated via the histogram mechanism. The following line appears in the proof of the need for the second term in w:

$$|X_i - \hat{l}\sigma| \leq |X_i - \mu| + |\mu - \hat{l}\sigma| \leq 2\sigma\sqrt{2\log\left(\frac{6n}{\alpha}\right)} + \frac{\sigma}{2} \leq 4\sigma\sqrt{\log\left(\frac{n}{\alpha}\right)}$$

The final term in this inequality, $4\sigma\sqrt{\log\left(\frac{n}{\alpha}\right)}$, is intended to simplify the preceding term, $2\sigma\sqrt{2\log\left(\frac{6n}{\alpha}\right)} + \frac{\sigma}{2}$, without increasing the resulting value of w in the result shown above in § 2.2. Computing w using the second-to-last term in the above inequalities and constructing the confidence intervals in that fashion yields lower ratios of confidence interval lengths than those observed previously.

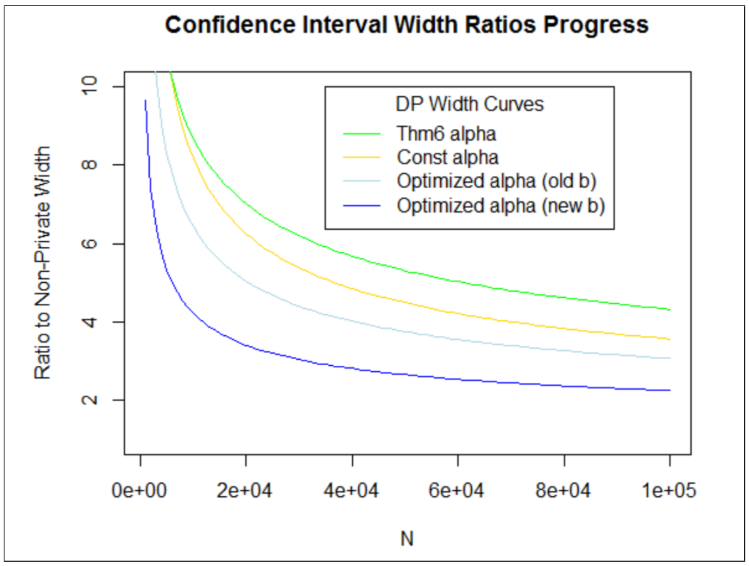


Figure 3: The dark blue curve plots the improved ratios based on the updated inequality bound.

2.4 Use of the Normal-Laplace Distribution

Since "Optimal Estimation of Differentially Private Confidence Intervals" assumes Normal data, mean estimators for data will be Normally distributed since linear combinations of Normal random variables are Normal as well. Differentially private mean estimators constructed via the Laplace mechanism are thus sums of a Normal and a Laplace distribution. The convolution of a Normal and a Laplace distribution follows a known distribution called the Normal-Laplace distribution. [4] A package exists in R providing the distribution, density, and quantile functions for the Normal-Laplace distribution. Using these numerical tools in R, the differentially private confidence intervals for means with known population variance were constructed by building intervals from the 1-$\frac{\alpha}{2}$% and $\frac{\alpha}{2}$% quantiles from the Normal-Laplace distribution to provide the desired 1-α level coverage. The value for w initially shown above can be replaced under this model with the 1-$\frac{\alpha}{2}$% quantile from the Normal-Laplace distribution. The length ratios achieved using the Normal-Laplace distribution method are lower than those achieved with previous models.

2.5 Bypassing Boole's Inequality

Similar to the discussion in § 2.3 the maximum distance of a point X_i in a dataset from the population mean μ was used in the derivation of the initial w value. Taking $Z_i = X_i - \mu$, the following inequality is achieved for a constant c via Boole's inequality:

[4]See Reed, William J. "The Normal-Laplace Distribution and its Relative". 2004. Available at: http://www.math.uvic.ca/faculty/reed/NL.draft.1.pdf.

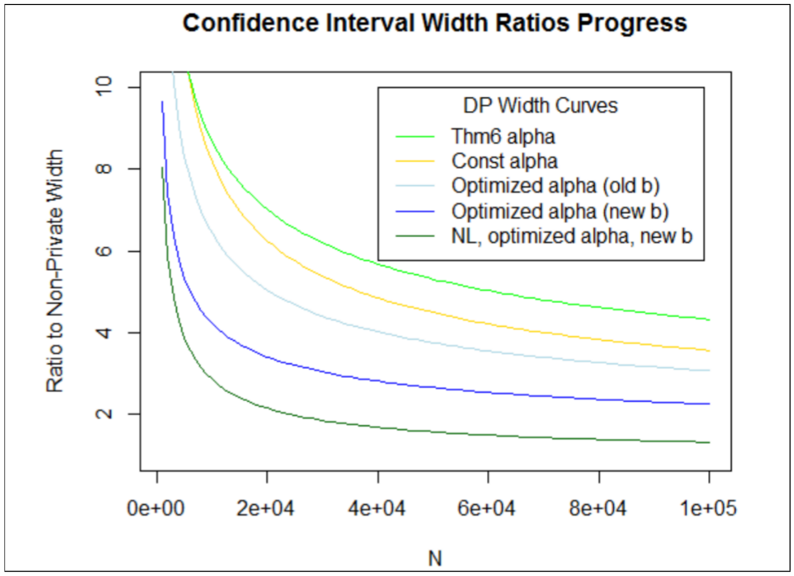


Figure 4: The dark green curve plots the improved ratios based on Normal-Laplace distribution quantiles.

$$\Pr\left[\max_i |Z_i| \geq c\right] \leq 2ne^{-\frac{c^2}{2\sigma^2}}$$

The event $\max_i |Z_i| \geq c$ can be expressed as a union and the above probability is equivalent to:

$$\Pr\left[\max_i |Z_i| \geq c\right] = \Pr[|Z_1| \geq c \cup |Z_2| \geq c \cup ... \cup |Z_n| \geq c] = \Pr[\exists i \text{ s.t. } |Z_i| \geq c]$$

By taking the complement and using the assumption of independence of the Z_i relied upon in the paper the following can be stated:

$$1 - \Pr[\exists i \text{ s.t. } |Z_i| \geq c] = \Pr[\forall i \text{ s.t. } |Z_i| < c] = \prod_{i=1}^{n} \Pr[|Z_i| < c]$$

In place of the additive bound term $2ne{-}\frac{c^2}{2\sigma^2}$, the following equality is achieved:

$$\prod_{i=1}^{n} \Pr[|Z_i| < c] = \left(1 - 2e{-}\frac{c^2}{2\sigma^2}\right)^n$$

Solving the following for the constant c provides the constant term necessary to achieve at least 1 - α level coverage:

$$\left(1 - 2e{-}\frac{c^2}{2\sigma^2}\right)^n \geq 1 - \alpha$$

The value $c = \sigma\sqrt{2\log\left(\frac{2}{1-(1-\alpha)^{1/n}}\right)}$ solves this inequality. Previously the constant term provided by the additive bound was $c = \sigma\sqrt{2\log\left(\frac{2n}{\alpha}\right)}$. Using the new bound to construct the sensitivity of the Laplace draw for the differentially private mean estimate yielded lower length ratios.

