HW 6

Due on **Monday, April 29, 2024 at 10am**

CS 574: Randomized Algorithms, Spring 2024 Version: **1.0**

Submission guidelines same as previous homework.

16 (100 PTS.) JL Lemma works for angles.

Show that the Johnson-Lindenstrauss lemma also $(1 \pm \varepsilon)$-preserves angles among triples of points of P (you might need to increase the target dimension however by a constant factor).

(**Hint:** Use the fact that the constructed dimension reduction operator is a linear operator that maps lines to lines. Specifically, for every angle, construct a Isosceles triangle that its edges are being preserved by the projection (add the vertices of those triangles [conceptually] to the point set being embedded). Argue, that this implies that the angle is being preserved.)

17 (100 PTS.) Set System and Hitting.

Let $(U, \mathcal{F})$ be a set system, where $|U| = n$, and $\mathcal{F} \subseteq 2^U$, and $|\mathcal{F}| = O(n \log n)$. Furthermore, for every set $f \in \mathcal{F}$, we have that $|f| \geq 10 \log n$. Prove, that there is a subset $X \subseteq U$, such that $|X| = O(n \frac{\log \log n}{\log n})$, and for all $f \in \mathcal{F}$, we have $f \cap X \neq \emptyset$. (Hint: Course name.)

18 (100 PTS.) Estimate sum.

You are given a (multi)set of n number $X = \{x_1, \ldots, x_n\}$, all taken from $[\![m]\!] = \{1, \ldots, m\}$. You are given an oracle, that for any set $U \subseteq X$, and an interval $[\alpha.\beta]$ it returns if $U \cap [\alpha, \beta] \neq \emptyset$. Note, that you can not evaluate the value of x_i directly (you can of course figure out the value of x_i by doing a binary search using the oracle, for example, but that would be expensive).

You are also given parameters $\varepsilon, \delta \in (0, 1)$, describe a randomized algorithm that output a number Y, such that with probability $\geq 1 - \delta$, we have $(1 - \varepsilon)\tau \leq Y \leq (1 + \varepsilon)\tau$, where $\tau = \sum_i x_i$. Importantly, your algorithm should use as few oracle queries as possible. In particular, for credit, assuming ε is a constant, your algorithm should perform a polylogarithmic number of queries.

Hint: Consider the case that $m = 2$. Then solve for the case that $m = 4$, and so on. Of course, since m can be larger than n, you do not want to have linear dependency on m in your final solution. You can safely assume that $m = n^{O(1)}$.