Multi-party DP Write-up 1
Linear Least Squares

Notations.

Let $D = [\mathbf{X}, \mathbf{y}]$ be a database of size n, with $\mathbf{X} \in \mathbb{R}^{n\times d}$ and $\mathbf{y} \in \mathbb{R}$. $[n]$ stands for the set $\{1, \ldots, n\}$. We use $z \sim N(\mu, \sigma)$ to denote a variable following the Gaussian distribution with mean μ and standard deviation σ.

Linear Least Squares .

In a linear regression model the response variable is a linear function of the regressors:

$$\mathbf{y} = \mathbf{X}\theta + \epsilon \tag{1}$$

where ϵ denote the error terms.

The loss function under linear least squares model is defined as

$$L(\theta; D) = \Sigma_{i\in[n]}(\theta \cdot \mathbf{x_i} - \mathbf{y_i})^{\mathbf{2}} \tag{2}$$

where $\cdot$ is the inner product of two vectors.

Linear least squares model has a nice closed form expression for θ, such that its loss function $L(\theta; D)$ is minimized.

$$\theta = (\mathbf{X}^\top\mathbf{X})^{\mathbf{-1}}\mathbf{X}^\top\mathbf{y} \tag{3}$$

Distributed Computing Problem

Let $A_1, \ldots A_t$ and B be $t+1$ parties (agents with computing power). Each A_i owns some of the data $\{D_j\}_{j\in S_i}$, where $S_i \subseteq [n]$. For simplicity, we assume $\{S_1, \ldots, S_t\}$ partition $[n]$. The task for them is to jointly compute θ under the linear least squares model, such that the entire computation is differential-private (both intermediate output and final output should be differential private).

Privacy & Utility

There is a trade-off between privacy and utility in any mechanism.

1. We use (ϵ, δ)-differential privacy to evaluate the privacy level of our algorithm.

2. We use the concept of worst case (over input data) *expected excess risk* to evaluate the utility of our algorithm.

Definition 1. *Given loss function $L(\theta; D)$ over database D, its worst case expected excess risk is defined as*

$$E[L(\theta; D) - L(\theta^*; D)] \tag{4}$$

where θ is the output of the algorithm, θ^* is the optimal minimizer of the loss function and the randomness comes from the algorithm.

Naive Algorithm.

1. Each A_i computes $(\mathbf{C_i}, \mathbf{d_i})$=$(\Sigma_{j \in S_i} \mathbf{x_j} \mathbf{x_j}^\top + z_1^{(d \times d)}, \Sigma_{j \in S_i} \mathbf{x_j} y_j + z_2^{(d)})$ and transfer it to party B, where $z_1 \sim N(0, \sigma_1)$ and $z_2 \sim N(0, \sigma_2)$.
2. Agent B computes $(\Sigma_{i \in [t]} \mathbf{C_i})^{-1} (\Sigma_{i \in [t]} \mathbf{d_i})$ and output it as the solution θ.

Analysis of the Naive Algorithm with $d = 1$

Definition 2. *The sensitivity of function f is defined as $\Delta(f) = sup_{(D,D')}(|f(D) - f(D')|_2)$, where (D, D') is a pair of neighboring databases.*

Theorem 1 (DP with Gaussian noise)**.** *Let f be the target function, and $z \sim N(0, \sigma)$. Let $F(D) = f(D) + z$ be the output function. Fix any $\delta > 0$, we have F is (ϵ, δ)-DP, where $\epsilon = \frac{\Delta(f)\sqrt{2 \ln(2/\delta)}}{\sigma}$.*

Privacy Analysis
Let $c(D) = \Sigma_{j \in [n]} x_j^2$ and $d(D) = \Sigma_{j \in [n]} x_j y_j$.

1. Each A_i's algorithm is $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-DP, where $\epsilon_1 = \frac{\Delta(c)\sqrt{2 \ln(2/\delta_1)}}{\sigma_1}$ and $\epsilon_2 = \frac{\Delta(d)\sqrt{2 \ln(2/\delta_2)}}{\sigma_2}$.

   When $d = 1$, each A_i computes and outputs (c_i, d_i)=$(\Sigma_{j \in S_i} x_j^2 + z_1$, $\Sigma_{j \in S_i} x_j y_j + z_2)$, where $z_1 \sim N(0, \sigma_1)$ and $z_2 \sim N(0, \sigma_2)$. It is obvious that $\Delta(c_i) = \Delta(c)$ and $\Delta(d_i) = \Delta(d)$ for all $i \in [t]$.

   Fix $\delta_1 > 0$, c_i is (ϵ_1, δ_1)-DP, where $\epsilon_1 = \frac{\Delta(c)\sqrt{2 \ln(2/\delta_1)}}{\sigma_1}$. Similarly, d_i is (ϵ_2, δ_2)-DP, where $\epsilon_2 = \frac{\Delta(d)\sqrt{2 \ln(2/\delta_2)}}{\sigma_2}$. By the composition theorem of differential privacy, each A_i's algorithm is $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-DP.

