Spectral Graph Theory & Laplacian Spectrum Optimization

ESTR4999 PRESENTATION: YU, LETIAN (SID:1155124365)

THE CHINESE UNIVERSITY OF HONG KONG

Background

- **Spectral Graph Theory**

Relationship between the graph structure and its spectral properties

- **Wide Range of Application**
  - Spectral clustering: associated eigenvalues and eigenvectors
  - Graph representation learning: Laplacian matrix
  - Chemistry, signal processing…

Definitions & Properties

- Properties of Eigenvalues: Rayleigh Quotients

Theorem 3.1: Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix, where $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$ are the eigenvalues and $x_1, x_2, \cdots, x_n$ are the corresponding orthonormal vectors $(\|x_i\|^2 = 1, \langle x_i, x_j \rangle = 0 \quad \forall i \ne j))$ such that $Ax_i = \lambda_i x_i$ for $i = 1, 2, \cdots, n$. For all $0 \le k \le n-1$, we have

$$\lambda_{k+1} = \min_{x\in\mathbb{R}^n : x \perp \text{span}(x_1,\cdots,x_k)} \frac{x^T A x}{x^T x}, \qquad (1)$$

and any optimal solution x would be the eigenvector related to λ_{k+1}.

Extensions:

$$\begin{aligned}
\lambda_k &= \min_{x\in\mathbb{R}^n : x \perp \text{span}(x_1,\cdots,x_{k-1})} \frac{x^T A x}{x^T x} \\
&= \min_{x\in\mathbb{R}^n : x \in \text{span}(x_k,\cdots,x_n)} \frac{x^T A x}{x^T x} \\
&= \max_{x\in\mathbb{R}^n : x \perp \text{span}(x_{k+1},\cdots,x_n)} \frac{x^T A x}{x^T x} \\
&= \max_{x\in\mathbb{R}^n : x \in \text{span}(x_1,\cdots,x_k)} \frac{x^T A x}{x^T x}.
\end{aligned}$$

Graph Algebraic Connectivity

- **Second smallest eigenvalue of the Laplacian matrix**
  - Fiedler value, graph algebraic connectivity
  - Related to graph connectivity
  - Measure of network robustness
  - Corresponding normalized eigenvector: Fiedler Vector
- **Normalized algebraic connectivity**
  - Second smallest eigenvalue of the normalized Laplacian matrix
  - Similar properties to algebraic connectivity

Definitions
&
Properties

Algebraic Connectivity:

- Connectivity
- Sparsity

Definitions & Properties

Define the **adjacency matrix** $A = [a_{ij}]$, where $a_{ij} = 1$ if $(i, j) \in E$, and $a_{ij} = 0$ otherwise.

Define *d(i)* as the degree of node *i* in graph *G*, and **degree matrix** $D \in R^{n \times n}$ is the diagonal matrix where *D(i, i) = d(i)*. Define the **graph Laplacian matrix** $L_G = D - A$, which is PSD.

Let $\lambda_2(L_G)$ be the second smallest eigenvalue of G, which is also the algebraic connectivity. For simplicity, we write $\lambda_2(L_G)$ as λ_2 here. (Note: λ_1 *is always zero*.)

- *Theorem 3.3:* $\lambda_2 = 0$ iff G is disconnected.

Extended Theorem:

- $\lambda_k = 0$ iff G has at least k components

Definitions & Properties

Define κ(G) to be the **vertex connectivity** of G, which is the smallest non-negative number of vertex removal so that the remaining graph is still connected.

Let $G - S$ be the resulting graph after removing vertices in S as well as all associated edges.

Then we have the following:

Lemma 3.4: $\lambda_2(L_G) \leq \lambda_2(L_{G-S}) + |S|$, for all $S \subseteq V$.

Corollary 1: $\lambda_2(L_G) \leq \kappa(G)$.

Proof: To prove this corollary, simply let S be a set of vertices of size $\kappa(G)$ that disconnects G.

$$\lambda_2(G-S) = 0 \Rightarrow \lambda_2(G) \leq 0 + \kappa(G)$$

Definitions & Properties

- $\lambda_2(L_G)$ is monotone increasing in the edge set:
  - If G1 = (V, E1) and G = (V, E2) s.t. E1 $\subseteq$ E2,

  → $\lambda_2(L_{G1}) \leq \lambda_2(L_{G2})$.

