15-859N Spectral Graph Theory and The Laplacian Paradigm, Fall 2018

Homework 2 Version 2 **Due: Wednesday Oct 17**

Gary Miller TAs: TBA

Instructions. Collaboration is permitted in groups of size at most three. You must write the names of your collaborators on your solutions and must write your solutions alone.

Question	Points	Score
1	25	
2	25	
3	25	
4	25	
Total:	100	

(25) 1. **Eigenvalues of Cartesian Products**

Let $G = (V, E, w)$ and $H = (V', E', w')$ be two non-negatively weighted simple graphs. Let $G \otimes H = (\bar{V}, \bar{E}, \bar{w})$ be their Cartesian product, where:

- The vertices are $\bar{V} = V \times V'$
- The edges are $\bar{E} = \{((x, x'), (y, y')) \,|\, [x = y \wedge (x', y') \in E'] \vee [x' = y' \wedge (x, y) \in E]\}$
- $\bar{w}((x, x'), (x, y')) = w'(x', y')$ and $\bar{w}((x, x'), (y, x')) = w(x, y)$.

1. Show that the eigenvalues of $L_{G\otimes H}$ are the direct sum of those of L_G and L_H. That is if the eigenvalues of L_G are $\{\lambda_1, \ldots, \lambda_n\}$ and those of L_H are $\{\mu_1, \ldots, \mu_m\}$ the those of $L_{G\otimes H}$ are $\{\lambda_i + \mu_j \,|\, 1 \le i \le n,\ 1 \le j \le m\}$
2. Show that the eigenvectors of $L_{G\otimes H}$ are the direct product of those of L_G and L_H. The direct product of column vectors $\bar{a}$ and $\bar{b}$ is the matrix ab^T, reformatted (flattened) as a vector.

(25) 2. **A Bad Example for Spectral Partitioning**

Define **threshold spectral partitioning** of a possibly weighted graph $G = (V, E)$ to be the vertex partition one gets by; 1) Finding the eigenvector x for λ_2, 2) Sorting the vertices by their value in x, and 3) Returning the best threshold cut.

Let P_n^ϵ, for n even, be the weighted path graph on n vertices where all the edges have unit weight except the middle one which has weight ϵ.

Consider the Cartesian product $M_{cn}^{\epsilon} = P_{c\sqrt{n}} \otimes P_{\sqrt{n}}^{\epsilon}$

1. Show that threshold spectral partitioning on the graph M_{cn}^{ϵ} will generate a quotient cut of size $\Omega(1/\sqrt{n})$ for c sufficiently large and $\epsilon = 1/\sqrt{n}$ while the best cut is of size $O(1/n)$.
2. How big does c need to be for this to happen?

(25) 3. **Cut and Eigenvalues of the Product-Demand Graph**

In this problem we define the product-demand graph and give bounds on the isoperimetric cut and the fundamental vector of its Laplacian.

Let $\mu = (\mu_1, \ldots, \mu_n)$ be a vector of positive real numbers.

Definition 1 *The* **Product-Demand Graph for** μ *is a graph* $P_\mu = (V, E, w, \mu)$ *such that:*

1. $V = \{V_1, \ldots, V_n\}$
2. *Each edge* E_{ij} *has weight* $w_{ij} = \mu_i \cdot \mu_j$
3. *The* **Mass** *of vertex* V_i *is* μ_i

We denote this graph by P_μ *and its Laplacian by* L_μ*.*

Define the Mass of $S \subseteq V$, denote $\text{Mass}(S) = \sum_{i \in S} \mu_i$.

The **cut** from S to $\overline{S}$ is $Cut(S, \overline{S}) = \sum_{i \in S} \sum_{j \in \overline{S}} w_{ij} = \text{Mass}(S) \cdot \text{Mass}(\overline{S})$

The **Isoperimetric Number** for a graph G is:

$$\Phi(G) = \min_{S \subset V} \frac{Cut(S, \overline{S})}{\min\{\text{Mass}(S), \text{Mass}(\overline{S})\}} \tag{1}$$

Observe that the degree of $deg(V_i) = \mu_i \sum_{j \neq i} \mu_j < \mu_i \cdot \bar{\mu}$ where $\bar{\mu} = \sum \mu_i$.

1. Show the following bound on the Isoperimetric Number:

$$(1/2)\text{Mass}(V) \leq \Phi(P_\mu) \leq \text{Mass}(V) \tag{2}$$

2. Show the following lower bound on λ_2:

$$\frac{\text{Mass}(P_\mu)}{8} \leq \lambda_2. \tag{3}$$

You may find it useful to use the Cheeger inequality from lecture.

(25) 4. **Constant Time Sampling**

Suppose we are given a set of real numbers $r_1, r_2, \ldots, r_n > 0$ such that $\sum_{i=1}^{n} r_i = 1$. and a constant time algorithm which generates number uniformly in the unit interval.

The goal is to computes the random variable $R \in \{1, \ldots, n\}$ such that the probability that $R = i$ is r_i.

It is easy to see how to compute a sample of R in $O(\log n)$ time per sample. How?

Give an algorithm that preprocesses $r_1, \ldots, r_n$ such that samples can be generated in constant time per sample. Your algorithm should do the preprocessing in $O(n))$ time.