1. (20 points) **(Simpler JL Dimension Reduction?)** In class, we saw how to reduce the dimension of n points from D to $\ell = O(\frac{\log n}{\epsilon^2})$ using projections into ℓ random directions, and preserve all pairwise distances to within $(1 \pm \epsilon)$ factor. One of the suggestions in class was whether we can simply reduce dimension by sampling a set S of ℓ random coordinates (say with repetition), and representing each point only using the coordinates of S (possibly scaling each coordinate). Will this work?

   (a) (5 points) Give a simple example of n vectors in n dimensional space, and sampling $S = \ell$ coordinates will not preserve all pairwise distances to within a factor of $(1 \pm \epsilon)$. That is, if $x_1, x_2, \ldots, x_n$ are the n vectors in n dimensional space, and $\widehat{x}_1, \widehat{x}_2, \ldots, \widehat{x}_n$ are the projections in ℓ dimensional space by only using the coordinates in S, then w.h.p there exists no universal scaling C such that $(1-\epsilon)||x_i - x_j||_2 \leq ||C\widehat{x}_i - C\widehat{x}_j||_2 \leq (1+\epsilon)||x_i - x_j||_2$. (Hint: think of sparse vectors.)

   Solution:

   Proof. $1 + 1 = 2$. □

   (b) (10 points) Now suppose we additionally had the promise that every coordinate in each vector is small, i.e., at most, say $C/\sqrt{n}$, for some constant C (for simplicity, assume that all vectors are unit vectors). Then show that projecting down to random subset of ℓ coordinates (for a suitable ℓ) actually works to achieve a weak form of dimensionality reduction, i.e., all n vector lengths are approximately preserved up to scaling.

   Solution:

   Proof. $1 + 1 = 2$. □

   (c) (5 points) Explain in words why the projection onto a random k dimensional subspace (as we did in class) intuitively works. (Hint: try viewing this in two steps, a random rotation of all points followed by sampling coordinates at random.) Hopefully, you would convince yourself that this is another illustration of the power of randomization, where it acts as insurance against some bad events conspiring.

   Solution:

   Proof. $1 + 1 = 2$. □

2. (10 points) **(Mysteries of High-Dimensional Geometry)** It is a basic fact in linear algebra that there can be at most n orthogonal vectors in $\mathbb{R}^n$. We now explore how many vectors can be mutually *almost orthogonal* to each other. That is, we will try to define a exponentially large (in n) collection of vectors such that for any pair u and v, $|\langle u, v \rangle| \leq \epsilon$.

(a) (5 points) Pick two vectors uniformly at random in $\{-1, 1\}^n$ (i.e., each coordinate of each vector is independently 1 or -1 with probability $1/2$). Use your favorite concentration bound to show that $\Pr[|\langle x, y\rangle| > \epsilon n] \leq O(\exp(-\epsilon^2 n/10))$

Solution:

Proof. $1 + 1 = 2$. □

(b) (5 points) Sample a set S of vectors from $\{-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\}^n$ independently and uniformly at random. How large a set S can you tolerate while ensuring that every pair of vectors from S is ϵ-orthogonal (i.e., absolute value of inner product is at most ϵ)?

Solution:

Proof. $1 + 1 = 2$. □

3. (20 points) **(LPs: How good are they?)** Often times, we use a Linear Programming approach to find good approximation algorithms for NP-hard optimization problems. For example, the current best known approximation algorithms for problems in scheduling, routing, resource allocation, etc. are all via LP relaxations. However, we should be careful in directly using LPs, as sometimes they suffer from what are called integrality gaps, where the LP optimal solution can be highly fractional, and can't be used to recover good integral solutions.

(a) (5 points) Consider the vertex cover problem: given a graph $G = (V, E)$ where each vertex has a non-negative cost c_v, we want to choose the cheapest set S of vertices such that every edge is incident on at least one vertex in S. Write this as an Integer Programming formulation, and also mention the changes to obtain the corresponding LP relaxation.

Solution:

Proof. $1 + 1 = 2$. □

(b) (5 points) Design a simple deterministic 2-approximation algorithm using an optimal LP solution. That is, find a solution S such that $w(S) \leq 2\mathsf{Cost}(\text{Opt LP Soln})$.

Solution:

Proof. $1 + 1 = 2$. □

(c) (5 points) Now consider the following problem: given a collection of items $1, 2, \ldots, n$ where item i has utility $u_i \geq 0$ and cost $c_i \geq 0$. There is also a utility requirement of $U \geq 0$. The goal is to choose the cheapest subset of items such that their total

utility exceeds the requirement U. Write an Integer Programming formulation, and the corresponding LP relaxation.

Solution:

Proof. $1 + 1 = 2$. □

(d) (5 points) Can there be a good rounding algorithm like Vertex Cover? In other words, given any instance, is it always possible to round a feasible LP solution to an integer solution S such that $c(S) \leq O(1)\text{cost(LP OPT)}$? If not, give an example saying why.

Solution:

Proof. $1 + 1 = 2$. □