cs208 HW 3

Anthony Rentsch

4/2/2019

Note: I include the code I wrote directly in line for some questions that ask only to implement a method. For other questions, my code can be found in the Appendix and on Github. I worked with Bhaven Patel and Lipika Ramaswamy for this assignment.

Question 1

(a) To prove that this mechanism is ϵ-DP, I will show that (i) the percentile trimming transformation is 1-Lipschitz, (ii) that the Laplace noise injection mechanism is ϵ-DP, and (iii) that this implies that the entire mechanism $M(x)$ is $(1 * \epsilon)$-DP.

(i) A mapping T from dataset to dataset is c-Lipschitz iff $\forall x, x'\ d(T(x), T(x')) \leq c * d(x, x')$. Here let's consider that x and x' only differ on one element. It follows that $d(x, x') = 1$.

Now consider the percentile trimming transformation in this mechanism. It again follows that $d(T(x), T(x')) = 1$ since the maximum number of rows that these two datasets will differ on is 1. Returning the inequality in the definition of a Lipschitz constant, we see that this transformation is 1-Lipschitz.

(ii) First, we observe that

$$\frac{1}{.9n} \sum_{P_{.05} \leq x \leq P_{0.95}} x_i$$

is simply an estimator for the mean of x after trimming the bottom and top 5% of the data. For simplicity, replace $.9n$ with n' and call this mechanism M'. Note that the global sensitivity of this query is $GS_q = D/n'$. Since the Laplace noise is scaled by $\frac{GS_q}{\epsilon}$, M' is ϵ-DP.

(iii) In class, we discussed a lemma that states that if M is ϵ-DP and T is c-Lipschitz, then $M \circ T$ is $(c * \epsilon)$-DP. Following from (i) and (ii), we then have that $M = M' \circ T$ is $(1 * \epsilon)$-DP.

Below is the implementation of this mechanism.

```
sgn <- function(x) {     # function borrowed from class
  return(ifelse(x < 0, -1, 1))
}

rlap = function(mu=0, b=1, size=1) {     # function borrowed from class
  p <- runif(size) - 0.5
  draws <- mu - b * sgn(p) * log(1 - 2 * abs(p))
  return(draws)
}

trimmedMean <- function(x, d, n, epsilon) {
  scale <- d/(epsilon*0.9*n)
  quants <- quantile(x, c(0.05,0.95))
  x_trimmed <- x[x>quants[1] && x<quants[2]]
  mean_trimmed <- (1/(0.9*epsilon*n))*sum(x_trimmed)
  mean_release <- mean_trimmed + rlap(mu=0,b=scale,size=1)
  return(mean_release)
}
```

(b) Let's first consider the Lipschitz constant of the transformation $[x]_{P_{05}}^{P_{95}}$.

- this must be 2-Lipschitz

(c)

- describe mechanism
- implment with code

(d)

- code

(e)

