New York University Tandon School of Engineering
Computer Science and Engineering

CS-GY 6763: Homework 3.
Due Tuesday, November 22th, 2022, 11:59pm.

Collaboration is allowed on this problem set, but solutions must be written-up individually. Please list collaborators for each problem separately, or write "No Collaborators" if you worked alone.

Problem 1: Differentially Private Odds Ratios.

(15 pts) When conducting private data release, we often want to calculate a given statistic with an ϵ-differential privacy guarantee. In class, we saw one way that we can calculate a *mean* in a differentially private manner. For this problem, we will consider calculating a differentially private *odds ratio*.

An *odds ratio* is a statistical measure that gives us a sense of the correlation between two events (usually binary). It is often used to assess treatment effects. In this problem, we will consider the scenario where a group of n individuals either were or were not vaccinated, and either did or did not contract COVID-19.

To be more concrete, consider a dataset X of individuals labeled with numbers $1, \ldots, n$. The dataset has two binary dimensions, $S \in \{0,1\}^n$ and $V \in \{0,1\}^n$. For some individual i, $S(i) = 1$ if individual i has had COVID, and $S(i) = 0$ otherwise. Similarly, $V(i) = 1$ if individual i was vaccinated, and $V(i) = 0$ otherwise.

Consider the following scalar valued queries:

$$A = \sum_{i=1}^{n} \mathbb{1}[S(i) = 1 \text{ and } V(i) = 1\}]$$
$$B = \sum_{i=1}^{n} \mathbb{1}[S(i) = 0 \text{ and } V(i) = 1\}]$$
$$C = \sum_{i=1}^{n} \mathbb{1}[S(i) = 1 \text{ and } V(i) = 0\}]$$
$$D = \sum_{i=1}^{n} \mathbb{1}[S(i) = 0 \text{ and } V(i) = 0\}]$$

Then, an *odds ratio* τ can be calculated as:

$$\tau = \frac{A/C}{B/D} \tag{1}$$

Intuitively, if $\tau = 1$, that suggests that S and V are uncorrelated, while a $\tau < 1$ suggests that vaccines are associated with lower "odds" of COVID, and a $\tau > 1$ suggests that vaccines are associated with higher "odds" of COVID.

(a) Show that the sensitivity of the τ statistic, which we denote $\Delta\tau = \max_{X,X'} |\tau(X) - \tau(X')|$ can grow as $\Omega(n)$, even if we assume say that $A, B, C, D > c$ for some constant c.

(b) That the sensitivity of τ is unbounded makes it difficult to directly add Laplace noise to ensure privacy without strong assumptions over our data. However, we can still calculate τ privately with a simple algorithm, with reasonable error. Show that there exists an ϵ-differentially private estimator $\hat{\tau}$ such that for a constant c:

$$Pr\left[\left(1 - c\epsilon \ln\frac{1}{\rho}\right) \leq \frac{\hat{\tau}}{\tau} \leq \left(1 + c\epsilon \ln\frac{1}{\rho}\right)\right] \geq 1 - \rho. \tag{2}$$

You may assume that ϵ is small – e.g. so that $\epsilon \ln\frac{1}{\rho} < 1$. You can also assume that A, B, C, D is each bigger than $\frac{1}{\epsilon^2}$.

Hint: Consider separate private estimators for the queries A,B,C and D.

Note: Previously we asked you to show the weaker bound:

$$Pr\left[\left|\frac{\hat{\tau}}{\tau}\right| > (1+c\epsilon)\cdot \ln\left(\frac{1}{\rho}\right)\right] \leq \rho. \tag{3}$$

If you already showed just that, then you will receive full credit.

(c) We've provided you with a synthetic X for a population of 100,000 people in `covid_data.csv`. In reality, it is often unfeasible to calculate a statistic like τ over the entire population (because we can't survey everyone), so let's understand how subsampling here affects the accuracy of τ in practice. Please implement each step yourself - no external libraries beyond *numpy, pandas,* etc. (or equivalents in your language of choice).

(a) Calculate τ for the entire population. This is ground truth τ.

