Laplace, Fiedler, & Markov: A Graph Reconstruction Problem

Garrett J. Kepler
Washington State University
Combinatorics, Linear Algebra, Number Theory Seminar
September 23rd, 2024

Dedication

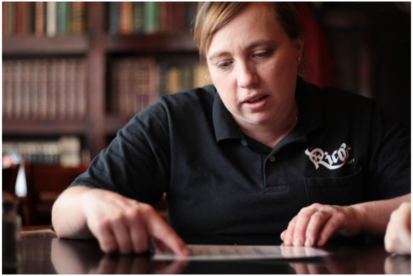
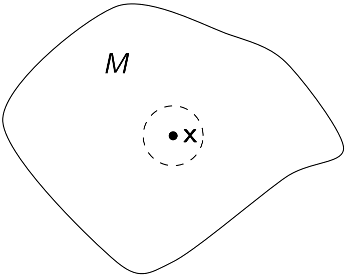


Figure 1: Tawny - Human Extraordinaire

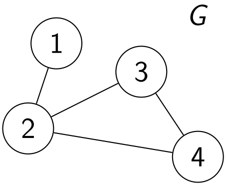
Question (Asked at Rico's)

There are 3 bags with k objects in each. Each pair of bags share $k-1$ objects though no bag is the same. How many objects *must* all three share?

- Let $\mathbf{x}$ be some element in a set M. Define some function u over points in M (i.e. temperature).
- At $\mathbf{x}$, calculate the difference between the value of u over some neighborhood of $\mathbf{x}$ and its value at $\mathbf{x}$ (i.e. $\Delta u_x = \frac{d^2u}{dx_1^2} + \frac{d^2u}{dx_2^2} + \cdots + \frac{d^2u}{dx_n^2}$). Think second derivative test.
- The Laplacian Δu can tell us about the "propagation" of u over M.



- Let $\mathbf{i}$ be some node in a graph G. Define some function f over nodes in G (i.e. temperature).
- At $\mathbf{i}$, calculate the difference between the value of f over the neighbors of $\mathbf{i}$ and its value at $\mathbf{i}$ (i.e. $L(f(\mathbf{i})) = \sum_{j\sim\mathbf{i}} f(\mathbf{i}) - f(j)$).
- The Laplacian L can tell us about the "propagation" of f through G.



Questions of Quantity

Question

Given a function u over M, how can we measure how well u propagates?

Question

Given a function f over G, how can we measure how well f propagates?

Questions of Quantity

Answer

In the discrete case, through the exploration of the *edge conductance* $\phi(G)$ for $S \subset V$:

$$\phi(S) = \frac{|E(S, V \setminus S)|}{vol(S)} \text{ and } \phi(G) = \min_S \phi(S)$$

where $vol(S) = \sum_{v \in S} \deg(v)$

Questions of Location

Question

Given a set M, how can we find where u does not propagate well?

Question

Given a graph G, how can we find where f does not propagate well?

Answer

This line of questioning leads to the *Sparsest Cut Problem*. The goal of which is to find the edges $E(S, V \setminus S)$ such that $\phi(S)$ is minimized. This problem is very much NP Hard, but approximation algorithms exist. Some rely on what we call the Fiedler vector!

Preliminaries

Definition

Let $G = (V, E)$ be an undirected graph on n nodes. The *Laplacian* of G is the matrix $L = L(G) \in M_n$ defined as follows:

$$L_{ij} = \begin{cases} \text{degree of node } i & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and node } i \sim \text{node } j \\ 0 & \text{if } i \neq j \text{ and node } i \nsim \text{node } j \end{cases}$$

with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.
Properties:

- L is symmetric, positive-semidefinite $\implies$ L has nonnegative eigenvalues
- 0 is an eigenvalue of L corresponding to the all ones vector $\mathbb{1}$

Fiedler Miracles

Definition

We call the second smallest eigenvalue of L, λ_2, and its corresponding eigenvector the *Fiedler value* and *Fiedler vector* respectively.

Theorem

For undirected G, we obtain Cheeger's Inequality:

$$\frac{\lambda_2}{2} \leq \phi(G) \leq \sqrt{2\lambda_2}$$

Informally, the amount of "bottlenecking" in a graph is bounded by the Fiedler value.