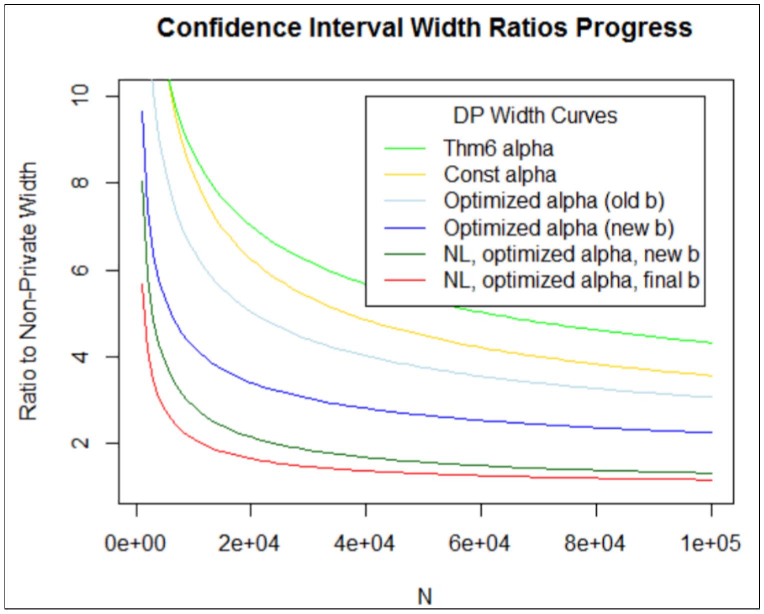


Figure 5: The red curve plots the improved ratios based on Normal-Laplace distribution quantiles with the updated sensitivity.

2.6 Considering Bounds on Data

Algorithm 1 for finding the range of a data set in a differentially private manner assumes that the range of data is unknown. In practice this may be an unnecessary assumption. Certain data types may have well-known bounds or bounds that are known to specialists. Under such circumstances, the actual bounds of the data could be used in place of running Algorithm 1. The sensitivity used in the scale parameter of the Laplace draw in Algorithm 4 for computing a differentially private mean estimate would be, for a dataset d with sample size n: $\frac{|\max(d)-\min(d)|}{\epsilon n}$. In this case, Algorithm 4 would be the only differentially private algorithm necessary to run since it would be unnecessary to use Algorithm 1. As such, the division of ϵ into two terms ϵ_1 and ϵ_2 would be unnecessary and the full ϵ could be provided as an input to Algorithm 4. The ratio of lengths of private to non-private confidence intervals assuming the true range of data is known are lower than all of those achieved above and are close to 1, as desired. A potential update to Algorithm 4 may be to add a conditional statement that runs Algorithm 1 only if bounds on the data are truly unknown and uses actual data bounds when known.

2.7 Optimal Division of ϵ

Algorithm 4 calls Algorithm 1 with a parameter ϵ_1 and computes a differentially private mean estimate with a second parameter ϵ_2. The privacy loss for constructing a confidence interval using Algorithm 4 is $\epsilon_1 + \epsilon_2 = \epsilon$. At present, discussion as to the optimal division of ϵ in Algorithm 4 is underway. Difficulty arises in determining what function to optimize the values of ϵ_1 and ϵ_2 over. Nonetheless an optimal division of ϵ may exist and provide improved results.

3 Differentially Private Confidence Intervals for Means with Unknown Population Variance

Algorithm 2, Algorithm 3, and Algorithm 5 in "Optimal Estimation of Differentially Private Confidence Intervals" are the algorithms used to construct differentially private confidence intervals for

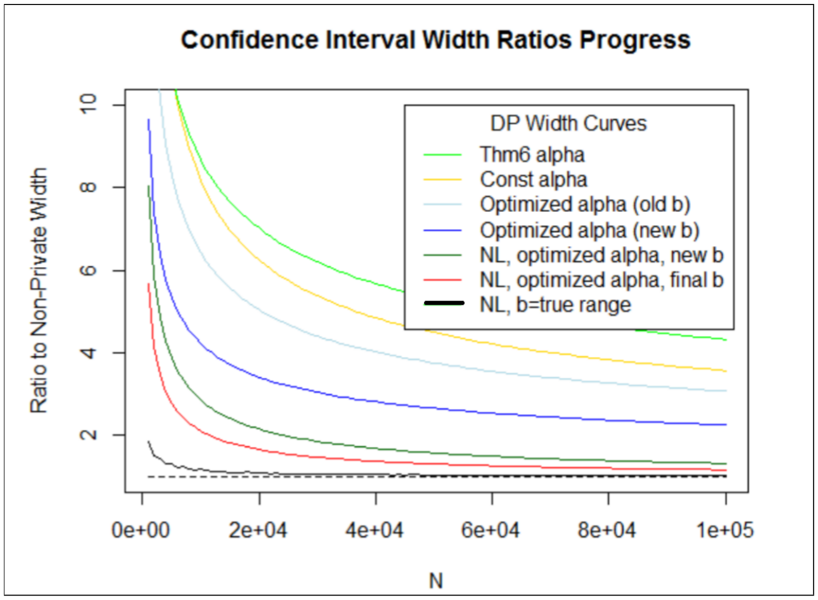


Figure 6: The black curve plots the ratios achieved with the true range known. The dashed line is at a ratio value of 1.

means with unknown population variance. Throughout section 3, references to "the algorithms" and the "confidence intervals" refer respectively to these three algorithms and to confidence intervals for means with unknown population variance.

