Privacy in Statistics and Machine Learning — Spring 2025
Homework 2: Due Wednesday, March 19, 2025

Adam Smith (based on materials developed with Jonathan Ullman)

Collaboration and Honesty Policy Reminder: Collaboration in the form of discussion is allowed. However, all forms of cheating (copying parts of a classmate's assignment, plagiarism from papers or old posted solutions) are NOT allowed. A rough rule of thumb: you should be able to walk away from a discussion of a homework problem with no notes at all and write your solution on your own. Finding answers to problems on the Web or from other outside sources (these include anyone not enrolled in the class) is forbidden.

- *You must write up each problem solution by yourself without assistance, even if you collaborate with others to solve the problem.*
- You must identify your collaborators. If you did not work with anyone, you should write "Collaborators: none."
- Asking and answering questions in every forum the class provides (on Piazza, in class, and in office hours) is encouraged!
- Even though looking up answers is forbidden, using the web, generative AI, or similar resources to find alternative explanations of concepts you need for the homework *is* allowed, and encouraged. Asking Deepseek to solve a problem for you is not ok; asking it for an explanation of Chernoff bounds or for examples of encoding quardatic constraints in Gurobi is fine. You must **document your use of outside sources** and describe it at a high level in your solutions.

Problems to be handed in

1. **(In-class exercise from lecture 6)** Suppose you have a graph with a fixed vertex set V, and where each individual data point x_i is an undirected edge $\{u, v\} \in V \times V$. For example, the nodes might represent locations, and an edge $\{u, v\}$ might represent the locations between which an individual travels most often.

   Consider the problem of finding a near-minimum cut in the graph. This is a partition of V into two disjoint sets A, B of nodes. The *weight* of the cut is the number of edges that cross from A to B (so $u \in A$ and $v \in B$ or vice versa). The weight of a cut can be as large as the size of the data set n, and n can be as large as $\Omega(|V|^2)$.

   (a) Use the exponential algorithm (or report noisy max) to design an algorithm that returns a cut with expected weight $\text{min-weight} + O(|V|/\varepsilon)$.
   It's OK if your algorithm runs in time polynomial in $2^{|V|}$.

   (b) There can be multiple distinct minimum cuts in a graph. However, one neat (and highly non-trivial to prove) fact is that if $w^* \geq 1$ is the number of edges in the minimum cut, the number of distinct cuts with weight $\leq cw^*$ is at most $O(|V|^{2c})$. Using this fact, prove that the error of the exponential mechanism (or RNM) is actually much better than the bound in part (a). Namely, it outputs a cut with expected weight $\text{min-weight} + O(\log(|V|)/\varepsilon)$.

   (c) Give an ε-differentially private algorithm A that runs in time polynomial in V, with the following guarantee: if the minimum cut in the input graph G has weight $w^* \geq \ln |V|$, then $A(G)$ returns a cut with weight $w^* + O(\log(|V|)/\varepsilon)$ with probability at least $1 - 1/|V|$.

2. (**More on node-private graph analysis**) Recall from Homework 1 that two graphs are *node neighbors* if one can be obtained from the other by removing a node and all of its edges. Let $f(G)$ denote the number of triangles in an undirected graph G. You showed in the homework that, on the set of graphs with at most n vertices, the global sensitivity of f is $\binom{n-1}{2}$.

   (a) Given a real-valued parameter $k > 0$ and a graph G, consider the following linear program:

Variables:	x_t for every triangle t in G	
Constraints:	For every triangle t,	$0 \le x_t \le 1$, and
	for every node u,	$\sum_{t:t \text{ contains } u} x_t \le k.$
Objective:	Maximize $\sum_t x_t$.	

   Let $f_k(G)$ denote the value of this linear program (that is, the maximum possible value of the objective function). Show that

      i. Show that the value of this linear program equals the number of triangles in G *if and only if* every node u is contained in at most k triangles; and

      ii. f_k has global sensitivity k (with no assumption on the neighboring inputs).

   So: one way to deal with the high sensitivity of f is to instead release an approximation to f_k. In graphs where no node is involved in too many triangles, this will in fact approximate f.

   (b) Let us now explore a different approach to dealing with the high sensitivity of f.

   Given a graph G and an integer $\lambda > 0$, let $m_\lambda(G)$ denote the smallest value achieved by $f(H)$ on all subgraphs of G that are obtained by removing up to λ nodes (and their edges) from G (where f is the number of triangles).

   One way to deal with the high sensitivity of f is, instead of aiming for an absolute error guarantee, to aim to release a value somewhere between $m_\lambda(G)$ and $f(G)$, for a value λ that isn't too big. This means, roughly, that we are returning the number of triangles in "almost all" of G.

      i. Show that increasing λ decreases $m_\lambda(G)$. That is, $m_{\lambda+1}(G) \le m_\lambda(G)$.

      ii. Fix a parameter $\varepsilon > 0$ and a positive integer τ. Consider the sequence

$$\vec{m}(G) = (m_\tau(G), m_{\tau-1}(G), \ldots, m_1(G), m_0(G))$$

      (where the last term $m_0(G)$ equals $f(G)$). Let G' be a neighboring graph obtained by adding a new node along with an arbitrary set of edges to G. Show that the sequences $\vec{m}(G)$ and $\vec{m}(G')$ are interleaved, that is:

$$m_\tau(G) \le m_\tau(G') \le m_{\tau-1}(G) \le m_{\tau-1}(G') \le \cdots \le m_1(G) \le m_1(G') \le m_0(G) \le m_0(G')\,.$$

      iii. Suppose we run the exponential mechanism to select a integer that is roughly a median of $\vec{m}(G)$ where G is the input graph. More precisely, suppose we known a public upper bound n on the size of G. Since f takes values in $\mathcal{Y} = \{0, 1, 2, ..., \binom{n}{3}\}$, we use $\mathcal{Y}$ as our set of pissble outputs, and score function $q(y; G) = |rank_{\vec{m}(G)(y)} - \frac{\tau}{2}$.
      Show that the score function has sensitivity 1 under insertion or deletion of nodes. Furthermore, show that if $\tau > 4\ln(n/\beta)/\varepsilon$, then this mechanism will, with probability at least $1 - \beta$ produce a value between $m_\tau(G)$ and G.

3. **(In class exercise from Lecture 6)** Show that the accuracy guarantees we showed for the exponential mechanism (and RNM) are basically tight in general. Specifically, consider the family of data sets $\left\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(d)}\right\}$ defined as follows: in $\mathbf{x}^{(j)}$, one candidate j receives $q_{\max} = n = \frac{\ln(d)}{2\varepsilon}$ votes and all others receive 0 votes.

   (a) Show that on such inputs, the algorithm A_{EM} will return a candidate other than j (that is, a candidate who received 0 votes) with constant probability, independent of d.

   (b) **(*)** Show that for **every** ε-differentially private A algorithm, if we choose J uniformly at random in $[d]$, then with constant probabililty $A(\mathbf{x}^{(J)})$ will return a candidate other than J. [That is, A will fail to find the winner for many datasets of this form.]

4. (Programming problem, TBA)