Princeton Univ Fall '23	COS 521: Advanced Algorithms
Homework 4	
Out: *Oct 31*	Due: *Nov 19*

Instructions:

- Upload your non-extra solutions to Gradescope in a single PDF file, and mark your solution to each problem. Please make sure you are uploading the correct PDF! Please anonymize your submission (i.e., do not list your name in the PDF), but if you forget, it's OK.
- If you choose to do extra credit, upload your solution to the extra credits as a single separate PDF file to Gradescope. Please again anonymize your submission.
- You may collaborate with any classmates, textbooks, the Internet, etc. Please upload a brief "collaboration statement" listing any collaborators as a separate PDF on Gradescope (if you forget, it's OK). But always **write up your solutions individually**.
- For each problem, you should have a solid writeup that clearly states key, concrete lemmas towards your full solution (and then you should prove those lemmas). A reader should be able to read any definitions, plus your lemma statements, and quickly conclude from these that your outline is correct. This is the most important part of your writeup, and the precise statements of your lemmas should tie together in a correct logical chain.
- A reader should also be able to verify the proof of each lemma statement in your outline, although it is OK to skip proofs that are clear without justification (and it is OK to skip tedious calculations). Expect to learn throughout the semester what typically counts as 'clear'.
- You can use the style of Lecture Notes and Staff Solutions as a guide. These tend to break down proofs into roughly the same style of concrete lemmas you are expected to do on homeworks. However, they also tend to prove each lemma in slightly more detail than is necessary on PSets (for example, they give proofs of some small claims/observations that would be OK to state without proof on a PSet).
- Each problem is worth twenty points (even those with multiple subparts), unless explicitly stated otherwise.

Problems:

§1 Given a data matrix $X \in \mathbb{R}^{n \times d}$ with n rows (data points) $x_1, \ldots, x_n \in \mathbb{R}^d$, the *k-means clustering problem* asks us to find a partition of our points into k disjoint sets (clusters) $\mathcal{C}_1, \ldots \mathcal{C}_k \subseteq \{1, \ldots, n\}$ with $\bigcup_{j=1}^{k} \mathcal{C}_j = \{1, \ldots, n\}$.

Let $c_j = \frac{1}{|\mathcal{C}_j|} \sum_{i \in \mathcal{C}_j} x_i$ be the centroid of cluster j. We want to choose our clusters to minimize the sum of squared distances from every point to its cluster centroid. I.e. we want to choose $\mathcal{C}_1, \ldots \mathcal{C}_k$ to minimize:

$$f_X(\mathcal{C}_1, \ldots \mathcal{C}_k) = \sum_{j=1}^{k} \sum_{i \in \mathcal{C}_j} \|c_j - x_i\|_2^2.$$

There are a number of algorithms for solving the k-means clustering problem. They typically run more slowly for higher dimensional data points, i.e. when d is larger. In this problem we consider what sort of approximation we can achieve if we instead solve the problem using dimensionality reduced vectors in place of $x_1, \ldots, x_n$.

Let $OPT_X = \min_{\mathcal{C}_1, \ldots \mathcal{C}_k} f_X(\mathcal{C}_1, \ldots \mathcal{C}_k)$.

Suppose that Π is a Johnson-Lindenstrauss map into $s = O(\log n/\epsilon^2)$ dimensions and that we select the optimal set of clusters for $\Pi x_1, \ldots, \Pi x_n$. Call these clusters them $\tilde{\mathcal{C}}_1, \ldots \tilde{\mathcal{C}}_k$. Show that they obtain objective value $f_X(\tilde{\mathcal{C}}_1, \ldots \tilde{\mathcal{C}}_k) \leq (1+\epsilon) OPT_X$, with high probability.

Hint: reformulate the objective function to only involve ℓ_2 distances between data points.

§2 (a) Let $m \geq 1$ be an integer, prove that there can be at most $2^{O(m)}$ points in $\mathbb{R}^m$ such that the distance between *every* pair of points is between 1 and 3.

(b) For any $n \geq 1$, construct a set of n points in $\mathbb{R}^n$ such that the distance between *every* pair of points is equal to 2.

(c) Prove that the new dimension in the JL lemma is optimal up to a constant factor when $\varepsilon = 0.1$, i.e., in general, we cannot hope to map an arbitrary set of n points in a high dimensional space to $\mathbb{R}^{c \log n}$ while the pairwise distances are preserved up to a 1 ± 0.1 factor, when $c > 0$ is a sufficiently small constant.

§3 Consider the following variant of online set cover. Offline, we are given a universe $U := \{1, \ldots, n\}$ of n elements and a family $\mathcal{S} := \{S_1, \ldots, S_m\}$ of m sets where $\bigcup_i S_i = U$. The algorithm starts with $A = \emptyset$ which denotes the collection of selected sets.

In each time step $t \in \{1, \ldots, T\}$, an adversary reveals an element $e_t \in U$, and the online algorithm has to immediately ensure that $e_t \in \bigcup_{S \in A} S$, i.e., if e_t is already covered then the algorithm doesn't need to select a new set, and otherwise the algorithm has to select a set into A that contains e_t. The goal of the algorithm is to minimize the size of A.

To be clear: it may be that not all elements of U are eventually revealed.