(b) Calculate $\tilde{\tau}_m$ from random subsamples of size m from the population, where $m \in \{100, 1000, 10000\}$. Run this over $b = 10$ random subsamples (specify a seed set of size b) for each m, and report the average error between $\tilde{\tau}_m$ and τ across these runs.

(c) Calculate an ϵ-differentially private estimate $\hat{\tau}_m$ for each m using the algorithm you designed in (2) for $\epsilon = 0.01$. Report on the relative errors between $\hat{\tau}_m$, $\tilde{\tau}_m$ and τ.

(d) Write 2-3 sentences of discussion on how well the the algorithm performs in this practical setting. How does population size affect DP algorithm accuracy? Also, does having received the vaccine increase or decrease your odds of having contracted COVID?

Problem 2: Separation Oracles

(12 pts) Describe efficient separation oracles for each of the following families of convex sets. Here, "efficient" means linear time plus $O(1)$ calls to any additional oracles provided to you.

(a) The set $A \cap B$, given separation oracles for A and B.

(b) The ℓ_1 ball: $\|\mathbf{x}\|_1 \leq 1$.

(c) Any convex set A, given a *projection oracle* for A. Recall that a projection oracle, given a point $\mathbf{x}$, returns

$$\text{Proj}_A(\mathbf{x}) = \arg\min_{y \in A} \|\mathbf{x} - \mathbf{y}\|_2.$$

(d) The ϵ-neighborhood of a convex set A:

$$\{x \mid \exists y \in A, \|\mathbf{x} - \mathbf{y}\|_2 \leq \epsilon\}$$

given a projection oracle for A.

Above you may wish to use the following fact that was stated but not proven in class: for any point $\mathbf{x}$, convex set A, and $\mathbf{z} \in A$, $\|\mathbf{z} - \text{Proj}_A(\mathbf{x})\|_2 \leq \|\mathbf{z} - \mathbf{x}\|_2$.

Problem 3: Approximating Eigenvalues Moments

(15 pts) Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a square symmetric matrix, which means it it guaranteed to have an eigendecomposition with real eigenvalues, $\lambda_1 \geq \ldots \geq \lambda_n$. While computing these eigenvalues naively takes $O(n^3)$ time, it turns out that we can compute their *sum* much more quickly: with n operations. This is because $\sum_{i=1}^n \lambda_i$ is exactly equal to the trace of $\mathbf{A}$, i.e. the sum of its diagonal entries $\text{tr}(\mathbf{A}) = \sum_{i=1}^n \mathbf{A}_{ii}$. We can also compute the sum of squared eigenvalues in $O(n^2)$ time by taking advantage of the fact that $\sum_{i=1}^n \lambda_i^2 = \|\mathbf{A}\|_F^2$ where $\|\mathbf{A}\|_F^2$ is the Frobenius norm. What about $\sum_{i=1}^n \lambda_i^3$? Or $\sum_{i=1}^n \lambda_i^4$? It turns out that no exact algorithms faster than a full eigendecomposition are known.

In this problem, however, we show how to *approximate* $\sum_{i=1}^n \lambda_i^k$ for any positive integer k in $O(n^2 k)$ time. By the way, this is not a contrived problem – it has a ton of applications in machine learning and data science that you can ask me about in office hours!

(a) Show that $\sum_{i=1}^n \lambda_i^k = \text{tr}(\mathbf{A}^k)$ where $\mathbf{A}^k$ denotes the chain of matrix products $\mathbf{A} \cdot \mathbf{A} \cdot \ldots \cdot \mathbf{A}$, repeated k times. For the remainder of the problem we use the notation $\mathbf{B} = \mathbf{A}^k$.

(b) Let $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^n$ be m independent random vectors, all with i.i.d. $\{+1, -1\}$ uniform random entries. Let $Z = \frac{1}{m}\sum_{i=1}^m (\mathbf{x}^{(i)})^T \mathbf{B}\mathbf{x}^{(i)}$. We will show that Z is a good estimator for $\text{tr}(\mathbf{B})$ and thus for $\sum_{i=1}^n \lambda_i^k$. Give a short argument that Z can be computed in $O(n^2 km)$ time (recall that $\mathbf{B} = \mathbf{A}^k$).