Algorithm 2 requires as input (i) a randomly generated data set, (ii) a privacy loss parameter ϵ, (iii) a confidence level parameter α, (iv) an assumed upper bound R on the mean of the data, (v) and an assumed lower bound σ_{min} on the standard deviation of the data. The output is a differentially private scale estimate for the data.

Algorithm 3 requires the same inputs as Algorithm 2, runs Algorithm 2 to compute a differentially private scale estimate, then runs Algorithm 1 with the output of Algorithm 2 to construct a differentially private range estimate for the data.

Algorithm 5 requires the same inputs as Algorithm 2 and Algorithm 3, runs Algorithm 3 to compute a differentially private range estimate for the data input, then constructs a differentially private confidence interval for a mean assuming unknown population variance.

The non-private confidence interval for a mean assuming unknown population variance σ^2 for sample size n, t-distribution quantile t^*, and sample variance s is $\left(\bar{X} - t^*\frac{s}{\sqrt{n}}, \bar{X} + t^*\frac{s}{\sqrt{n}}\right)$ for which the width is $2t^*\frac{s}{\sqrt{n}}$. The probabilistic confidence guarantee for some permitted level of error α is found via: $\Pr\left[\mu \in \left(\bar{X} \pm t^*\frac{s}{\sqrt{n}}\right)\right] = 1 - \alpha$. The coverage probabilities and lengths of the difference private confidence intervals constructed via Algorithm 2, Algorithm 3, and Algorithm 5 are compared to these non-private results.

3.1 Initial Performance

As in the known variance case, the confidence intervals for means with unknown variance are conservative. The coverage probabilities plotted below are those produced for construction of 95% confidence intervals for samples of varying sizes.

With regard to length as well, the initial constructions of confidence intervals for means with unknown population variance performed below the level of the non-private analogs.

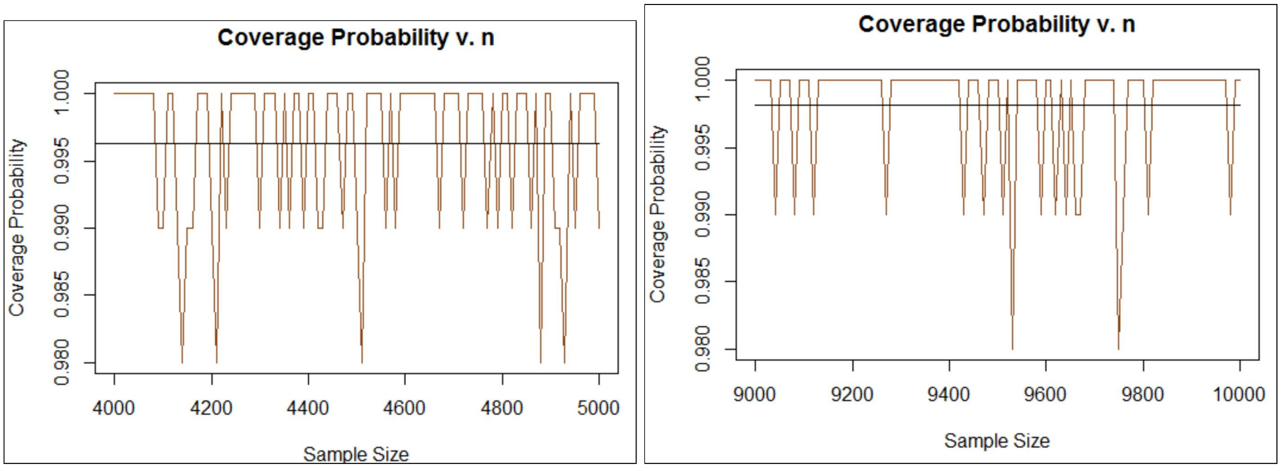


Figure 7: Left: Coverage probabilities for 95% confidence intervals computed for samples of sizes 4000-5000 with the average indicated by the black horizontal line. Right: Coverage probabilities for 95% confidence intervals computed for samples of sizes 9000-10000 with the average indicated by the black horizontal line.

	n	NP_Length	DP_Length	Dif_Len	Rat_Len
1	50	0.61916276	24.90030167	24.281138906	40.21608
2	100	0.39206128	12.86059875	12.468537466	32.80252
3	500	0.17588685	2.81273396	2.636847115	15.99172
4	1000	0.12536984	1.48569216	1.360322323	11.85047
5	5000	0.05548361	0.33616381	0.280680196	6.05879
6	10000	0.03916649	0.18701554	0.147849051	4.77489
7	100000	0.01242136	0.02964974	0.017228379	2.38700
8	1000000	0.00391599	0.00623192	0.002315931	1.59140
9	10000000	0.00123995	0.00163956	0.000399619	1.32229

Figure 8: The lengths of differentially private confidence intervals for means with unknown population variance are noticeably greater than those of the non-private analogs. A difference between the lengths below 1 is not achieved until sample sizes between 1000-5000. A ratio below 1.5 is not achieved until sample sizes between 1,000,000-10,000,000.

3.2 Potential Improvements

As indicated by the notable difference in performance of the differentially private algorithms from the non-private algorithms in the unknown variance case, there is room for improvement. Optimal division of α and ϵ in the algorithms is an initial change to consider as in the known variance case. Considering inequalities relied upon in the derivations of the algorithms as a source for tightening bounds or reducing sensitivities may be another method of improving results. Considering whether or not it is possible to use Normal-Laplace quantiles to model differentially private mean estimators may be helpful as well.

4 Appendix

Presented below for reference are the algorithms, simulation code, and plotting code discussed throughout this document.

4.1 Algorithms

The following algorithms are reproduced from "Optimal Estimation of Differentially Private Confidence Intervals" by Karwa & Vadhan.

Algorithm 1 Differentially Private Estimate of Range with known variance

Input: $X_1, \ldots, X_n$, ϵ, α, R, σ.
Output: An ϵ differentially private estimate of the range of $X_1, \ldots, X_n$.