- The more connected graph (on the same vertex set) has the greater algebraic connectivity

Definitions & Properties: Sparsity

- Let X ⊆ V, and E_C be the subset of edges connecting X and its complement X^c.
- Then we have:

$$\lambda_2(L_G) \leq \min_{x \subseteq V} \frac{n|E_c|}{|X||X^c|}.$$

- A graph with large algebraic connectivity cannot have very sparse cuts
- $\lambda_2(L_G)$ is small for a graph with sparse cuts
- This is related to graph partitioning by Fiedler vector

Graph Algebraic Connectivity

- **Applications**
  - Neural network regularization term: connectivity & sparsity
  - Consensus protocol for networks of dynamic agents:
  - Air Transportation Networks
  - Mixing rate of Markov chain

Problem Statement

Given an undirected, unweight and connected graph $G=(V, E)$ with n nodes and m edges.

Let A be the adjacency matrix of G, D be the degree matrix, and $\mathcal{L}$ be the normalized Laplacian matrix. Define λ_2 to be the second smallest eigenvalue associated with $\mathcal{L}$.

By adding $k = \gamma m$ edges to the original graph, we would like to maximize the new λ_2.

In application, we set $k = \min(\gamma m, |\bar{E}|), \gamma < 1$. Here G supposes to be sparse, and $\bar{E}$ is the set of candidate edges.

Notation	Description
$G = (V, E)$	Undirected, unweighted and connected graph G
V	Node set
E	Edge set
$\bar{E}$	Set of potential edges that can be added
n, m	The number of nodes and edges in G
γ	The percentage of edges to be added
$k = \gamma m$	The number of edges to be added
A	The adjacency matrix of G
D	The degree matrix of G
$\mathcal{L}$	The normalized Laplacian matrix of G
λ_2	The second smallest eigenvalue associate with $\mathcal{L}$

Definitions & Properties

- Normalized Adjacency Matrix: $\mathscr{A} \equiv D^{-1/2}AD^{-1/2}$
- $$D^{-1/2} = \begin{pmatrix} \frac{1}{\sqrt{d(1)}} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sqrt{d(2)}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sqrt{d(n)}} \end{pmatrix}$$
- Normalized Laplacian Matrix: $\mathscr{L} = I - \mathscr{A} = D^{-1/2}(D-A)D^{-1/2} = D^{-1/2}L_G D^{-1/2}$

Definitions & Properties

- *Proposition 3.5:* Let $\alpha_1 \geq \cdots \geq \alpha_n$ be the eigenvalues of $\mathscr{A}$ and let $\lambda_1 \leq \cdots \lambda_n$ be the eigenvalues of $\mathscr{L}$. Then

$$1 = \alpha_1 \geq \cdots \geq \alpha_n \geq -1,$$

$$0 = \lambda_1 \leq \cdots \leq \lambda_n \leq 2.$$

- Connectivity: $\lambda_2(\mathscr{L}) = 0$ iff G is disconnected

Problem Statement

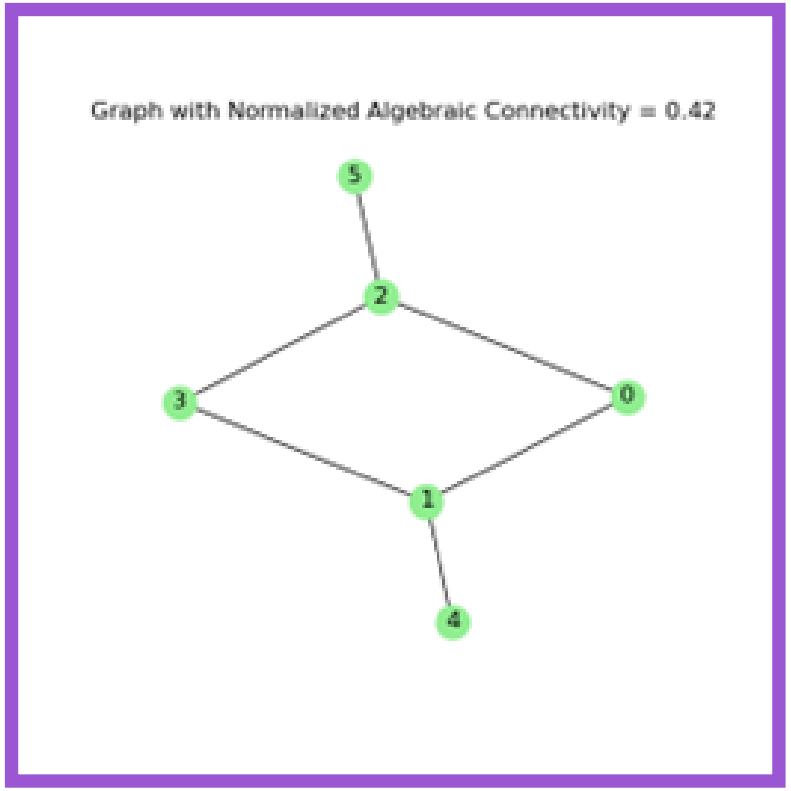
$$maximize\ \lambda_2\left(\mathcal{L}\left(\mathrm{E}\bigcup E_{add}\right)\right)$$

