New York University Tandon School of Engineering
Computer Science and Engineering

CS-GY 6763: Homework 4.
Due Wed., December 15th, 2021, 11:59pm.

Collaboration is allowed on this problem set, but solutions must be written-up individually. Please list collaborators for each problem separately, or write "No Collaborators" if you worked alone.

Problem 1: Optimal Low-Rank Approximation

(10 pts) In class we claimed that the best low-rank approximation to any matrix $X \in \mathbb{R}^{n\times d}$ is given by $XV_kV_k^T$, where $V_k \in \mathbb{R}^{d\times k}$ contains the top k right singular vectors of $X = U\Sigma V^T$ – i.e., the top k eigenvectors of the positive semidefinite matrix X^TX. Here you will prove this from scratch, using just basic properties of projection matrices and eigenvectors.

1. Let $X \in \mathbb{R}^{n\times d}$ be as above, and let $M \in \mathbb{R}^{n\times d}$ be a candidate k rank approximation that has singular value decomposition $M = QDZ^T$ for orthonormal $Q \in \mathbb{R}^{n\times k}$, $Z \in \mathbb{R}^{d\times k}$, and diagonal $D \in \mathbb{R}^{k\times k}$. Prove that $\|X-M\|_F^2 = \|XZZ^T - M\|_F^2 + \|X - XZZ^T\|_F^2$ and conclude that, if $M = \arg\min_{\text{rank } kB} \|X-B\|_F^2$, then $M = XZZ^T$.
2. Using a similar argument as above, one can show that, if $M = \arg\min_{\text{rank } kB} \|X-B\|_F^2$, then $M = QQ^TX$. Use this and part (1) to prove that $X^TXZ = ZD^2$ for any optimal rank k approximation $M = QDZ^T$. Conclude that each column of Z is an eigenvector of X^TX. **Hint: It may be helpful to prove as an intermediate step that $XZ = QD$ and $Q^TX = DZ^T$.**
3. Complete the proof, showing that the best low-rank approximation of X is given by $XV_kV_k^T$ where V_k contains the top k eigenvectors of X^TX.

Problem 2: Matrix Concentration from Scalar Concentration

(15 pts) This problem asks you to prove a simplified (and slightly weaker) version of the matrix concentration result used in Lecture 10. Construct a random *symmetric* matrix $R \in \mathbb{R}^{n\times n}$ by setting $R_{ij} = R_{ji}$ to $+1$ or -1, uniformly at random. Prove that, with high probability,

$$\|R\|_2 \leq c\sqrt{n\log n},$$

for some constant c. This is much better than the naive bound of $\|R\|_2 \leq \|R\|_F = n$ and it's nearly tight: we always have that $\|R\|_2^2 \geq \|R\|_F^2/n$ (do you see why?) so $\|R\|_2 \geq \sqrt{n}$ no matter what.

Here are a few hints that might help you along:

- Recall that for a matrix R, $\|R\|_2 = \max_{x\in\mathbb{R}^n} \frac{\|Rx\|_2}{\|x\|_2}$. When R is symmetric, it also holds that $\|R\|_2 = \max_{x\in\mathbb{R}^n} \frac{|x^TRx|}{x^Tx}$.
- Try to first bound $\frac{|x^TRx|}{x^Tx}$ for one particular x. You might want to use a Hoeffding bound.
- Then try to extend the result to hold for all x simultaneously, using an ϵ-net argument.

Problem 3: Spectral Methods for Cliques

(10 pts) A common tasks in data mining is to identify large *cliques* in a graph. For example, in social networks, large cliques can be indicators of fraudulent accounts or networks of accounts designed to promote certain content. In this problem, we consider a spectral heuristic for finding a large clique based on the top eigenvector of the graph adjacency matrix A:

- Compute the leading eigenvector v_1 of A.
- Let $i_1, \ldots, i_k \in \{1, \ldots, n\}$ be the indices of the k entries in v_1 with largest absolute value.

- Check if nodes $i_1, \ldots, i_k$ form a k-clique.

We will analyze this heuristic on a natural random graph model. Specifically, let G be an Erdos-Renyi random graph: we start with n nodes, and for every pair of nodes (i, j), we add an edge between the pair with probability $p < 1$. To simplify the math, also assume that we add a self-loop at every vertex i with probability p. Then, choose a fixed subset S of k nodes to form a clique. Connect all nodes in S with edges and add self-loops. We will argue that, for sufficiently large k, we can expect the heuristic above to identify the nodes in the clique.

1. Let A be the adjacency matrix of a random graph generated as above. What is $\mathbb{E}[A]$? Prove that the rank of $\mathbb{E}[A]$ is 2.