Show that every deterministic algorithm achieves a competitive ratio of at best $\Omega(\log(mn))$.

Hint: The staff solution considers instances where m and n are polynomially-related, so that $\log n = \Theta(\log m)$. The staff solution also considers a class of sequences where the optimal offline solution is always one, but for any deterministic algorithm, there is always a sequence that requires $\Omega(\log(mn))$ sets.

Remark: The $\Omega(\log(mn))$ bound also holds against randomized online algorithms, but you do not have to prove this.

§4 (Approximate LP Solving via Multiplicative Weights) This exercise develops an algorithm to approximately solve Linear Programs.

Consider the problem of finding if a system of linear inequalities as below admits a solution - i.e., whether the system is feasible. This is an example of a feasibility linear program and while it appears restrictive, one can use it solve arbitrary linear programs to obtain approximate solutions. **For all subparts, you may assume that $|a_{ij}| \leq 1$ and $|b_i| \leq 1$ for all i, j.**

$$\begin{aligned} a_1^\top x &\geq b_1 \\ a_2^\top x &\geq b_2 \\ &\vdots \\ a_m^\top x &\geq b_m \\ x_i &\geq 0 \ \forall \ i \in [n] \\ \sum_{i=1}^{n} x_i &= 1. \end{aligned} \tag{1}$$

(a) Design a simple algorithm to solve the following linear program, which has only two non-trivial constraints. Below, the weights $w_1, w_2, \ldots, w_m$ are fixed (along with the vectors $a_j^\top$ and numbers b_j), and $x_1, \ldots, x_n$ are the variables.

$$\begin{aligned} \max \sum_{j=1}^{m} w_j (a_j^\top x - b_j) & \\ x_i \geq 0 \ \forall i \in [n] & \\ \sum_{i=1}^{n} x_i = 1. & \end{aligned} \tag{2}$$

(b) Prove that if there exist non-negative weights $w_1, w_2, \ldots, w_m$ such that the value of the program above is negative, then the system (1) is infeasible.

(c) The above setting of finding weights that certify infeasibility of (1) might remind you of the setting of weighting the experts via multiplicative weights update rule discussed in the class. Use these ideas to obtain an algorithm that a) either finds a set of non-negative weights certifying infeasibility of LP in (1) or b) finds a

solution x that approximately satisfies all the constraints in (1), i.e., for each $1 \le j \le m$, $a_j^\top x - b_j \ge -\epsilon$, and for each $1 \le i \le n$, $x_i \ge 0$, and $\sum_{i=1}^n x_i = 1$. Prove that your algorithm terminates after solving $O(\ln(m)/\varepsilon^2)$ LPs of form (2) (you do not need to analyze the remaining runtime).

(Hint: Identify m "experts" - one for each inequality constraint in (1) and maintain a weighting of experts (starting with the uniform weighting of all 1s, say) for times $t = 0, 1, \ldots,$ - these are your progressively improving guesses for the weights. Solve (2) using the weights at time t. If the value of (2) is negative, you are done, otherwise think of the "cost" of the j^{th} expert as $a_j^\top x^{(t)} - b_j$ where $x^{(t)}$ is the solution to the LP (2) at time t and update the weights.)

§5 The standard Cauchy distribution has the following probability density function

$$p(x) = \frac{1}{\pi(1+x^2)},$$

for $x \in \mathbb{R}$. It has the following property: Let X_1, X_2, X be independent standard Cauchy random variables, then for any $a, b \in \mathbb{R}$, $aX_1 + bX_2$ has the same distribution as $(|a|+|b|)X$ (you don't need to prove it!). That is, any weighted sum of independent standard Cauchy random variables is still a Cauchy random variable, scaled by the sum of absolute values of the weights.

Design an LSH for ℓ_1 distance: Fix $r_1 < r_2$, design a random function $f : \mathbb{R}^n \to \mathbb{Z}$ such that there exists $p_1 > p_2$,

- for any $\|x - y\|_1 \le r_1$, we have $\Pr_f[f(x) = f(y)] > p_1$, and
- for any $\|x - y\|_1 \ge r_2$, we have $\Pr_f[f(x) = f(y)] < p_2$,

where $\|x - y\|_1 = \sum_{i=1}^n |x_i - y_i|$.

Note: You don't need to specify the values of p_1, p_2 given r_1, r_2, it suffices to show that they exist.

Hint: Consider the inner product of x and a vector with random Cauchy coordinates.

Extra Credit:

§1 (Extra credit, follows "Given a data matrix...") Instead, suppose we reduce our points to k dimensions using the SVD. I.e. let $V_k \in \mathbb{R}^{d \times k}$ have the first k right singular vectors of X. Show that, if $\tilde{\mathcal{C}}_1, \ldots \tilde{\mathcal{C}}_k$ are the optimal clusters for $V_k^T x_1, \ldots, V_k^T x_n$, then $f_X(\tilde{\mathcal{C}}_1, \ldots \tilde{\mathcal{C}}_k) \le 2OPT_X$.

Hint: show that for every set of clusters, there is an orthonormal matrix $C \in \mathbb{R}^{n \times k}$ such that $f_X(\mathcal{C}_1, \ldots \mathcal{C}_k) = \|X - CC^T X\|_F^2$. I.e. reformulate k-means as a k-rank approximation problem.