1: Let $r = \lceil \frac{R}{\sigma} \rceil$. Divide $[-R-\frac{\sigma}{2}, R+\frac{\sigma}{2}]$ into $2r$ bins of length σ each in the following manner. A bin labeled i is $[(i-0.5)\sigma, (i+0.5)\sigma]$, for $i = \{-r, \ldots, r\}$.
2: Construct a histogram on the interval $[-R-\frac{\sigma}{2}, R+\frac{\sigma}{2}]$ using the sample $X_1, \ldots, X_n$ using the bins created in the previous step. Note that some points may fall outside the range of the histogram.
3: Use the Laplace mechanism with ϵ budget to pick the largest noisy bin of the histogram. Let this (random) bin be $\hat{l}$.
4: The range of the data is $[X_{\min}, X_{\max}]$, where

$$X_{\min} = \sigma\hat{l} - 4\sigma\sqrt{\log\left(\frac{n}{\alpha}\right)}, X_{\max} = \sigma\hat{l} + 4\sigma\sqrt{\log\left(\frac{n}{\alpha}\right)}$$

Algorithm 2

Input: $X_1, \ldots, X_n$, ϵ, α, R, $\sigma_{\min}$.
Output: An ϵ differentially private approximate estimate of the scale of $X_1, \ldots, X_n$.

1: Divide the positive half of the real line into bins of exponentially increasing length. The bins are of the form $[2^i, 2^{i+1}]$ for $i = i_{\min}, \ldots, i_{\max}$. where $i_{\max} = \log_2 R$ and $i_{\min} = \log_2 \sigma_{\min}$.

$$[\sigma_{\min}, \ldots, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8, \ldots, R]$$

2: Form pairs of $\frac{n}{2}$ differences $Y_i = X_{i_1} - X_{i_2}$. Note that the pairing here is random.
3: Construct a histogram of $|Y_1|, \ldots, |Y_k|$ using the bins in Step 1. Let c_i be the number of counts in interval i.
4: Use a Laplace mechanism to select an interval with the largest number of counts i^*, i.e.

$$i^* = \max_i \{c_i + Z_i\}$$

where $Z_i \sim Lap\left(0, \frac{2}{\epsilon}\right)$.
5: Output $\sigma^* = 2^{i^*+1}$

Algorithm 3 Differentially Private Estimate of Range with unknown variance

Input: $x_1, \ldots, x_n$, ϵ, α, R, $\sigma_{\min}$.
Output: An ϵ differentially private estimate of the range of the data $x_1, \ldots, x_n$.

1: Run Algorithm 2 with $\epsilon_1 = 0.5\epsilon$ on the data $x_1, \ldots, x_n$ to obtain an estimate of scale σ^*.
2: Run Algorithm 1 with $\epsilon_2 = 0.5\epsilon$, $\alpha = \alpha$ and $\sigma = \sigma^*$ to get $[X_{\min}, X_{\max}]$.
3: Output σ^* and $[X_{\min}, X_{\max}]$.

Algorithm 4 Differentially Private Confidence Interval with known variance.

Input: $X_1, \ldots, X_n$, $\alpha_0, \alpha_1, \alpha_2$, ϵ, σ, R
Output: An $\epsilon = \epsilon_1 + \epsilon_2$ differentially private $\alpha = \alpha_0 + \alpha_1 + \alpha_2$ level confidence interval of μ

1: Run Algorithm 1 with $\epsilon = \epsilon_1$ and $\alpha = \alpha_1$ to get an estimate of the range $[X_{\min}, X_{\max}]$.
2: Let

$$Y_i = X_i \mathbb{I}(X_i \in \mathcal{R})$$

where $\mathbb{I}(.)$ is the indicator function and $\mathcal{R} = [X_{\min}, X_{\max}]$
3: Let

$$\tilde{\mu} = \frac{\sum_i Y_i}{n} + Z_1$$

where Z_1 is a Laplace random variable with mean 0 and scale parameter

$$b_1 = \frac{4\sigma \cdot \sqrt{\log \frac{n}{\alpha_1}}}{\epsilon_2 n}.$$

4: Let

$$w = \frac{\sigma}{\sqrt{n}} z_{1-\frac{\alpha_0}{2}} + \frac{4\sigma\sqrt{\log \frac{n}{\alpha_1}}}{\epsilon_2 n} \log\left(\frac{1}{\alpha_2}\right)$$

where $\alpha_0 + \alpha_1 + \alpha_2 = \alpha$, and $z_{1-\frac{\alpha_0}{2}}$ is the $1 - \frac{\alpha_0}{2}$ quantile of a standard normal variable.
5: Output the confidence interval:

$$[\tilde{\mu} - w, \tilde{\mu} + w]$$

Algorithm 5

Input: $X_1, \ldots, X_n$, α, ϵ, R, $\sigma_{\min}$.
Output: An $\epsilon = \epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4$ differentially private α level confidence interval of μ.

1: Run Algorithm 3 with $\epsilon = \epsilon_1$ and $\alpha = \alpha_1$ to get an estimate of scale σ^* and range $[X_{\min}, X_{\max}]$.

2: Let Y_i be the value of X_i truncated to lie between $[X_{\min}, X_{\max}]$.

$$\left[\sigma^* \hat{l} - 2\sigma^* \sqrt{\log \frac{n}{\alpha_1}}, \sigma^* \hat{l} + 2\sigma^* \sqrt{\log \frac{n}{\alpha_1}}\right]$$

3: Let

$$\tilde{\mu} = \frac{\sum_i Y_i}{n} + Z_2$$

where Z_2 is a Laplace random variable with scale parameter

$$b_2 = \frac{4\sigma^* \left(\sqrt{\log \frac{n}{\alpha_1}}\right)}{n\epsilon_2}$$

and mean 0.

4: Truncate $\tilde{\mu}$ to lie in the interval

$$\left[\sigma^* \hat{l} - 2\sigma^* \sqrt{\log \frac{n}{\alpha_1}}, \sigma^* \hat{l} + 2\sigma^* \sqrt{\log \frac{n}{\alpha_1}}\right]$$

5: Let

$$s_1^2 = \frac{\sum_i (Y_i - \tilde{\mu})^2}{n-1}.$$

Note that the sensitivity of s_1^2 is

$$\frac{4\sigma^* \left(\sqrt{\log \frac{n}{\alpha_1}}\right) \left|\hat{l}\sigma^* - \tilde{\mu}\right|}{n} \leq \frac{8\sigma^* \left(\sqrt{\log \frac{n}{\alpha_1}}\right)}{n} := \epsilon_3 b_3$$

Hence let $\tilde{s}^2 = s_1^2 + Z_3 + b_3 \log \frac{1}{\alpha_3}$ where $Z_3 \sim Lap(0, b_3)$.