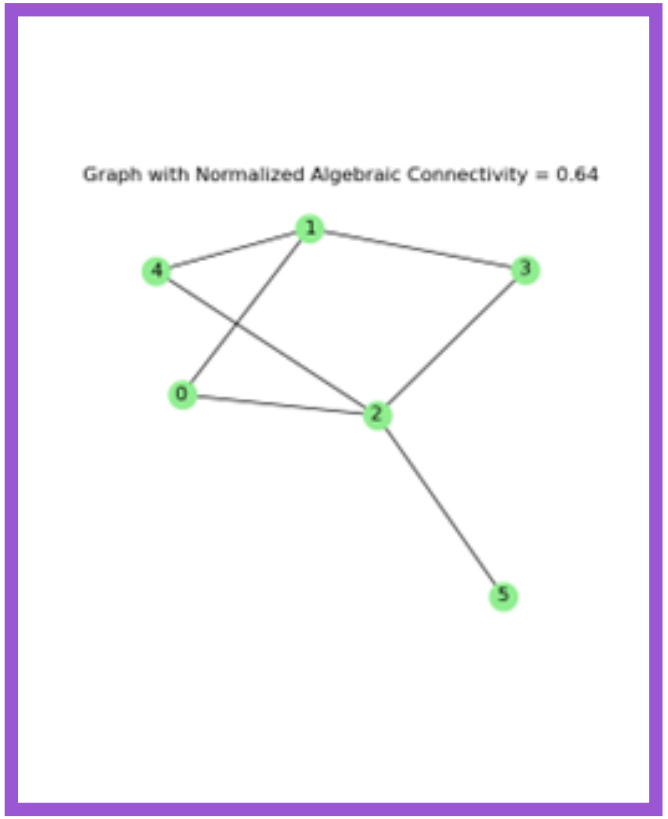
$$subject\ to\ |E_{add}| = \mathrm{k},$$
$$E_{add} \subseteq \bar{E},$$
$$k = \min(\gamma m,\ |\bar{E}|),$$
$$\gamma < 1.$$

Notation	Description
$G = (V, E)$	Undirected, unweighted and connected graph G
V	Node set
E	Edge set
$\bar{E}$	Set of potential edges that can be added
n, m	The number of nodes and edges in G
γ	The percentage of edges to be added
$k = \gamma m$	The number of edges to be added
A	The adjacency matrix of G
D	The degree matrix of G
$\mathscr{L}$	The normalized Laplacian matrix of G
λ_2	The second smallest eigenvalue associate with $\mathscr{L}$

Example for illustration

(k=1)





Related Work

- **edgewire (***H. Chan and L. Akoglu***)**: degree-preserving edge rewiring method to optimize robustness measures including the algebraic connectivity
- *A. Clark, Q, et al.; R. Wang, et al.*: using submodular optimization to maximize Fiedler value of the grounded Laplacian matrix (i.e., the principal submatrix of the original Laplacian matrix)
- *S. Fallat and S. Kirkland*: effects of component change of weighted graph on algebraic connectivity
- *G. Li, et al.*: MDMD algorithm to maximize algebraic connectivity
- *A. Ghosh and S. Boyd*: heuristic for edge addition strategy by SDP relaxation

Our Focus

- Edge Addition Problem
- Algorithm Implementation for Normalized Algebraic Connectivity
- Developed Codes with CUDA Acceleration
- Balance efficiency and effectiveness

Algorithms & Implementation

- Local Greedy Search
- Random Selection
- Greedy by Edge Rank
- Node Degree-based Selection
  - Min Degree Random Selection
  - Min Degree Max Distance

Local Greedy Search

- Global Maximum: combinatorial complexity $O\left(\binom{|\bar{E}|}{k}\right)$, where $|\bar{E}| \sim n^2$.
- Local Minimum: $O(k|\bar{E}|)$
- Optimal when $k = 1$
- The process of eigenvalue calculation is computationally expensive!

