CSE 521 Problem Set 4

Tyler Chen

Problem 1

Let G be a graph, and let $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of the normalized Laplacian matrix of G, $\tilde{L}_G$. Show that G is bipartite if and only if $\lambda_n = 2$.

Solution

Note: I did this problem before the hint came out so I used a different approach.

Recall that a graph $G = (V, E)$ is bipartite if there is a set $S \subset V$ such that for each edge $e = (u, v) \in E$, one of u, v is in S and the other is in $\overline{S}$.

For convenience define $\mathbb{1}_S[v] = +1$ if $v \in S$ and -1 otherwise. For a set $S \subset V$ define v_S to be the vector with i-th entry $\mathbb{1}_S[i]$.

We use D to denote the matrix with degrees on the diagonal, and denote the degree of vertex i by d_i. Similarly, we use A do denote the adjacency matrix of G. Recall that $L_G = D - A$ and $\tilde{L}_G = D^{-1/2} L_G D^{-1/2} = I - D^{-1/2} A D^{-1/2}$.

Now note that since $\|A\| \leq \max_i d_i$ (by Gershgorin's circle theorem),

$$\|D^{-1/2} A D^{-1/2}\| \leq \|D^{-1/2}\| \, \|A\| \, \|D^{-1/2}\| = \frac{1}{\max_i d_i} \|A\|_2 \leq 1$$

This also means that,

$$\left\|\tilde{L}_G\right\| = \left\|I - D^{-1/2} A D^{-1/2}\right\| \leq \|I\| + \left\|D^{-1/2} A D^{-1/2}\right\| \leq 2$$

For symmetric matrices, the spectral norm gives the magnitude of the largest eigenvalue. Therefore, $|\lambda_n| \leq 2$ so $\lambda_n = 2$ if and only if -1 is an eigenvalue of $D^{-1/2} A D^{-1/2}$.

It therefore suffices to show that -1 is an eigenvalue of $D^{-1/2} A D^{-1/2}$ if and only if G is bipartite.

First, suppose G is bipartite and let v_S be defined as above, where S is one of the sides of G. Then, if $j \sim i$, $\mathbb{1}_S[j] = -\mathbb{1}_S[i]$.

Now, note that the i-th row of A has nonzero entries only in coordinates corresponding to vertices in the opposite side of the bipartite vertex partition as i. Therefore,

$$(A v_S)_i = \sum_{j=1}^{n} A_{i,j} (v_S)_i = \sum_{j \sim i} A_{i,j} (v_S)_j = \sum_{j \sim i} A_{i,j} \mathbb{1}_S[j] = \mathbb{1}_S[j] d_i = -\mathbb{1}_S[i] d_i$$

We then have,

$$(D^{-1/2} A D^{-1/2} D^{1/2} v_S)_i = (D^{-1/2} A v_S)_i = -\mathbb{1}_S[i] \sqrt{d_i} = -(D^{-1/2} v_S)_i$$

This proves $D^{1/2} v_S$ is an eigenvector of $D^{-1/2} A D^{-1/2}$ with eigenvalue -1.

We now prove the converse. If G is disconnected, the graph is trivially bipartite. Assume G is connected and suppose -1 is an eigenvalue of $D^{-1/2} A D^{1/2}$. That is, there is some vector $x \neq 0$ such that,

$$D^{-1/2} A D^{-1/2} x = -x$$

Equivalently, there is some vector $y = D^{1/2}x \neq 0$ such that,

$$Ay = -y$$

since G is connected, meaning D is full rank.

WLOG assume $\|y\|_\infty = 1$ and let i be a row such that $|y_i| = 1$. Then,

$$\sum_{j=1}^{n} A_{i,j}y_j = (Ay)_i = -(Dy)_i = -d_i y_i = -\operatorname{sgn}(y_i)d_i$$

Now note that $A_{i,j} \in \{0, 1\}$ and by definition of d_i, exactly d_i of $\{A_{i,j} : j = 1, \dots, n\}$ are nonzero. Moreover, $|y_j| \leq 1$ by our choice of y.

A sum with d_i terms each at most one in magnitude has magnitude d_i only if each term has magnitude 1. In particular, we this means that $y_j = -\operatorname{sgn}(y_i)$ for each $j \sim i$; i.e. triangle inequality in $\mathbb{R}$ is tight if and only if all signs are the same.

For each $j \sim i$, we can repeat the argument. At each step, all the vertices k_j connected to j will be shown to have modulus one. Since G is connected, this means that in at most $\operatorname{diam}(G) < \infty$ steps each vertex will have been touched by some sum and therefore shown to have modulus one. Therefore, since y is real, all entries are $+1$ or -1.

Clearly y gives the left and right vertex sets of G as $\operatorname{sgn}(j) = -\operatorname{sgn}(i)$ if $i \sim j$. That is, for every edge, the two vertices of the edge belong to separate classes.

Problem 2

We say a graph G is an expander graph if the second eigenvalue of the normalized Laplacian matrix $\tilde{L}_G$, λ_2 is at least a constant independent of the size of G. It follows by Cheegers inequality that if G is an expander, then $\phi(G) \geq \Omega(1)$ independent of the size of G. It turns out that many optimization problems are "easier" on expander graphs. In this problem 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. (Expander Mixing Lemma). Let G be a d-regular graph and $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq -1$ be the eigenvalues of the normalized adjacency matrix of G, A/d. Let $\lambda^* = \max\{\lambda_2, |\lambda_n|\}$. Then for any two disjoints sets $S, T \subset V$,

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

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 $1 + \mathcal{O}(1/\sqrt{d})$ approximation algorithm for max cut in most graphs.

Solution

Intuitively, the expander mixing lemma tells us that if λ^* is small (as in the case of expanders), that sets of the same size have about the same cut.

First observe, that since $k(n-k)$ is maximized at $k = n/2$,

$$\max_{T \subseteq V} |T||\overline{T}| = |T|(n - |T|) = \max_{k \in \{0,\ldots,n\}} k(n-k) \leq \frac{n^2}{4}$$

Let S be a set attaining the max cut. Then, for any $T \subseteq V$,

$$d\sqrt{|T||\overline{T}|} \leq \frac{dn}{2} = |E| \leq 2|E(S, \overline{S})|$$

By the expander mixing lemma we have,

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

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

Adding these together we have,

$$|E(S,\overline{S})| - |E(T,\overline{T})| + \frac{d}{n}\left(|T||\overline{T}| - |S||\overline{S}|\right) \leq d\cdot\lambda^*\left(\sqrt{|S||\overline{S}|} + \sqrt{|T||\overline{T}|}\right) \leq 4\lambda^*|E(S,\overline{S})|$$

Therefore,

$$|E(T,\overline{T})| \geq (1-4\lambda^*)|E(S,\overline{S})| + \frac{d}{n}\left(|T||\overline{T}| - |S||\overline{S}|\right)$$

Now, note that by picking T to maximize $|T||\overline{T}|$, we can always choose $|T||\overline{T}| \geq |S||\overline{S}|$. In particular, taking $|T| = n/2$ if n is even (or $T = (n\pm 1)/2$ if n is odd) will guarantee,

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

Problem 3

You are given data containing grades in different courses for 5 students; say $G_{i,j}$ is the grade of student i in course j. (Of course, $G_{i,j}$ is not defined for all i, j since each student has only taken a few courses.) We are trying to "explain" the grades as a linear function of the students innate aptitude, the easiness of the course and some error term.

$$G_{i,j} = \text{aptitude}_i + \text{easiness}_j + \epsilon_{i,j}$$

where $\epsilon_{i,j}$ is an error term of the linear model. We want to find the best model that minimizes the sum of the $|\epsilon_{i,j}|$'s.

(a) Write a linear program to find aptitude_i and easiness_j for all i, j minimizing $\sum_{i,j} |\epsilon_{i,j}|$.
(b) Use any standard package for linear programming (Matlab/CVX, Freemat, Sci-Python, Excel etc.; we recommend CVX on matlab) to fit the best model to this data. Include a printout of your code, the objective value of the optimum, $\sum_{i,j} |\epsilon_{i,j}|$, and the calculated easiness values of all the courses and the aptitudes of all the students.

	MAT	CHE	ANT	REL	POL	ECO	COS
Alex		C+	B	B+	A−	C	
Billy	B−	A−			A+	D+	B
Chris	B−		B+		A−	B	B+
David	A+		B−	A		A−	
Elise		B−	D+	B+		B	C+

Assume A = 4, B = 3 and so on. Also, let B+ = 3.33 and A− = 3.67.

Solution

We first note that the solution will only be unique up to a constant factor, since we could add any constant to the each of the aptitude, and subtract the same constant from each of the easiness, without chancing the error.

Let $n = 5$ be the number of students and $m = 7$ be the number of classes. Define x to be the vector of length $n + m$, where the first n entries represent the aptitude of the students and last m entries represent the easiness of the classes. That is, $\text{aptitude}_i = x_i$ and $\text{easiness}_j = x_{n+j}$.

Now, reshape the grade matrix G to a vector g of nonzero entries, starting in the top left and moving across rows skipping any empty entries. The length of G should be $\text{nnz}(G)$. For convenience of notation, let $k \leftrightarrow i, j$ be the bijection mapping the indices of nonzero entries of G to their indices in g. That is, $g_k = G_{i,j}$.

At the same time, construct a coefficient matrix, C of size $\text{nnz}(G)$ by $n + m$, with rows corresponding to nonzero entries of $G_{i,j}$, where the i-th and $n + j$-th entries are 1 and all other entries are zero.

Then the k-th row of $g - Cx$ corresponds to $G_{i,j} - \text{aptitude}_i - \text{easiness}_j$. That is,

$$\epsilon_{i,j} = (g - Cx)_k = g_k - C_k x$$

We rewrite this as a linear program. First, observe that $\min \sum_{i,j} |\epsilon_{i,j}|$ is equivalent to $\min_k y_k$ subject to $y_k \geq \epsilon_{i,j}$ and $y_k \geq -\epsilon_{i,j}$.

This gives the constraints,

$$\begin{aligned} y_k &\geq g_k - C_k x & \Leftrightarrow \quad & -C_k x - y_k \leq -g_k \\ y_k &\geq -(g_k - C_k x) & \Leftrightarrow \quad & C_k x - y_k \leq g_k \end{aligned}$$

Writing these in matrix form yields the linear program,

$$\min \begin{bmatrix} 0^T & 1^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{s.t.} \qquad \begin{bmatrix} -C & -I \\ C & -I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} -g \\ g \end{bmatrix}$$

where 0^T is the all zeros row vector of length $n+m$, 1^T is the all ones row vector of length $\text{nnz}(G)$, and I is the identity of size $\text{nnz}(G)$.

We implement this in numpy/scipy using the built in linear program solver with the default method (simplex).

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import numpy as np
import scipy as sp
from scipy import optimize

# load data from problem
G_letter = [["","C+","B","B+","A-","C",""],
     ["B-","A-","","","A+","D+","B"],
     ["B-","","B+","","A-","B","B+"],
     ["A+","","B-","A","","A-",""],
     ["","B-","D+","B+","","B","C+"]]

# convert letter grade to GPA
grade_map = {"A+":4.33,"A":4,"A-":3.66,"B+":3.33,"B":3,"B-":2.66,"C
    +":2.33,"C":2,"D+":1.66,"":0}

# construct grade matrix
G = np.vectorize(lambda x: grade_map[x],otypes=[float])(np.array(G_letter
    ))

# get nonzero entries
nze = np.nonzero(G)
nnz = len(nze[0])
n,m = np.shape(G)

# construct flattened g matrix
g = G[np.nonzero(G)]

# construct coefficient matrix
C = np.zeros((nnz,n+m))
C[np.arange(nnz),nze[0]] = 1
C[np.arange(nnz),n+nze[1]] = 1
```

```
# set up linear program
A = np.block([[-C,-np.eye(nnz)],[C,-np.eye(nnz)]])
b = np.block([-g,g])
c = np.block([np.zeros(n+m),np.ones(nnz)])

# solve linear program
sol = sp.optimize.linprog(c,A,b)

# get easiness and aptitude
aptitude = sol.x[:n]
easiness = sol.x[n:n+m]
```

This gives the results:

Name	Alex	Billy	Chris	David	Elise
Aptitude	1.99	2.66	2.99	2.66	1.99

Class	MAT	CHE	AND	REL	POL	ECO	COS
Easiness	0	0.67	0.34	1.34	1.67	0.01	0.34

Problem 4

In the congestion minimization problem we are given a connected (undirected) graph $G = (V, E)$ and a set of pairs s_i, t_i of vertices of G for $1 \leq i \leq k$. We want to choose *exactly one* path between each pair s_i, t_i (k paths total) such that for each edge $e \in G$, the number of paths that use e is as small as possible. Consider the LP-relaxation of this problem:

$$\min z \qquad \text{s.t.} \qquad \forall e : \sum_{P : e \in P} f_P \leq z, \qquad \forall i : \sum_{P \in \mathcal{P}_{s_i,t_i}} f_P = 1, \qquad \forall P : \ f_P \geq 0$$

Here, $\mathcal{P}_{s_i,t_i}$ represent the set of all paths connecting s_i to t_i.

1. Prove that the above LP gives a relaxation of the problem
2. **Extra Credit**: Design an algorithm to round the solution to exactly one path connecting each s_i to t_i.
3. **Extra Credit**: Prove that your algorithm gives an approximation factor of $\mathcal{O}(\log n / \log \log n)$ to the problem.

Solution

(a) We first write the congestion problem using $\mathcal{P}_{s_i,t_i}$ as described above. We interpret the problem to be finding a set of k paths connecting s_i to t_i, such that the maximum number of overlaps on any edge is minimized.

For any path P, let $f_P = 1$ if path P is chosen, and 0 otherwise. Then, that each s_i has exactly one path to t_i means,

$$\forall i : \sum_{P \in \mathcal{P}_{s_i,t_i}} f_P = 1$$

For a given edge e, the number of paths using this edge is,

$$\sum_{P : e \in P} f_P$$

Finally, the problem is to minimize the maximum over all e of this quantity. With this definition of f_P we can write the congestion problem as,

$$\min z \quad \text{s.t.} \quad \forall e : \sum_{P : e \in P} f_P \leq z, \quad \forall i : \sum_{P \in \mathcal{P}_{s_i,t_i}} f_P = 1, \quad \forall P : \ f_P = \begin{cases} 1 & \text{if } P \text{ chosen} \\ 0 & \text{o.w.} \end{cases}$$

Thus, relaxing the final constraint to $f_P \geq 0$ gives the LP relaxation. That is, the optimum of the this problem is feasible in the LP.

(b) A naive deterministic algorithm is, for each i, to round the maximum of f_P for $P \in \mathcal{P}_{s_i,t_i}$ to one, and all others to zero. This will give k paths, each connecting s_i to t_i for $i = 1, \ldots, k$.

A naive randomized algorithm is, for each i, to pick P from $\mathcal{P}_{s_i,t_i}$ with probability f_P.

(c) We will show,

$$\text{RD} \leq \frac{\log n}{\log \log n}\text{OPT-LP} \leq \frac{\log n}{\log \log n}\text{OPT}$$

Note that the second inequality is immediate since the linear program we use is a relaxation of the original problem.

I didn't actually solve this, but maybe the approach would be to first compute the expected congestion of an edge, then bound the probability it is within some error of the LP optimum, and then use union bound to compute the probability that the maximum congestion over all edges is within this error of the LP optimum.

Fix an edge e. Let N_e be the congestion on edge e after rounding. Then, the expected number of paths which contain e after applying the rounding algorithm is,

$$\mathbb{E}[N_e] = \sum_{P:e\in P} \mathbb{P}[\text{pick path } P \text{ from some } \mathcal{P}_{s_i,t_i}] = \sum_{P:e\in P} f_P \leq z$$

By Chernoff bound we have,

$$\mathbb{P}[N_e < (1+\epsilon)z] \geq \mathbb{P}[N_e < (1+\epsilon)\mathbb{E}[N_e]] \geq 1 - \exp\left(-\frac{\epsilon^2\mathbb{E}[N_e]}{2+\epsilon}\right)$$

Problem 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 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, $\left\|c^1 - c^2\right\|_1$. The minimum distance of a code is $\min_{c^1,c^2} \left\|c^1 - c^2\right\|_1$.

(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 $\delta = \frac{1}{n}\min \left\|c^1 - c^2\right\|_1$. 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 $S \subset L$ of size $|S| \leq \gamma n_L$,

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

In the above, $N(S) \subset R$ are the 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. 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 $S \subset L$, 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 $S \subset L$ such that $|S| \leq \gamma n$,

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

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

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

Solution

(a) Since C is a linear code, there is a matrix $M \in \{0,1\}^{(n-m)\times n}$ such that $My = 0$ if and only if y is a codeword.

We require that for all $x, y \in \{0,1\}^m$ that $Ax \neq Ay$ and $MAx = MAy = 0$.

The first condition means that A must injective. The second requires that the range of A be contained in the kernel of M.

If A is injective, it must have rank m. However, $\text{rank}(\ker(M)) \leq m$ so we also have $\text{rank}(\ker(M)) = m$. Therefore, we must pick A so that $\text{range}(A) = \ker(M)$.

(b) By taking $c_2 = 0$ it is obvious that $\min_{c^1,c^2} \|c^1 - c^2\|_1 \leq \min_c \|c\|_1$.

Suppose c^1, c^2 attain the minimum hamming distance. Since c^1 and c^2 are codewords, $Mc^1 = Mc^2 = 0$. But then $M(c^1 - c^2) = 0$. Therefore $\min_c \|c\|_1 \leq \min_{c^1,c^2} \|c^1 - c^2\|_1$.

This proves $\min_{c^1,c^2} \|c^1 - c^2\|_1 = \min_c \|c\|_1$.

(c) Fix some $S \subset L$ satisfying $|S| \leq \gamma n$. Then, by the theorem,

$$|N(S)| \geq D(1-\epsilon)|S|$$