Problem Set 5 CS265, Autumn 2023
Due: Friday 11/3 at 11:59 pm on Gradescope
Group members: INSERT NAMES HERE

Please follow the homework policies on the course website.

---

1. **(14 pt.) [Another way to sketch sparse vectors.]** Suppose that A is an list of length n, containing elements from a large universe $\mathcal{U}$. Our goal is to estimate the frequencies of each element in $\mathcal{U}$: that is, for $x \in \mathcal{U}$, how often does x appear in A?

   The catch is that A is too big to look at all at once. Instead, we see the elements of A one at a time: $A[0], A[1], A[2], \ldots$. Unfortunately, $\mathcal{U}$ is also really big, so we can't just keep a count of how often we see each element.

   In this problem, we'll see a construction of a randomized data structure that will keep a "sketch" of the list A, use small space, and will be able to efficiently answer queries of the form "approximately how often did x occur in A"?

   Specifically, our goal is the following: we would like a (small-space) data structure, which supports operations `update`(x) and `count`(x). The `update` function inserts an item $x \in \mathcal{U}$ into the data structure. The `count` function should have the following guarantee, for some $\delta, \epsilon > 0$. After calling `update` n times, `count`(x) should satisfy

   $$C_x \leq \texttt{count}(x) \leq C_x + \epsilon n \tag{1}$$

   with probability at least $1 - \delta$, where C_x is the true count of x in A.

   (a) **(3 pt.)** Your friend suggests the following strategy (this will not be our final strategy). We start with an array R of length b initialized to 0, and a random hash function $h : \mathcal{U} \to \{0, 1, \ldots, b-1\}$. You can assume that h is drawn from some universal hash family, i.e $P(h(x) = h(y)) = 1/b$ for any $x \neq y$. Then the operations are:

   - `update`(x): Increment $R[h(x)]$ by 1.
   - `count`(x): return $R[h(x)]$.

   For every entry $A[i]$ in the list it encounters, the scheme calls `update`$(A[i])$.

   After sequentially processing all n items in the list, what is the expected value of `count`(x)?

   (b) **(2 pt.)** Show that there is a choice of b that is $O(1/\epsilon)$ so that, for any fixed $x \in \mathcal{U}$, we have

   $$\Pr[\texttt{count}(x) < C_x] = 0$$

   and

   $$\Pr[\texttt{count}(x) \geq C_x + \epsilon n] \leq \frac{1}{e}.$$

   [**HINT:** *The first of the requirements is true no matter what b is.*]

   (c) **(2 pt.)** Explain how you would use T copies of the construction in part (a) to define a data structure that, for any fixed $x \in \mathcal{U}$, satisfies (1) with high probability. How big

do you need to take T so that the (1) is satisfied with probability at least $1 - \delta$? How much space does your modified construction use? (It should be sublinear in $|\mathcal{U}|$ and n). Give a complete description and analysis of the data structure, and explain how much space it uses. You may assume that it takes $O(\log |\mathcal{U}|)$ bits to store the hash function h and $O(\log n)$ to store each element in the array R.

(d) Explain how to use your algorithm to solve the following problem:

i. **(4 pt.)** Given a k-sparse vector $a \in \mathbb{Z}_{\geq 0}^N$ ($\mathbb{Z}_{\geq 0}$ is the set of non-negative integers), design a randomized matrix $\Phi \in \mathbb{R}^{m \times N}$ for $m = O(\frac{k \log N}{\epsilon})$ so that the following happens. With probability at least 0.99 over the choice of Φ, you can recover $\tilde{a}$ given Φa, so that simultaneously for all $i \in 1, ..., N$, we have

$$|\tilde{a}[i] - a[i]| \leq \frac{\epsilon \|a\|_1}{2k}.$$

[**HINT:** *Think of the k-sparse vector a as being the histogram of the items in the list A from the previous parts.*]
[**HINT:** *How can you represent a hash function as a matrix multiplication?*]
[**HINT:** *Note that we want a tighter bound, and we want the bound to hold simultaneously for all i. How can we change b and T to achieve this?*]

ii. **(3 pt.)** Now, assuming the above holds for all i, use the k-sparseness of a to construct $\hat{a}$ from $\tilde{a}$ such that

$$\|\hat{a} - a\|_1 \leq \epsilon \|a\|_1.$$

iii. **(0 pt.) [This question is zero points, but worth thinking about.]** How does the guarantee in the previous part compare to the RIP matrices (and the compressed sensing guarantee that we can get from them, Theorem 1 in the Lecture 9 lecture notes) that we saw in class? (i.e., is this guarantee weaker? Stronger? Incomparable? The same?)

SOLUTION:

2. **(6 pt.)[Bayesians and Frequentists.]**

Suppose that there are n statisticians, with each one being either a "Bayesian" or a "Frequentist". We have a bunch of math problems to solve, and suppose that each statistician has a list of at least $1 + \log_2(n)$ problems that they would be happy to work on. The statisticians' lists may be different and may overlap—it could even be the case that everyone has the same list.

Unfortunately, "Bayesians" and "Frequentists" sometimes have trouble collaborating.

Use the probabilistic method to show that it is possible to assign the math problems to the statisticians such that 1) each statistician is assigned a problem to work on, and 2) for each problem, either it is unassigned, it is assigned to only Bayesians, or it is assigned to only Frequentists [ie we won't need to worry about collaboration squabbles]. Note that it is fine

to assign lots of people to a single problem, as long as all the people assigned are the same type, and it is fine for some problems to be unassigned.

[**HINT:** *To use the probabilistic method, you just have to define a probability distribution over assignments such that with* positive *probability, the assignment satisfies the criteria. A first attempt might be to assign each statistician to a random problem from their list of problems—this distribution does* not *seem to work...you'll have to come up with a more interesting distribution.*]

SOLUTION:

3. **(12 pt.)[Who needs geometry when you have the probabilistic method?]**

Suppose that your friend has – perhaps adversarially – drawn k points on the surface of a flat table, which you can think of as an infinite plane. Your goal will be to cover all the points with pennies, *without any pennies overlapping each other*. How big can k such that it is *always* possible for you to cover these k points with non-overlapping pennies? [Think of pennies as circles of radius 1.]

(a) **(1 pt.)** Prove that there is a way of placing non-overlapping pennies on the plane such that the pennies cover more than 90% of the area of the plane. [Hint: the optimal configuration is the hexagonal tiling used to arrange oranges in the supermarket. The precise area covered in this optimal configuration will be the following fraction: $\pi/(2\sqrt{3}) \approx 0.907$.]

(b) **(4 pt.)** Prove that there is some configuration of $k = 1000$ points such that it is impossible to cover them all with non-overlapping pennies.

(c) **(0 pt.)** What is the smallest value of k for which you can prove that there is a configuration that is impossible to cover?

(d) **(6 pt.)** Prove that for any configuration of 10 points, its possible to cover them via at most 10 non-overlapping pennies. [Hint: Use part (a) and the probabilistic method!]

(e) **(1 pt.)** Would your probabilistic proof work for a set of 11 points? Why or why not?

(f) **(0 pt.)** Can you come up with a non-probabilistic method proof for the fact that any 10 points can be covered with non-overlapping pennies?

SOLUTION: