CSE 521: Design and Analysis of Algorithms **Winter 2017**

Problem Set 4

Deadline: Feb 26th (at 12:00PM) in Canvas

1) Prove the following Matrix equations:

   a) Let $A \in \mathbb{R}^{n\times n}$ and let $U \in \mathbb{R}^{k\times n}$ be a matrix with orthonormal columns $U_1, \ldots, U_k$. So, $UU^T = \sum_{i=1}^{k} U_i U_i^T$ is a projection matrix. Show that

   $$\|A - UU^T A\|_F^2 = \|A\|_F^2 - \|U^T A\|_F^2.$$

   b) We say a matrix $U \in \mathbb{R}^{n\times n}$ is a unitary matrix if all singular values of U are 1. Show that for any unitary matrix U,

   $$UU^T = I.$$

   Use this to show that unitary matrices are rotation matrices, i.e., for any vector v,

   $$\|Uv\| = \|v\|.$$

2) In this problem we discuss a fast algorithm for approximately estimating the low rank approximation with respect to the Frobenius norm.

   a) Let $A \in \mathbb{R}^{m\times n}$ and suppose we want to estimate Av for a vector $v \in \mathbb{R}^n$. Here is a randomized algorithm for this task. Choose the i-th column of A, A_i, with probability

   $$p_i = \frac{\|A_i\|^2}{\|A\|_F^2}$$

   and let $X = A_i v_i / p_i$. Show that $\mathbb{E}[X] = Av$. Calculate $\mathrm{Var}(X) = \mathbb{E}\left[\|X\|^2\right] - \|\mathbb{E}X\|^2$.

   b) Next, we use a similar idea to approximate A. For $1 \le i \le s$ let $X_i = \frac{A_j}{\sqrt{s p_j}}$ with probability p_j where $1 \le j \le n$. Let $X \in \mathbb{R}^{m\times s}$ and let X_i be the i-th columns of X. Note that $XX^T = \sum_{i=1}^{s} X_i X_i^T$. Show that

   $$\mathbb{E}XX^T = AA^T.$$

   Show that $\mathbb{E}\|XX^T - AA^T\|_F^2 \le \frac{1}{s}\|A\|_F^4$.

   c) **Extra Credit:** Let $X = \sum_{i=1}^{s} \sigma_i u_i v_i^T$ be the SVD decomposition of X where $\sigma_1 \ge \cdots \ge \sigma_s$. Let U_k be the matrix with columns $u_1, \ldots, u_k$. So, $U_k U_k^T = \sum_{i=1}^{k} u_i u_i^T$ is a projection matrix. We want to show that for any such matrix X and U_k,

   $$\|A - U_k U_k^T A\|_F^2 \le \|A - A_k\|_F^2 + 2\sqrt{k}\|AA^T - XX^T\|_F, \tag{4.1}$$

   where A_k is the best rank k approximation of A. Note that if this is true we can simply let $s = O(k/\epsilon^2)$ and then a random X chosen from part (b) would give

   $$\|A - U_k U_k^T A\|_F^2 \le \|A - A_k\|_F^2 + \epsilon\|A\|_F^2.$$

   Also, note that the algorithm runs in time $\mathrm{nnz}(A) + O(mk^2/\epsilon^4)$ as we need to compute the SVD of X.

   It remains to prove (4.1). First, by part (a) of Problem 1, we have

   $$\|A - U_k U_k^T A\|_F^2 \le \|A\|_F^2 - \|A^T U_k\|_F^2.$$

Show that

$$\left| \|A^T U_k\|_F^2 - \sum_{i=1}^{k} \sigma_i^2 \right| \leq \sqrt{k} \|AA^T - XX^T\|_F.$$

You can use without proof that

$$\left| \sum_{i=1}^{k} \sigma_i^2 - \sum_{i=1}^{k} \sigma_i(A)^2 \right| \leq \sqrt{k} \|AA^T - XX^T\|_F,$$

where $\sigma_i(A)$ is the i-th largest singular value of A. Use the above two equations to conclude (4.1).

3) In this problem we see applications of expander graphs in optimization. In particular, we see that the maximum cut problem is easy in *strong* expander graphs. First, we explain the expander mixing lemma which asserts that expander graphs are very similar to complete graphs.

Theorem 4.1 (Expander Mixing Lemma)**.** *Let G be a d-regular graph and $1 = \lambda_1 \geq \lambda_2 \geq \ldots \lambda_n \geq -1$ be the eigenvalues of the normalized adjacency matrix of G, A/d. Let $\lambda^* = \max\{\lambda_2, |\lambda_n|\}$. Then, for any two disjoint sets $S, T \subseteq V$,*

$$\left| |E(S,T)| - \frac{d \cdot |S| \cdot |T|}{n} \right| \leq d \cdot \lambda^* \sqrt{|S||T|}.$$

Note that $d|S||T|/n$ is the expected number of edges between S, T in a random graph where is an edge between each pair of vertices i, j with probability d/n. So, the above lemma says that in an expander graph, for any large enough sets $|S|, |T|$, then the number of edges between S, T is very close to what you see in a random graph.

Use the above theorem to design an algorithm for the maximum cut problem that for any d regular graph returns a set T such that

$$|E(T, \overline{T})| \geq (1 - 4\lambda^*) \max_S |E(S, \overline{S})|.$$

Note that the performance of such an algorithm may be terrible if $\lambda^* > 1/4$, but in strong expander graphs, we have $\lambda^* \ll 1$; for example, in Ramanujan graphs we have $\lambda^* \leq 2/\sqrt{d}$. So the number of edges cut by the algorithm is very close to optimal solution as $d \to \infty$. It turns out that in random graph $\lambda^* \leq 2/\sqrt{d}$ with high probability. So, it is easy to give a $1 + O(1/\sqrt{d})$ approximation algorithm for max cut in most graphs.

4) In this problem you are supposed to implement the spectral partitioning algorithm that we discussed in class. You are given a giant network, "com-DBLP" input in https://snap.stanford.edu/data/ and you should find a sparse cut in this network. My code has found a cut of sparsity about 1%. Note that since the graph is huge you need to carefully store the edges of this graph. You should also use the power method to find the 2nd smallest eigenvalue of the normalized Laplacian matrix. In the output you should write the sparsity of the cut that you find and the id of the vertices in the smaller side of the cut. Please submit your code and the output to Canvas.

5) **Extra Credit.** In this problem we see applications of expander graphs in coding theory. Error correcting codes are used in all digital transmission and data storage schemes. Suppose we want to transfer m bits over a noisy channel. The noise may flip some of the bits; so 0101 may become 1101. Since the transmitter wants that the receiver correctly receives the message, he needs to send $n > m$ bits encoded such that the receiver can recover the message even in the presence of noise. For example, a naive way is to send every bit 3 times; so, 0101 becomes 000111000111. If only 1 bit were flipped in the transmission receiver can recover the message but even if 2 bits are flipped, e.g., 110111000111 the recover is impossible. This is a very inefficient coding scheme.

An error correcting code is a mapping $C : \{0,1\}^m \to \{0,1\}^n$. Every string in the image of C is called a codeword. We say a coding scheme is linear, if there is a matrix $M \in \{0,1\}^{(n-m)\times n}$ such that for any $y \in \{0,1\}^n$, y is a codeword if and only if

$$My = 0.$$

Note that we are doing addition and multiplication in the field F_2.

a) Suppose C is a linear code. Construct a matrix $A \in \{0,1\}^{n\times m}$ such that for any $x \in \{0,1\}^m$, Ax is a code word and that for any distinct $x, y \in \{0,1\}^m$, $Ax \neq Ay$.

The rate of a code C is defined as $r = m/n$. Codes of higher rate are more efficient; here we will be interested in designing codes with r being an absolute constant bounded away from 0. The Hamming distance between two codewords c^1, c^2 is the number of bits that they differ, $\|c^1 - c^2\|_1$. The minimum distance of a code is $\min_{c^1,c^2} \|c^1 - c^2\|_1$.

b) Show that the minimum distance of a linear code is the minimum Hamming weight of its codewords, i.e., $\min_c \|c\|_1$.

Note that if C has distance d, then it is possible to decode a message if less than $d/2$ of the bits are flipped. The minimum relative distance of C is $\delta = \frac{1}{n} \min \|c^1 - c^2\|_1$. So, ideally, we would like to have codes with constant minimum relative distance; in other words, we would like to say even if a constant fraction of the bits are flipped still one can recover the original message.

Next, we describe an error correcting code scheme based on bipartite expander graphs with constant rate and constant minimum relative distance. A $(n_L, n_R, D, \gamma, \alpha)$ expander is a bipartite graph $G(L \cup R, E)$ such that $|L| = n_L, |R| = n_R$ and every vertex of L has degree D such that for any set $S \subseteq L$ of size $|S| \leq \gamma n_L$,

$$N(S) \geq \alpha |S|.$$

In the above, $N(S) \subseteq R$ is the number of neighbors of vertices of S. One can generate the above family of bipartite expanders using ideas similar to Problem 1. We use the following theorem without proving it.

Theorem 4.2. *For any $\epsilon > 0$ and $m \leq n$ there exists $\gamma > 0$ and $D \geq 1$ such that a $(n, m, D, \gamma, D(1-\epsilon))$-expander exists. Additionally, $D = \Theta(\log(n_L/n_R)/\epsilon)$ and $\gamma n_L = \Theta(\epsilon n_R/D)$.*

Now, we describe how to construct the matrix M. We start with a $(n_L, n_R, D, \gamma, D(1-\epsilon))$ expander for $n_L = n, n_R = n - m$. For our calculations it is enough to let $n = 2m$. We name the vertices of L, $\{1, 2, \ldots, n\}$; so each bit of a codeword corresponds to a vertex in L. We let $M \in \{0,1\}^{(n-m)\times n}$ be the Tutte matrix corresponding to this graph, i.e., $M_{i,j} = 1$ if and only if the i-th vertex in R is connected to the j-th vertex in L. Observe that by construction this code has rate $1/2$. Next, we see that δ is bounded away from 0.

c) For a set $S \subseteq L$, let $U(S)$ be the set of unique neighbors of S, i.e., each vertex in $U(S)$ is connected to exactly one vertex of S. Show that for any $S \subseteq L$ such that $|S| \geq \gamma n$,

$$|U(S)| \geq D(1 - 2\epsilon)|S|.$$

d) Show that if $\epsilon < 1/2$ the minimum relative distance of C is at least γn.

The decoding algorithm is simple to describe but we will not describe it here.