6:

$$w = \frac{\tilde{s}}{\sqrt{n}} t_{n-1, \frac{\alpha_0}{2}} + b_2 \log \frac{1}{\alpha_2}$$

where $\alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 = \alpha$, and $t_{n-1, \frac{\alpha_0}{2}}$ is the $1 - \frac{\alpha_0}{2}$ quantile of the t distribution with $n-1$ degrees of freedom.

7: The confidence interval is

$$[\tilde{\mu} - w, \tilde{\mu} + w]$$

Algorithm 6

Input: $x_1, \ldots, x_{n_1}$, $y_1, \ldots, y_{n_2}$, α, ϵ, R, $\sigma_{\min}$.

Output: An $\epsilon = \epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4$ differentially private α level confidence interval of $\mu_1 - \mu_2$.

1: Run Algorithm 3 on $X_1, \ldots, X_{n_1}$ with $\epsilon = \epsilon_1$ and $\alpha = \alpha_1$ to obtain an estimate of scale σ_1^* and range $[X_{\min}, X_{\max}]$.

2: Run Algorithm 3 on $Y_1, \ldots, Y_{n_2}$ with $\epsilon = \epsilon_2$ and $\alpha = \alpha_2$ to obtain an estimate of scale σ_2^* and range $[Y_{\min}, Y_{\max}]$.

3: Truncate the samples $X_1, \ldots X_{n_1}$ and $Y_1, \ldots Y_{n_2}$ to lie in $[X_{\min}, X_{\max}]$ and $[Y_{\min}, Y_{\max}]$ respectively. Let $\hat{X}_i$ be the truncated version of X_i and $\hat{Y}_i$ be the truncated version of Y_i.

4: Let

$$\tilde{\mu}_1 = \frac{\sum_i \hat{X}_i}{n_1} + Z_3 \text{ and } \tilde{\mu}_2 = \frac{\sum_i \hat{Y}_i}{n_2} + Z_4$$

where Z_3 and Z_4 are Laplace random variables with mean 0 and scale parameters

$$b_3 = \frac{\sigma_1^* \sqrt{\log\left(\frac{n_1}{\alpha_1}\right)}}{\epsilon_3 n_1} \text{ and } b_4 = \frac{\sigma_2^* \sqrt{\log\left(\frac{n_2}{\alpha_2}\right)}}{\epsilon_4 n_2}$$

5: Truncate $\tilde{\mu}_1$ and $\tilde{\mu}_2$ to lie between $[X_{\min}, X_{\max}]$ and $[Y_{\min}, Y_{\max}]$ respectively.

6: Let

$$s_1^2 = \frac{\sum_i (\hat{X}_i - \tilde{\mu}_1)^2}{n_1 - 1} \text{ and } s_2^2 = \frac{\sum_i (\hat{Y}_i - \tilde{\mu}_2)^2}{n_2 - 1}$$

7: Let

$$\tilde{s}_1^2 = s_1^2 + Z_5 \text{ and } \tilde{s}_2^2 = s_2^2 + Z_6$$

where Z_5 and Z_6 are Laplace random variables with means 0 and scale parameters

$$b_5 = \frac{8\sigma_1^* \left(\sqrt{\log \frac{n_1}{\alpha_1}}\right)}{\epsilon_5 n_1} \text{ and } b_6 = \frac{8\sigma_2^* \left(\sqrt{\log \frac{n_2}{\alpha_2}}\right)}{\epsilon_6 n_2}$$

8: Let

$$\tilde{s}^2 = \frac{(n_1 - 1)\tilde{s}_1^2 + (n_2 - 1)\tilde{s}_1^2}{n_1 + n_2 - 1} + \frac{(n_1 - 1) b_5 \log \frac{1}{\alpha_5} + (n_2 - 1) b_6 \log \frac{1}{\alpha_6}}{n_1 + n_2 - 1}$$

9:

$$w = \frac{\tilde{s}}{\sqrt{n_1 + n_2}} t_{n_1+n_2-1, \frac{\alpha_0}{2}} + b_3 \log \frac{1}{\alpha_3} + b_4 \log \frac{1}{\alpha_4}$$

where $\alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 + \alpha_6 = \alpha$, and $t_{n_1+n_2-1, \frac{\alpha_0}{2}}$ is the $1 - \frac{\alpha_0}{2}$ quantile of the t distribution with $n_1 + n_2 - 1$ degrees of freedom.

10: The confidence interval is

$$[\tilde{\mu} - w, \tilde{\mu} + w]$$

Algorithm 7

Input: $(X_1, Y_1), \ldots, (X_n, Y_n)$, α, ϵ, R, $\sigma_{\min}$.

Output: An ϵ differentially private α level confidence interval of $\mu_1 - \mu_2$ using the paired samples.

1: Let $D_i = X_i - Y_i$.

2: Apply Algorithm 5 on input $D_1, \ldots, D_n$.