2. Derive expressions for the two non-zero eigenvalues of $\mathbb{E}[A]$, and their corresponding eigenvectors. **Hint:** First argue that, up to multiplying by a constant, any eigenvector v must have $v[i] = 1$ for all $i \notin S$ and $v[i] = \alpha$ for all $i \in S$, where α is a constant. Then use some high school algebra 2!

3. Using your results from (2) above, argue that, up to a positive scaling, the top eigenvector v_1 has $v[i] = 1$ for all $i \notin S$ and $v[i] = \alpha$ for all $i \in S$, where $\alpha > 1$. In other words, the largest entries of v_1 exactly correspond to the nodes in the clique!

4. To prove the algorithm works, it is possible to use a matrix concentration inequality to argue that the top eigenvector of A is close to that of $E[A]$. Instead of doing that, let's verify things experimentally. Generate a graph G according to the prescribed model with $n = 900$, $k = |S| = 30$, and $p = .1$. Compute the top eigenvector of A and look at its 30 largest entries in magnitude. What fraction of nodes in the clique S are among these 30 entries? Repeat the experiment and report the average fraction recovered.

Problem 4: 18th Century Style Compressed Sensing

(10 pts) In Lecture 12 it was mentioned that there exist simpler compressed sensing schemes that work when noise/numerical precision is not an issue. Let $q_1, \ldots, q_n \in \mathbb{R}$ be any set of *distinct* numbers. E.g. we could choose $[q_1, \ldots, q_n] = [1, \ldots, n]$. Consider the sensing matrix $A \in \mathbb{R}^{2k \times n}$:

$$A = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ q_1 & q_2 & q_3 & \ldots & q_n \\ (q_1)^2 & (q_2)^2 & (q_3)^2 & \ldots & (q_n)^2 \\ \vdots & \vdots & \vdots & & \vdots \\ (q_1)^{2k-1} & (q_2)^{2k-1} & (q_3)^{2k-1} & \ldots & (q_n)^{2k-1} \end{bmatrix}$$

This A does not obey any sort of RIP property. Nevertheless, show that, if $x \in \mathbb{R}^n$ is a k sparse vector – i.e. $\|x\|_0 \leq k$ – then we can recover x from Ax. You don't need to give an efficient algorithm. Just argue that for any given $y \in \mathbb{R}^{2k}$, there is at most one k-sparse x such that $y = Ax$. (Hint: Use that a non-zero degree p polynomial cannot have more than p roots. You may also want to use that the column and row rank of a matrix are always equal.)

Bonus 1: Sparse Recovery for Dense Vectors

(5 pts extra credit) A compressed sensing scheme typically recovers x from a linear sketch Ax whenever x is k-sparse. When x is not k-sparse, there is no guarantee about what is returned. E.g., for the measurement matrix A described above, for any specified k, there exists an algorithm $Decode(y)$ which returns x if $y = Ax$ for a k-sparse x. If $y \neq Ax$ for some k-sparse x, $Decode(y)$ can return anything. In this problem we consider a method that will still return *something useful* when x is not k-sparse.

In particular, your goal is to design a measurement matrix $B \in \mathbb{R}^{c \log n \times n}$, where c is a constant, such that *for any* x (i.e. not necessarily sparse) it is possible to recover a single index/value pair (i, x_i) with

$x_i \neq 0$ from Bx with constant probability (e.g. with success probability 9/10). Your algorithm can return *any* (i, x_i) as long as $x_i \neq 0$. **Hint: One possible B takes the form:**

$$B = \begin{bmatrix} AD_0 \\ AD_1 \\ AD_2 \\ \dots \\ AD_s \end{bmatrix}$$

where $D_1, \dots, D_s$ are carefully (and randomly) constructed diagonal matrices and A is the matrix from Problem 3 with $k = O(1)$.

Bonus 2: Communicating in the Dark is Easier with Shared Random Coins

(5 pts extra credit) Suppose Jesse holds a subset of elements $J \subseteq \{1, \dots, n\}$. Leslie holds another subset $L \subseteq \{1, \dots, n\}$. Jesse and Leslie do not know what elements the other holds. Using as little communication as possible, Jesse wants to figure out if she or Leslie hold any unique elements – i.e. if there is any $j \in J \cup L - J \cap L$.

Show that, for some constant c, Leslie can send Jesse a single message of $O(\log^c n)$ bits that allows her to find such a j if one exists, with constant success probability.

You can assume that Jesse and Leslie decide on a strategy in advance, and that they have access to an unlimited source of shared random bits (e.g. that are published by some third party). You might want to use the result from Problem 4.

This result should surprise you! Even if Leslie *knew* all of Jesse's elements, $O(\log n)$ bits would be needed to communicate if they hold any unique elements. Here we are saying that nearly the same communication complexity can be achieved with *no prior knowledge* of J.