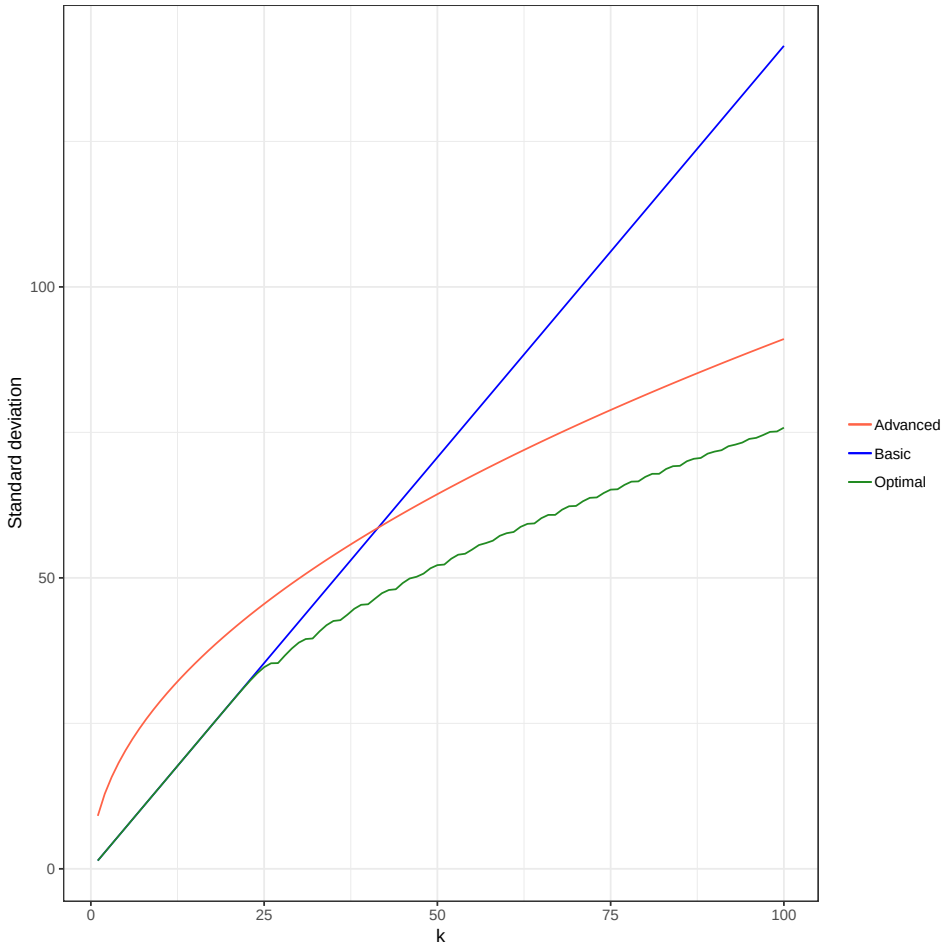
- informal sketch of proof

(f)

- have to re-implement laplace clamped mean from last hw
- code + plots
- description of results/ situation in which this would be a good approach

Question 2

The "optimal" composition theorem strictly improves upon the standard deviation of the injected noise from the basic composition theorem when $k \geq 17$ and advanced composition strictly improves upon basic composition when $k \geq 42$. Up to $k = 17$, basic and "optimal" composition correspond to roughly the same standard deviation. The standard deviation from advanced composition is strictly larger than the standard deviation from "optimal"" composition, but the ratio between the two appears to remain constant as $k \to \infty$.



Question 3

Here, I examine the utility of synthetic data generated using the DP histogram approach. To do this, I (1) generate synthetic data by adding Laplace distributed noise to the *age* x *education* x *income* bins counts for the PUMS data, (2) sample 10,000 synthetic observations based on these bin counts, (3) estimate a regression model relating age and education to income, and (4) compute the mean squared error, squared bias, and variance of the coefficient estimates for age, education, and the intercept.

As a pre-process to this whole procedure, I clip the income variable to 1 so that I can work with the log of the income. I then use the log of the clipped income for the remainder of the problem, i.e., to compute the sensitive regression on the entire dataset and to produce the histogram release and corresponding synthetic data.

The table below summarizes my results. **findings here**

Metric	Intercept	Education	Age
DP MSE	0.2181647	0.0000379	0.0004556
DP Variance	0.0067352	0.0000010	0.0000758
DP Biasˆ2	0.2114295	0.0000370	0.0003797
Sampling MSE	0.2174255	0.0000242	0.0010317

BONUS

Appendix

I put the code for all of my analyses here. You can also find it on Github.

```
# Set up
require(plyr); require(dplyr); require(ggplot2)
pums <- read.csv("MaPUMS5full.csv")
```

Question 1

```
# code to be added
```

Question 2

```
# parameters
global_epsilon = 1
global_delta = 10^(-9)
global_sens = 1
max_k = 100

# Laplace sd
laplaceSD <- function(epsilon) { return((sqrt(2)/epsilon)) }

# basic
basicComposition <- function(epsilon, k) { return(epsilon/k) }

# advanced
advancedComposition <- function(epsilon, k, delta) { return(epsilon/sqrt(2*k*log(1/delta))) }

# optimal
# use PSIlence:::update_parameters

# compute sds
sds <- matrix(NA, nrow=100, ncol=4)
for (k in 1:max_k) {
  epsilon_comp <- basicComposition(global_epsilon, k)
  epsilon_adv <- advancedComposition(global_epsilon, k, global_delta)

  init <- rep(c(1/k, 0), k)
  params <- matrix(init, nrow=k, ncol=2, byrow=TRUE)
  inverse <- PSIlence:::update_parameters(params, hold=0, eps=global_epsilon, del=global_delta)
  epsilon_opt <- max(inverse[,1])

  sds[k,1] <- laplaceSD(epsilon_comp)
  sds[k,2] <- laplaceSD(epsilon_adv)
  sds[k,3] <- laplaceSD(epsilon_opt)
  sds[k,4] <- k
}

sds_df <- data.frame(sds)
names(sds_df) <- c("basic", "advanced", "optimal", "k")
```

```
q2_plot <- ggplot(sds_df) + geom_line(aes(x=k, y=basic, colour="Basic")) +
  geom_line(aes(x=k, y=advanced, colour="Advanced")) +
  geom_line(aes(x=k, y=optimal, colour="Optimal")) +
  labs(x="k", y="Standard deviation") +
  scale_colour_manual(values=c("#FF6347", "#0000FF", "#228B22")) +
  theme_bw() +
  theme(legend.title = element_blank())
pdf("plots/q2_plot.pdf", width=8, height=8)
q2_plot
dev.off()

# when do advanced/optimal beat basic?
paste0("Advanced: ", min(sds_df$k[sds_df$advanced < sds_df$basic]))
paste0("Optimal: ", min(sds_df$k[sds_df$optimal < sds_df$basic]))
```

Question 3

```
# code to be added
```