4.2 R Implementations of the Algorithms

The following programs were used to run the simulations of the algorithms discussed above.

```
#####################################################
# Classical CI for unknown mean with known variance
#####################################################
f.ci.np = function(X, sig, q) {
n = length(X)
ci.np = c(mean(X) - q*(sig/sqrt(n)), mean(X) + q*(sig/sqrt(n)))
ci.np.length = abs(ci.np[2] - ci.np[1])
return(list(ci.np, ci.np.length))
}
```

```
#install.packages("distr")
library(distr)

##################################################
# Algorithm 1: Range algorithm with known variance
##################################################
f.range.dp = function(X, eps, a, R, sig) {
n = length(X)

#### Step 1 and Step 2 ####
X[X < (-R-sig/2) | X > (R+sig/2)] = NA
hist.X = hist(X, breaks=seq(-R-sig/2, R+sig/2+sig, sig), plot=F)

#### Step 3 ####
b = 2/eps  # historgram sensitivity
k = length(hist.X$counts)  # number of bins
hist.X$noisy_counts = hist.X$counts + r(DExp(rate=1/b))(k)  # add laplace noise to each bin
max_noisy_bin = which(hist.X$noisy_counts == max(hist.X$noisy_counts))  # find bin with largest
sig_l.hat = hist.X$mids[max_noisy_bin]  # set sig*l.hat as the midpoint of the largest noisy bi

#### Step 4 ####
w = sig*sqrt(2*log(2/(1-(1-a)^(1/n)))) + sig/2
return (c(sig_l.hat - w,  sig_l.hat + w))
}

#install.packages("distr")
library(distr)

################################
# Algorithm 2: DP scale estimate
################################
f.scale.dp = function(X,e,a,R,s) {

#### Step 1 ####
i.min = log2(s)
i.max = log2(R)

breaks = rep(NA,length(i.min:i.max))

j = 1
for (i in i.min:(i.max+1)) {
breaks[j] = (2^i)
j = j+1
}

#### Step 2 ####
X = X[X != X[sample(1:length(X),1)]]
Y = rep(NA,length(X)/2)
```

```
j = 1
for (i in 1:(length(X)/2)) {
Y[i] = abs(X[j]-X[j+1])
j = j+2
}

#### Step 3 ####
Y[Y < breaks[1] | Y > breaks[length(breaks)]] = NA
hist.Y = hist(Y,breaks=breaks,plot=F)

#### Step 4 ####
f = function(k) {
k + r(DExp(rate=1/(2/e)))(1)
}
hist.Y$counts = sapply(hist.Y$counts, f)
i.star = hist.Y$breaks[[which(hist.Y$counts == max(hist.Y$counts))]]

#### Step 5 ####
return(2^(log2(i.star) + 1))
}

#install.packages("distr")
library(distr)
source("Alg_1.R")
source("Alg_2.R")

######################################################
# Algorithm 3: DP range estimate with unknown variance
######################################################
f.range2.dp = function(X,e,a,R,s) {
scale.dp = f.scale.dp(X,0.5*e,a,R,s)
range.dp = f.range.dp(X,0.5*e,a,R,scale.dp)
return(list(scale.dp,range.dp))
}

#install.packages("NormalLaplace")
#install.packages("distr")
library(NormalLaplace)
library(distr)
source("Alg_1.R")

##########################################################
#Algorithm 4: DP CI algorithm for mean with known variance
##########################################################
f.ci.dp = function(X, a, eps, sig, R) {
n = length(X)

#### Partition epsilon values ####
if (length(eps) == 1) {
```

```
# partition a into a[1] and a[2]
eps = c(eps/4, 3*eps/4)
}
else if (length(eps) > 2 | length(eps) < 1) {
# error
print('err')
}

#### Partition alpha values ####
if (length(a) == 1) {
f.w = function(v) {
a.1 = v[1]
a.2 = a - a.1
b=2*(sig*sqrt(2*log(2/(1-(1-a.2/3)^(1/n)))) + sig/2) / (eps[2]*n)
qnl(1-a.1/2, mu=0, sigma=sig/sqrt(n), alpha=1/b, beta=1/b)
}
ui=matrix(c(
1,
-1))
ci=c(0, -a)
opt_res = constrOptim(theta=c(a/2), f.w, NULL, ui=ui, ci=ci)
a = c(opt_res$par, a - opt_res$par)
}
else if (length(a) > 2 | length(a) < 1) {
# error
print('err')
}


#### Step 1 ####
range.dp = f.range.dp(X, eps[1], a[2], R, sig)

#### Step 2 ####
# Winsorize data
X[X < range.dp[1]] = range.dp[1]
X[X > range.dp[2]] = range.dp[2]