Fiedler Miracles

Theorem

For undirected G, we obtain Cheeger's Inequality:

$$\frac{\lambda_2}{2} \leq \phi(G) \leq \sqrt{2\lambda_2}$$

Informally, the amount of "bottlenecking" in a graph is bounded by the Fiedler value.

Theorem

Let $G = (V, E)$ be a connected graph and let x be its corresponding Fiedler vector. Then the subgraphs induced by the vertex sets $V_1 = \{i : x_i > 0\}$ and $V_2 = V \setminus V_1$ are connected. Informally, we can partition G in to two connected components using the entries of the Fiedler vector.

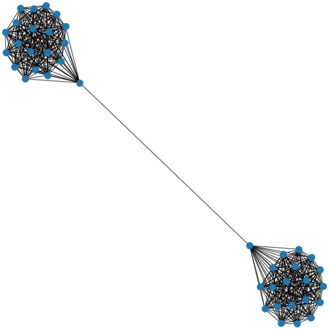
1. Motivation
2. Laplacian and Fiedler
3. **Partitioning Process**
4. Computational Methods
5. Reconstructibility
6. Quarrels, Queries, and Quotes

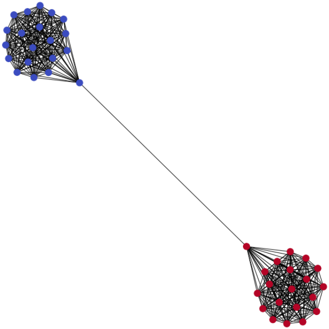
To Do

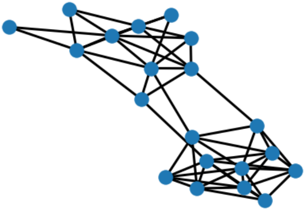
1. partitioning process and examples with their fiedler vector and value:
    1. graph with phi fiedler entries
    2. real nasty horrible unintuitive graph
    3. another fun graph

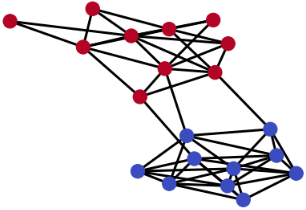
Intuition:

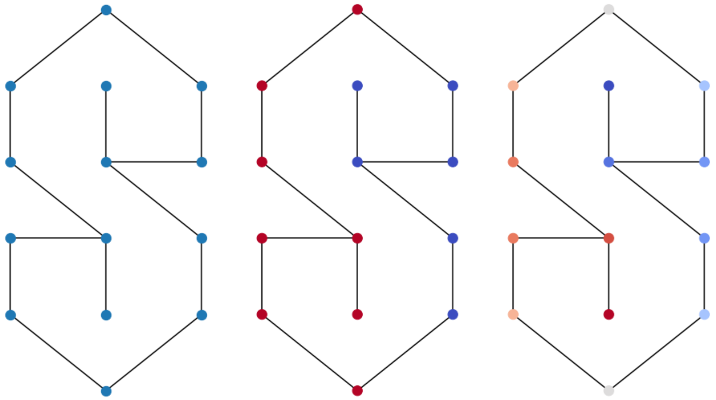
- There is a bottleneck in the graph G.
- The bottleneck is approximated by the Fiedler value.
- Using the Fiedler vector, cutting edges that connect oppositely signed nodes will result in a relatively efficient bi-partition cut.

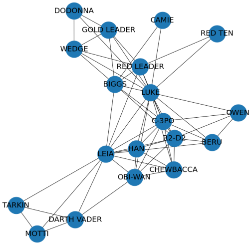


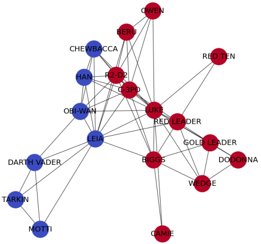


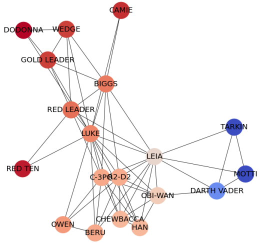


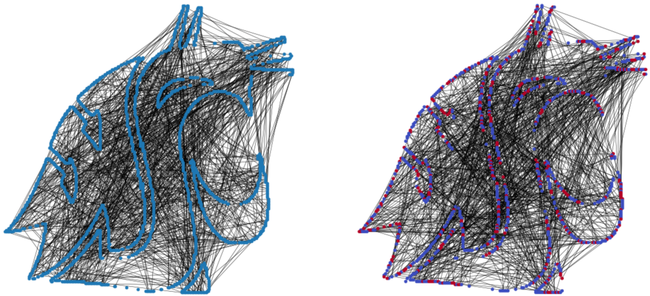


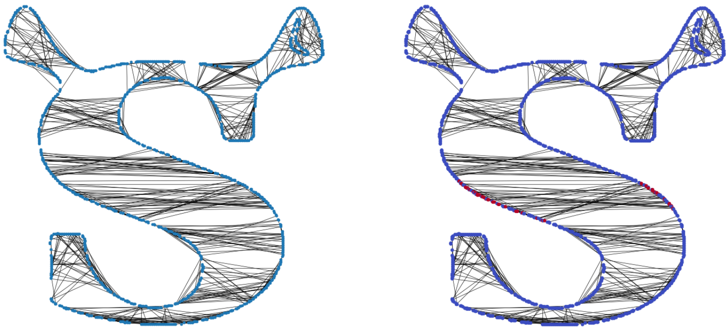



DODONNA
GOLD LEADER
CAMIE
RED TEN
WEDGE
RED LEADER
BIGGS
LUKE
OWEN
C-3PO
R2-D2
BERU
LEIA
HAN
CHEWBACCA
OBI-WAN
TARKIN
DARTH VADER
MOTTI

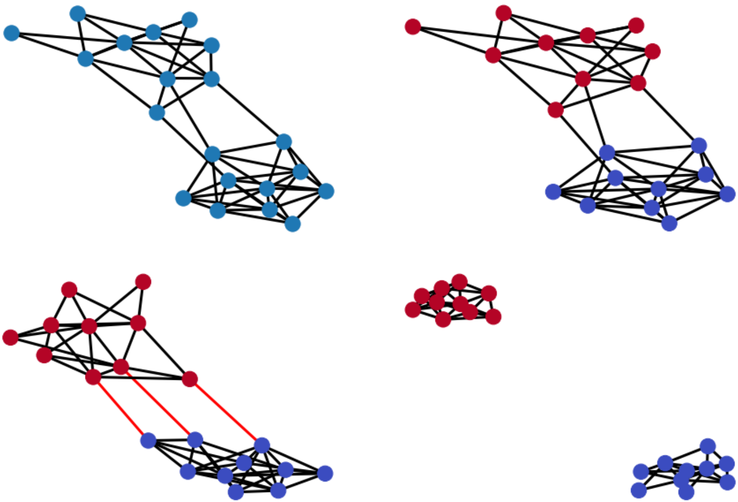

OWEN
BERU
CHEWBACCA
RED TEN
HAN
R2-D2
C-3PO
OBI-WAN
LUKE
RED LEADER
LEIA
GOLD LEADER
DODONNA
BIGGS
DARTH VADER
WEDGE
TARKIN
MOTTI
CAMIE


CAMIE
DODONNA
WEDGE
GOLD LEADER
BIGGS
RED LEADER
TARKIN
LUKE
LEIA
RED TEN
MOTTI
C-3PO
R2-D2
DARTH VADER
OBI-WAN
CHEWBACCA
OWEN
HAN
BERU





The Process



Question

Given a Fiedler-partitioned graph G, is the Fiedler cut unique for every x or every λ? Is it unique for every pair (x, λ)?

Question

Given a x, λ, or the pair (x, λ), is there a way to generate graphs with (x, λ) as their Fiedler vector and Fiedler value?

MCMC!

Brief Overview of Metropolis-Hastings

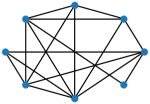
General Idea:

- We want to sample from distribution P over space X, but we cannot compute probabilities directly.
- So, we apply some score function f over X and pick a proposal distribution Q over X (ideally one that we know/that is easy to compute/has specific properties/etc).
- We start at some state x_n and generate a proposal state y using Q.
- Comparing our scores, we compute an acceptance probability
$$\alpha = \min\left(1, \frac{f(y)}{f(x_n)} \frac{Q_{y,x_n}}{Q_{x_n,y}}\right)$$
- Pick a number β uniformly at random from $[0, 1]$.
- If $\beta < \alpha$, $x_{n+1} = y$. If not, $x_{n+1} = x_n$.

First Steps

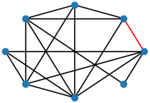
- Initialize: Start at some step x_n in our Markov chain.
- Proposal: Generate a proposed next step y.
- Transition: Calculate acceptance probability α. Compare to uniformly random $\beta \in [0, 1]$. Accept if $\beta < \alpha$. Stay otherwise.
- Convergence: Wait!

- Initialize:



$$\begin{bmatrix} -0.034\cdots \\ 0.132\cdots \\ 0.202\cdots \\ -0.062\cdots \\ 0.291\cdots \\ 0.183\cdots \\ 0.173\cdots \\ -0.887\cdots \end{bmatrix}$$

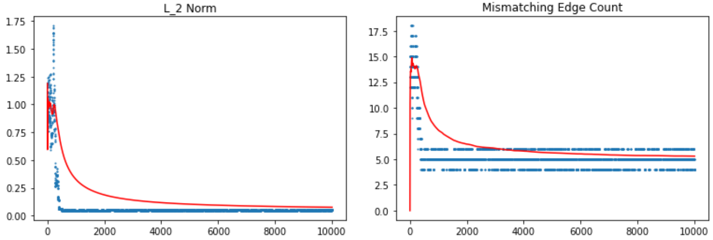
- Proposal:



$$\begin{bmatrix} -0.253\cdots \\ 0.432\cdots \\ 0.324\cdots \\ 0.063\cdots \\ -0.017\cdots \\ 0.082\cdots \\ 0.149\cdots \\ -0.781\cdots \end{bmatrix}$$

- Transition: Calculate α where $f(x) \propto ||\text{fied}_x - \text{fied}_{\text{desired}}||_2$ Compare. Accept/Reject.
- Convergence: Wait!

Results

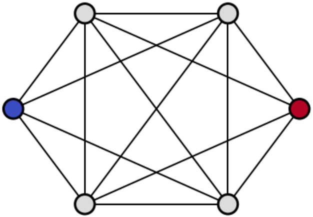

L_2 Norm
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
2000
4000
6000
8000
10000
Mismatching Edge Count
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
2000
4000
6000
8000
10000

MCMC Implications on Uniqueness

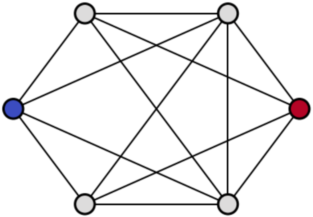
1. An immediate observation can be made from working with MCMC in this way:

   Depending on the desired graph, some proposal states will continue to be accepted even after convergence to within an arbitrary distance of the desired distribution.

2. In other words, some graphs do not have a unique adjacency structure corresponding to their Fiedler vector!



$\sigma(L(G)) = \{0, 4, 6, 6, 6, 6\}$



$\sigma(L(G)) = \{0, 4, 4, 6, 6, 6\}$

Combinatorial Intuition

- "Good Reconstructible":
  1. Given a Fiedler vector x and collection of neighborhoods S_i of node $i \in V(G)$
  2. What we want is the size of S_i to be as small as possible for every node in G
  3. $|S_i| = 1$ for all i in G implies uniquely reconstructible!
- "Bad Reconstructible":
  1. Given a Fiedler vector x and collection of neighborhoods S_i of node $i \in V(G)$
  2. What we want is the size of S_i to be as large as possible for every node in G

Hmm...

1. Can we guarantee existence or non-existence of any edge in G given (x, λ)?
2. Can we find a graph with a uniquely determined Fiedler cut? If so, count all the graphs like this.
3. We can reconstruct a graph entirely if we know all of its eigenvalues and eigenvectors. What properties of G allow us to reconstruct the graph if we know $k < n$ of them?
4. What methods allow us to most quickly estimate the edges in G?