(c) Prove that:

$$\mathbb{E}[Z] = \text{tr}(\mathbf{B}) \qquad \text{and} \qquad \text{Var}[Z] \leq \frac{2}{m}\|\mathbf{B}\|_F^2$$

Hint: Use linearity of variance but be careful about what things are independent!

(d) Show that if $m = O(\frac{1}{\epsilon^2})$ then, with probability 9/10,

$$|\,\text{tr}(\mathbf{B}) - Z| \leq \epsilon \|\mathbf{B}\|_F.$$

(e) Argue that, when $\mathbf{A}$ is positive semidefinite, $\epsilon\|\mathbf{B}\|_F \leq \epsilon\,\text{tr}(\mathbf{B})$, so the above guarantee actually gives the relative error bound,

$$(1-\epsilon)\,\text{tr}(\mathbf{B}) \leq Z \leq (1+\epsilon)\,\text{tr}(\mathbf{B}),$$

all with just $O(n^2 k/\epsilon^2)$ computation time.

Problem 4: Non-convex Optimization

(10 pts) Consider the problem of computing the top right singular vector $\mathbf{v}_1$ of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$. As mentioned, it is possible to frame this problem as an optimization problem and solve it with gradient descent. As discussed in class, a benefit of doing so is that it makes it easier to introduce stochastic and online methods, and possible use projection to add additional constraints.

Recall that $\mathbf{A}$'s right singular vectors are equal to the eigenvectors of $\mathbf{A}^T\mathbf{A}$ and the eigenvalues of $\mathbf{A}^T\mathbf{A}$ are equal to $\lambda_1 = \sigma_1^2, \ldots, \lambda_d = \sigma_d^2$, where $\sigma_1 > \ldots \sigma_d > 0$. You can assume there are no repeat singular values for this problem, so these are strict inequalities.

1. **Quick answer:** Assume we have know some coarse upper bound $\tilde{\lambda} \geq \lambda_1$. Let $f(\mathbf{x}) = \tilde{\lambda} \cdot \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}$. It is easy to check that $\mathbf{v}_1 = \arg\min_{\mathbf{x}\in\mathcal{S}} f(\mathbf{x})$ where $\mathcal{S} = \{\mathbf{y} : \|\mathbf{y}\|_2^2 \geq 1\}$. Prove that $f(\mathbf{x})$ is a convex function, but $\mathcal{S}$ is not a convex set.

   Since $\mathcal{S}$ is not convex, our generic analysis of projected gradient descent will give no guarantees for this problem. However, in class we will prove that the method actually does work for this problem – with a correctly chosen step size, it is exactly *exactly equivalent* to power method which we analyze.

2. Consider an alternative approach through unconstrained optimization. Let $g(\mathbf{x}) = -\frac{\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}}$. Now we have that $\mathbf{v}_1 \in \arg\min g(\mathbf{x})$. Prove that g is non-convex and derive an expression for its gradient $\nabla g(\mathbf{x})$. Show that $c \cdot \mathbf{v}_i$ is a statitionary point of g for any right singular vector $\mathbf{v}_i$ and scaling c.

3. g's non-convexity also rules out a direct convergence bound: in theory gradient descent could converge to any singluar vector of $\mathbf{A}$, not the top one. However, we can argue this is unlikely to happen. In particular, we claim that for any $i \neq 1$, $\mathbf{v}_i$ is actually just a *saddle point* of g, not a local minimum. To prove this, show that for any such $\mathbf{v}_i$, and *any* $t > 0$,

   $$\text{There exists a perturbation } \mathbf{z} \text{ with } \|\mathbf{z}\|_2 \leq t \text{ such that} \qquad g(\mathbf{v}_i + \mathbf{z}) < g(\mathbf{v}_i).$$

   If you are interested, you can find a some work on proving gradient methods won't get stuck at saddle points here https://arxiv.org/abs/1703.00887.