#### Step 3 ####
b = 2*(sig*sqrt(2*log(2/(1-(1-a[1]/3)^(1/n)))) + sig/2) / (eps[2]*n)
mean.dp = sum(X)/n + r(DExp(rate=1/b))(1)

#### Step 4 ####
# Margin of error
w = qnl(1-a[1]/2, mu=0, sigma=sig/sqrt(n), alpha=1/b, beta=1/b)

#### Step 5 ####
ci.dp = c(mean.dp - w, mean.dp + w)
ci.dp.length = ci.dp[2] - ci.dp[1]
return(list(ci.dp, ci.dp.length))
```

```
}

#install.packages("distr")
library(distr)
source("Alg_3.R")

##########################################
# Algorithm 5: DP-CI with variance unknown
##########################################
f.ci2.dp = function(X,a,e,R,s) {

#### Step 1 ####
al3.call = f.range2.dp(X,e[1],a[2],R,s)
scale.hat = al3.call[[1]]
range.hat = al3.call[[2]]

#### Step 2 ####
Y = X[X>=range.hat[1] & X<=range.hat[2]]

#### Step 3 ####
b2 = 4*s*sqrt(log(length(Y)/a[2])) / (e[2]*length(Y))
mean.dp = mean(Y) + r(DExp(rate=1/b2))(1)

#### Step 4 ####
if (mean.dp<range.hat[1] | mean.dp>range.hat[2]) {
mean.dp = min(abs(mean.dp-range.hat[1]),abs(mean.dp-range.hat[2])  )
}

#### Step 5 ####
var1 = sum((Y-mean.dp)^2)/(length(Y)-1)
b3 = 8*s*sqrt(log(length(Y)/a[2])) / (e[3]*length(Y))
var.dp = var1 + r(DExp(rate=1/b3))(1) + b3*log(1/a[4])

#### Step 6 ####
w = sqrt(var.dp)/sqrt(length(Y)) * qt(1-a[1]/2,length(Y)-1) + b2*log(1/a[3])

#### Step 7 ####
ci = c(mean.dp-w,mean.dp+w)
ln = abs(ci[2]-ci[1])

return(list(ci,ln))
}

#install.packages("distr")
library(distr)
source("Alg_3.R")

##########################################
# Algorithm 6: DP difference of means test
```

```
##########################################
f.dom.dp <- function(X,Y,a,e,R,s) {

# Take a to be a vector of length 7.
# Take e to be a vector of length 6.

n1 <- length(X)
n2 <- length(Y)

#### Step 1 ####
al3.call <- f.range2.dp(X,e[1],a[2],R,s)
scaleX.hat <- al3.call[[1]]
rangeX.hat <- al3.call[[2]]

#### Step 2 ####
al3.call <- f.range2.dp(Y,e[2],a[3],R,s)
scaleY.hat <- al3.call[[1]]
rangeY.hat <- al3.call[[2]]

#### Step 3 ####
X <- X[X>=rangeX.hat[1] & X<=rangeX.hat[2]]
Y <- Y[Y>=rangeY.hat[1] & Y<=rangeY.hat[2]]

#### Step 4 ####
b3 <- scaleX.hat*sqrt(log(n1/a[2])) / (e[3]*n1)
b4 <- scaleY.hat*sqrt(log(n2/a[3])) / (e[4]*n2)
mean.X.dp <- mean(X) + r(DExp(rate=1/b3))(1)
mean.Y.dp <- mean(Y) + r(DExp(rate=1/b4))(1)

#### Step 5 ####
if (mean.X.dp<rangeX.hat[1] | mean.X.dp>rangeX.hat[2]) {
mean.X.dp <- min(abs(mean.X.dp-rangeX.hat[1]),abs(mean.X.dp-rangeX.hat[2]))
}

if (mean.Y.dp<rangeY.hat[1] | mean.Y.dp>rangeY.hat[2]) {
mean.Y.dp <- min(abs(mean.Y.dp-rangeY.hat[1]),abs(mean.Y.dp-rangeY.hat[2]))
}

#### Step 6 ####
varX <- sum((X-mean.X.dp)^2)/(n1-1)
varY <- sum((Y-mean.Y.dp)^2)/(n2-1)

#### Step 7 ####
b5 <- 8*scaleX.hat*sqrt(log(n1/a[2])) / (e[5]*n1)
b6 <- 8*scaleY.hat*sqrt(log(n2/a[3])) / (e[6]*n2)
var.X.dp <- varX + r(DExp(rate=1/b5))(1)
var.Y.dp <- varY + r(DExp(rate=1/b6))(1)

#### Step 8 ####
```

```
var.global <- ((n1-1)*var.X.dp + (n2-1)*var.Y.dp)/(n1+n2-1) + ((n1-1)*b5*log(1/a[6])+(n2-1)*b6*

#### Step 9 ####
w <- sqrt(var.global)/sqrt(n1+n2) * qt(1-a[1]/2,n1+n2-1) + b3*log(1/a[4]) + b4*log(1/a[5])

#### Step 10 ####
mu.dp <- mean.X.dp - mean.Y.dp

# True interval
true.int <- t.test(X,Y,alternative="two.sided",paired=F,var.equal=F,conf.level=0.95)$conf.int

return (list(c(mu.dp-w,mu.dp+w),c(true.int[1],true.int[2])))
}

#install.packages("mvtnorm")
library(mvtnorm)
source("Alg_5.R")

#####################################
#Algorithm 7: DP CI for paired t-test
#####################################
f.al7 <- function(X,a,e,R,s.min) {

D <- rep(NA,length(X)/2)

#### Step 1 ####
for (i in 1:length(D)) {
D[i] <- X[i,1] - X[i,2]
}

#### Step 2 ####
# Choose alpha and epsilon values at some point
return(f.ci2.dp(D,a[1],a[2],a[3],a[4],e[1],e[2],e[3],e[4],R,s.min))
}
```

