New York University Tandon School of Engineering
Computer Science and Engineering

CS-GY 9223I: Homework 3.
Due Monday, December 2nd, 2019, 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: Clustering vs. Low-rank Approximation

(10 pts) Suppose we're given a data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with $d \leq n$ and singular values $\sigma_1, \ldots, \sigma_d$. Denote $\mathbf{X}$'s rows by $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$.

1. **(3 pts)** Prove the following statement from class: For any $k \leq d$,

$$\min_{\mathbf{B} \text{ with } \operatorname{rank}(\mathbf{B})=k} \|\mathbf{X} - \mathbf{B}\|_F^2 = \sum_{i=k+1}^{d} \sigma_i^2.$$

   You can use the fact that the SVD can be used to give the optimal rank k approximation.

Recall the k-means objective: partition a data set $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ into clusters $\mathcal{C}_1, \ldots, \mathcal{C}_k$ to minimize:

$$\operatorname{Cost}(\mathcal{C}_1, \ldots, \mathcal{C}_k) = \sum_{i=1}^{k} \sum_{x_j \in \mathcal{C}_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|_2^2$$

where $\mu_i = \frac{1}{|\mathcal{C}_i|} \sum_{x \in \mathcal{C}_i} \mathbf{x}$ is the centroid of the i^{th} cluster.

2. **(4 pts)** Prove that $\min \operatorname{Cost}(\mathcal{C}_1, \ldots, \mathcal{C}_k) \geq \sum_{i=k+1}^{d} \sigma_i^2$. In other words, the optimal k-means objective cost is always larger than the optimal low-rank approximation bost.

3. **(3 pts)** Describe data sets where:

   (a) $\min \operatorname{Cost}(\mathcal{C}_1, \ldots, \mathcal{C}_k) \neq 0$ and $\min \operatorname{Cost}(\mathcal{C}_1, \ldots, \mathcal{C}_k) = \sum_{i=k+1}^{d} \sigma_i^2$, and where

   (b) $\min \operatorname{Cost}(\mathcal{C}_1, \ldots, \mathcal{C}_k) \gg \sum_{i=k+1}^{d} \sigma_i^2$ (e.g., where the k-means cost 10x greater).

Problem 2: Applications of the SVD

1. **(5 pts)** Consider $\mathbf{A} \in \mathbb{R}^{n \times d}$ with singular value decomposition $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$. For $\mathbf{b} \in \mathbb{R}^n$, let $\mathbf{x}^* = \mathbf{V}\boldsymbol{\Sigma}^{-1}\mathbf{U}^T\mathbf{b}$. Prove that:

$$\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2^2 = \min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2.$$

2. **(10 pts)** Suppose you are given all pairs distances between a set of points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$. You can assume that $d \ll n$. Formally, you are given an $n \times n$ matrix $\mathbf{D}$ with $\mathbf{D}_{i,j} = \|\mathbf{x}_i - \mathbf{x}_j\|_2^2$. You would like to recover the location of the original points, at least up to possible rotation and translation (which do not change pairwise distances).

   Since we can only recover up to a translation, it may be easiest to assume that the points are centered around the origin. I.e. that $\sum_{i=1}^{n} \mathbf{x}_i = \mathbf{0}$.

   (a) Under this assumption, describe a polynomial time algorithm for learning $\sum_{i=1}^{n} \|\mathbf{x}_i\|_2^2$ from $\mathbf{D}$. Hint: expand $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2$ as $(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)$ and go from there.

   (b) Next, describe a polynomial time algorithm for learning $\|\mathbf{x}_i\|_2^2$ for each $i \in 1, \ldots, n$.

   (c) Finally, describe an algorithm for recovering a set of points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ which realize the distances in $\mathbf{D}$. Hint: This is where you will use the SVD! It might help to know (and prove to yourself) that $\mathbf{D}$ has rank $\leq d + 2$.

(d) **(Bonus: 5pts)** Implement your algorithm and run it on the U.S. cities dataset provided in `UScities.txt`. Note that the distances in the file are unsquared Euclidean distances, so you need to square them to obtain $\mathbf{D}$. Plot your estimated city locations on a 2D plot and label the cities to make it clear how the plot is oriented. Submit these images and your code with the problem set (in the same file, as plaintext) – I don't need to be able to run the code.

Problem 3: Sketches for Cut Estimation

(10 pts) Let $G(V, E)$ be a graph with vertex set V and edge set E. Suppose $|V| = n$ and G has edge vertex incidence matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$. Let $\mathbf{\Pi} \in \mathbb{R}^{k \times m}$ be a random Johnson-Lindenstrauss matrix with k rows. Suppose we sketch G by forming the $O(kn)$ matrix $\mathbf{\Pi B}$.

Given any vertex set $S \subseteq V$, describe an algorithm that estimates $\text{cut}(S, V \setminus S)$ up to $(1 \pm \epsilon)$ error with probability $1 - \delta$ using only the information in $\mathbf{\Pi B}$, as long as $k = O(\log(1/\delta)/\epsilon^2)$.

Problem 4: 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 11. 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 $\|R\|_2 \leq \|R\|_F = n$.

Here are a few hints that might help you along:

- Recall that for a matrix R, $\|R\|_2 = \max_{x \in \mathbb{R}^n} \frac{\|Mx\|_2}{\|x\|_2}$. When R is symmetric, it also holds that $\|R\|_2 = \max_{x \in \mathbb{R}^n} \frac{|x^T R x|}{x^T x}$.
- Try to first bound $\frac{|x^T R x|}{x^T x}$ 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.