2. B's algorithm.

   It is easy to see B's output is

$$\theta = \frac{\Sigma_{i\in[n]} x_i y_i + t z_2}{\Sigma_{i\in[n]} x_i^2 + t z_1}. \tag{5}$$

$$= \frac{d(D)}{c(D) + t z_1} + \frac{t z_2}{c(D) + t z_1}. \tag{6}$$

   We only consider the case when $(X^\top X)$ is invertible. Here, we assume WLOG that $c(D) = \Sigma_{i\in[n]} x_i^2 \geq b_l > 0$. We treat $\frac{d(D)}{c(D)+tz_1}$ as our target function and $\frac{tz_2}{c(D)+tz_1}$ as the noise.

   Pick b_z such that $0 < b_z < b_l$, let $1 - \delta_z$ be the probability that $tz_1 \in [-b_z, b_z]$. Then, with probability $1 - \delta_z$, $c(D) + tz_1 \geq b_l - b_z$, and therefore $\Delta(\frac{d(D)}{c(D)+tz_1}) \leq \Delta(d)/(b_l - b_z)$.

   We need to separate the discussion on the upper bound of $c(D)$.

   (a) If $c(D)$ does not have an upper bound, $\frac{tz_2}{c(D)+tz_1}$ could be 0 and B's algorithm does not have privacy guarantee. We need to modify B's algorithm by adding $z_3 \sim N(0, \sigma_3)$ to θ.

   (b) If $c(D) \leq b_u$. Then, with probability $1 - \delta_z$, $c(D) + tz_1 \leq b_u + b_z$ and therefore $\frac{tz_2}{c(D)+tz_1} \geq \frac{tz_2}{b_u+b_z}$. It is obvious that $\frac{tz_2}{b_u+b_z} \sim N(0, \frac{t}{b_u+b_z}\sigma_2)$.

   Fix $\delta_3 > 0$, we have B's algorithm is $(\epsilon_3, \delta_z + \delta_3)$-DP, where $\epsilon_3 \leq \frac{(\Delta(d)/(b_l-b_z))\sqrt{2\ln(2/\delta_3)}}{\frac{t}{b_u+b_z}\sigma_2} = \frac{\Delta(d)(b_u+b_z)\sqrt{2\ln(2/\delta_3)}}{(b_l-b_z)t\sigma_2}$

Utility Analysis

We calculate $E[L(\theta; D) - L(\theta^*; D)]$, where θ^* is the optimal minimizer to the loss function. In the following discussion, we assume $c(D) = \Sigma_{i\in[n]} x_i^2$ has some upper bound ($c(D) \leq b_u$), and therefore B's algorithm has certain level of privacy guarantee. In the case when $c(D)$ does not have an upper bound, it is easy to see that adding another $z_3 \sim N(0, \sigma_3)$ to θ will bring extra $\sigma_3^2 c(D)$ to the expected excess risk.

It is known that $\theta^* = \frac{d(D)}{c(D)}$ under the linear least model.

$$E[L(\theta; D) - L(\theta^*; D)] = E[\Sigma_{i\in[n]}(\frac{d(D) + tz_2}{c(D) + tz_1} x_i - y_i)^2 - (\frac{d(D)}{c(D)} x_i - y_i)^2] \tag{7}$$

We focus on its i-th term.

$$E[(\frac{d(D)+tz_2}{c(D)+tz_1}x_i - y_i)^2 - (\frac{d(D)}{c(D)}x_i - y_i)^2] \tag{8}$$

$$= E[(\frac{(d(D)+tz_2)^2}{(c(D)+tz_1)^2}x_i^2 + y_i^2 - 2\frac{d(D)+tz_2}{c(D)+tz_1}x_i y_i] - (\frac{d(D)^2}{c(D)^2}x_i^2 + y_i^2 - 2\frac{d(D)}{c(D)}x_i y_i) \tag{9}$$

$$= E[\frac{(d(D)+tz_2)^2}{(c(D)+tz_1)^2}x_i^2] - \frac{d(D)^2}{c(D)^2}x_i^2 \tag{10}$$

$$= E[\frac{(d(D)^2 + t^2 z_2^2 + 2d(D)tz_2)}{(c(D)^2 + t^2 z_1^2 + 2c(D)tz_1)}x_i^2] - \frac{d(D)^2}{c(D)^2}x_i^2 \tag{11}$$

$$= (\frac{d(D)^2 + t^2\sigma_2^2}{c(D)^2 + t^2\sigma_1^2} - \frac{d(D)^2}{c(D)^2})x_i^2 \tag{12}$$

Hence, we have $E[L(\theta; D) - L(\theta^*; D)] = (\frac{d(D)^2+t^2\sigma_2^2}{c(D)^2+t^2\sigma_1^2} - \frac{d(D)^2}{c(D)^2})c(D)$.

Algorithm based on Stochastic gradient descent

1. B chooses θ_0 from the convex set $\mathcal{C}$, learning rate η and a permutation π over n copies of $[n]$. B broadcasts (θ_0, η, π)

2. In each iteration i, the agent A_j that owns data $D_{\pi(i)}$ will compute $\theta_{i+1} = \Pi_{\mathcal{C}}[\theta_i - \eta(i)(n\Delta(l(\theta_i; D_{\pi(i)})) + z_i)]$, where z_i is a zero mean Gaussian noise and $\Pi_{\mathcal{C}}$ is a projection function to set $\mathcal{C}$. A_j broadcasts $G_i = n\Delta(l(\theta_i; D_{\pi(i)}))$.

3. Each of the remaining A_k will update $\theta_{i+1} = \Pi_{\mathcal{C}}[\theta_i - \eta(i)G_i]$