4.3 R Simulation and Plotting Code

The following scripts were run to produce the plots shown and discussed above.

```
#install.packages("distr")
library(distr)
source("CI_NP_1.R")
source("Range_DP_1.R")
source("CI_DP_1.R")
set.seed(42)


## The following codes tests the coverage probabilities of the two
```

```
## types of confidence intervals. The plotting code runs slowly.

sam.size <- 5000

##########################################
# Computes coverage probability for np-cis
# Optional: plots cis
##########################################
f.coverage.np <- function(sam.size) {

cis.np <- replicate(100,f.ci.np(rnorm(sam.size),1,1.96)[[1]])
cis.np.cover <- rep(0,100)

for (i in 1:100){
if ((0>=cis.np[1,i])&(0<=cis.np[2,i])) {
cis.np.cover[i] <- 1
}
}
return(sum(cis.np.cover)/100)
}

##############################################
# Plots np coverage probability by sample size
##############################################
s <- seq(1,sam.size,10)
p.cov.np <- rep(NA,length(s))
for (i in 1:length(s)) {
p.cov.np[i] <- f.coverage.np(i)
}
plot(s,p.cov.np,type="l",col="aquamarine3",main="Coverage Probability v. n",
xlab="Sample Size",ylab="Coverage Probability")
segments(1,mean(p.cov.np),sam.size,mean(p.cov.np))

##########################################
# Computes coverage probability for dp-cis
# Optional: plots cis
##########################################
f.coverage.dp <- function(sam.size) {

cis.dp <- replicate(100,f.ci.dp(rnorm(sam.size),0.02,0.02,0.01,0.1,0.1,1,4)[[1]])
cis.dp.cover <- rep(0,100)

for (i in 1:100){
if ((0>=cis.dp[1,i])&(0<=cis.dp[2,i])) {
cis.dp.cover[i] <- 1
}
}
return(sum(cis.dp.cover)/100)
}
```

```
##############################################
# Plots dp coverage probability by sample size
##############################################
p.cov.dp <- rep(NA,length(s))
for (i in 1:length(s)) {
p.cov.dp[i] <- f.coverage.dp(i)
}
plot(s,p.cov.dp,type="l",col="cornsilk4",main="Coverage Probability v. n",
xlab="Sample Size",ylab="Coverage Probability")
segments(1,mean(p.cov.dp),sam.size,mean(p.cov.dp))

#######################
# Plot np-cis against n
#######################
plot(NA,xlim=c(0,sam.size),ylim=c(-1,1),
main="NP-CIs v. n",xlab="Sample Size",ylab="Confidence Intervals")
for (i in 1:sam.size) {
ci.np <- f.ci.np(rnorm(i),1,1.96)
segments(i,ci.np[[1]][1],
i,ci.np[[1]][2],col="burlywood")
points(i,ci.np[[2]],pch=46,col="darkgreen")
}
segments(1,0,sam.size,0,col="black")

#######################
# Plot dp-cis against n
#######################
plot(NA,xlim=c(0,sam.size),ylim=c(-20,20),
main="DP-CIs v. n",xlab="Sample Size",ylab="Confidence Intervals")
for (i in 1:sam.size) {
ci.dp <- f.ci.dp(rnorm(i),0.02,0.02,0.01,0.1,0.1,1,4)
segments(i,ci.dp[[1]][1],
i,ci.dp[[1]][2],col="antiquewhite3")
points(i,ci.dp[[2]],pch=46,col="darkgreen")
}
segments(1,0,sam.size,0,col="black")

###################
# Comparing lengths
###################
sizes <- c(1,10,50,100,500,1000,5000,10000,100000,1000000,10000000)

np.lengths <- rep(NA, length(sizes))
for (i in 1:length(sizes)) {
np.lengths[i] <- f.ci.np(rnorm(sizes[i]),1,1.96)[[2]]
}

dp.lengths <- rep(NA, length(sizes))
```

```
for (i in 1:length(sizes)) {
dp.lengths[i] <- f.ci.dp(rnorm(sizes[i]),0.02,0.02,0.01,0.1,0.1,1,4)[[2]]
}

lengths.dif <- abs(dp.lengths - np.lengths)
lengths.rat <- dp.lengths/np.lengths

options("scipen"=100, "digits"=6)

data.frame(n=sizes,
NP_Length=np.lengths,
DP_Length=dp.lengths,
Dif_Len=lengths.dif,
Rat_Len=lengths.rat)

#install.packages("NormalLaplace")
#install.packages("distr")
library(NormalLaplace)
library(distr)
source("Alg_4.R")

## Checks coverage for Algorithm 4
f.al4.cov = function(n, mu) {
eps = c(0.05, .15)
a = 0.05 #c(0.0455, 1-.0455)
R = 4
mu = 0
sig = 1

reps = 1000
covered = 0
for (i in 1:reps) {
X = rnorm(n=n, mean=mu, sd=sig)
ci = f.ci.dp(X, a, eps, sig, R)[[1]]
if (ci[1] <= mu & ci[2] >= mu) {
covered = covered + 1
}
}

return(covered/reps)
}

#install.packages("NormalLaplace")
#install.packages("distr")
library(NormalLaplace)
library(distr)
source("Alg_4_Coverage.R")

# Set values
```

```
mu  = 0
n   = 5000
s   = seq(10,n,10)
cov = rep(NA, length(s))

# Populate coverage vector
for (i in 1:length(s)) {
cov[i] = f.al4.cov(s[i],mu)
}

# Plot coverage probabilities
plot(s,cov,type="l",col="aquamarine3",main="Coverage Probability v. n",
xlab="Sample Size",ylab="Coverage Probability")

# Plot mean coverage
segments(1,mean(cov),n,mean(cov))

#install.packages("NormalLaplace")
#install.packages("distr")
library(NormalLaplace)
library(distr)
source("Alg_0 - Classical_CI_Known_Var.R")
source("Alg_4.R")

## Creates table of length information

# Set constants
a0 = 0.048#0.02
a1 = 0.002#0.02
a2 = 0.000#0.01
e1 = 0.01#0.1
e2 = 0.19#0.1
s  = 1
R  = 4

# Sample sizes
sizes = c(50,100,500,1000,5000,10000,100000,1000000,10000000)
k = length(sizes)

# Get ci lengths
lns.np = rep(NA,k)
lns.dp = rep(NA,k)

for (i in 1:k) {
X = rnorm(sizes[i])
lns.np[i] = f.ci.np(X, s, qnorm(0.975))[[2]]
lns.dp[i] = f.ci.dp(X, a0, a1, a2, e1, e2, s, R)[[2]]
}
```

```
# Get differences and ratios
lns.dif = abs(lns.dp - lns.np)
lns.rat = lns.dp/lns.np

# Set decimal preferences
options("scipen"=100, "digits"=6)

# Create table
data.frame(n=sizes,
NP_Length=lns.np,
DP_Length=lns.dp,
Dif_Len=lns.dif,
Rat_Len=lns.rat)

#install.packages("NormalLaplace")
#install.packages("distr")
library(NormalLaplace)
library(distr)
source("Alg_5.R")

## Checks coverage for Algorithm 5
f.al5.cov <- function(n,mu) {

# Indexed: ci.data [(ci=1,ln=2), (replication num)]
ci.data <- replicate(100,f.ci2.dp(rnorm(n),0.02,0.01,0.01,0.01,0.1,0.1,0.1,0,4,1))

ci.ub <- rep(NA,100)
ci.lb <- rep(NA,100)

# Get ci bounds
for (i in 1:100) {
ci.ub[i] <- ci.data[1,i][[1]][2]
ci.lb[i] <- ci.data[1,i][[1]][1]
}

# Find coverage
ci.cov <- rep(0,100)

for (i in 1:100) {
if (ci.lb[i] <= mu & ci.ub[i] >= mu) {
ci.cov[i] <- 1
}
}

return(sum(ci.cov)/100)
}

#install.packages("NormalLaplace")
#install.packages("distr")
```

```
library(NormalLaplace)
library(distr)
source("Alg_5_Coverage.R")

# Set values
mu  <- 0
nl  <- 9000
nu   <- 10000
s   <- seq(nl,nu,10)
cov <- rep(NA, length(s))

# Populate coverage vector
for (i in 1:length(s)) {
cov[i] <- f.al5.cov(s[i],mu)
}

# Plot coverage probabilities
plot(s,cov,type="l",col="chocolate4",main="Coverage Probability v. n",
xlab="Sample Size",ylab="Coverage Probability")

# Plot mean coverage
segments(nl,mean(cov),nu,mean(cov))


#install.packages("NormalLaplace")
#install.packages("distr")
library(NormalLaplace)
library(distr)
source("Alg_0 - Classical_CI_Known_Var.R")
source("Alg_5.R")

## Creates table of length information

# Set constants
a0 <- 0.02
a1 <- 0.01
a2 <- 0.01
a3 <- 0.01
e1 <- 0.1
e2 <- 0.1
e3 <- 0.1
e4 <- 0
s  <- 1
R  <- 4

# Sample sizes
sizes <- c(50,100,500,1000,5000,10000,100000,1000000,10000000)
k <- length(sizes)
```

```
# Get ci lengths
lns.np <- rep(NA,k)
lns.dp <- rep(NA,k)

for (i in 1:k) {
X = rnorm(sizes[i])

# Get non-private ci
ci.np <- t.test(X)[[4]]

# Get non-private and private lengths
lns.np[i] <- abs(ci.np[[2]]-ci.np[[1]])
lns.dp[i] <- f.ci2.dp(X,a0,a1,a2,a3,e1,e2,e3,e4,R,s)[[2]]
}

# Get differences and ratios
lns.dif <- abs(lns.dp - lns.np)
lns.rat <- lns.dp/lns.np

# Set decimal preferences
options("scipen"=100, "digits"=6)

# Create table
data.frame(n=sizes,
NP_Length=lns.np,
DP_Length=lns.dp,
Dif_Len=lns.dif,
Rat_Len=lns.rat)
```