Algorithm 2: Greedy Search Local Maximum

Input: Graph G, k, candidate edge set $\bar{E}$
Output: Selected edge set for link addition

Initialization: Compute λ_2 of $\mathscr{L}$;
for *i from* 1 *to* k **do**
    Search edge set $\bar{E}$ for edge e that maximizes λ_2';
    Update G: $E \leftarrow E \cup \{e\}$, $\bar{E} \leftarrow \bar{E} \setminus \{e\}$;
    Append e to selected edge set;

Random Selection with Iteration

- Time complexity:

  O(k) for each iteration

- Still expensive for large graph with iteration

Algorithm 3: Random Selection

Input: Graph G, k, candidate edge set $\bar{E}$, *iter_num*
Output: Selected edge sets and result statistics

```
Initialization: Compute λ2 of ℒ;
for i from 1 to iter_num do
    Generate edge sequence of length k from Ē by
      random selection;
    Compute corresponding eigenvalues;
    Append edge sequence to selected edge sets;
Calculate mean and quantiles for derived result;
```

Greedy by Edge Rank

- Computationally efficient for edge selection
- rank_type ==’sum’, score((i, j)) = deg(i) +deg(j);
  rank_type ==’mul’, score((i, j)) = deg(i) × deg(j).
- greedy_method: max or min
- Drawback: the edge rank scheme might not differentiate between edges with different node degrees, say (1, 6) and (3, 4); the graph cut might not be affected.

Algorithm 4: Greedy by Edge Rank

Input: Graph G, k, candidate edge set $\bar{E}$, *rank_type*, *greedy_method*

Output: Selected edge set for link addition

```
Initialization: Compute λ2 of ℒ, compute edge rank
  for each edge based on rank_type;
for t from 1 to k do
    if greedy_method == 'max' then
        Select edge e = (i, j) of max edge rank;
    else
        Select edge e = (i, j) of min edge rank;
    Update G: E ← E ∪ e, Ē ← Ē \ e;
    Append e to selected edge set;
    Compute corresponding eigenvalue;
    Update edge rank;
```

Smallest Degree and Random Selection

- Experiments in literature shows that the increase of $\lambda_2(L_G)$ is large only if an edge with one node having smallest degree is chosen. We therefore adapt the idea to $\lambda_2(\mathcal{L})$
- Utilize graph topological structure
- Computationally efficient
- Random selection too many candidates, might not be effective!

Algorithm 5: Smallest Degree and Random Selection

Input: Graph G, k, candidate edge set $\bar{E}$
Output: Selected edge set

Initialization: Compute λ_2 of $\mathcal{L}$;
for t *from* 1 *to* k **do**
    Choose node i of minimum degree;
    Randomly choose j s.t. $e = (i, j)$ is in $\bar{E}$;
    Update G: $E \leftarrow E \cup e$, $\bar{E} \leftarrow \bar{E} \setminus e$;
    Update node degree and corresponding edge list;
    Compute corresponding eigenvalues;
    Append edge e to selected edge set;

Min Degree and Max Distance

- Computationally efficient: BFS **O(***k* * *(V+E)***)**
- Intuition: Cheeger's inequality, to add link between the sparse cut → find the node with maximum distance
- Combines topological structure

Algorithm 6: Min Degree and Max Distance

Input: Graph G, k, candidate edge set $\bar{E}$
Output: Selected edge set

Initialization: Compute λ_2 of $\mathscr{L}$;
for *t from* 1 *to k* **do**
    Choose node i of minimum degree;
    Choose j s.t. $distance(i, j)$ is maximized and $e = (i, j)$ is in $\bar{E}$;
    Update G: $E \leftarrow E \cup e$, $\bar{E} \leftarrow \bar{E} \setminus e$;
    Update node degree and corresponding edge list;
    Compute corresponding eigenvalues;
    Append edge e to selected edge set;

Properties of Normalized Algebraic Connectivity: Cheeger's Inequality

Definition 3.1: Let $S \subset V$, and $\delta(S)$ be the set of edges with exactly one endpoint in S. Define $\mathrm{vol}(S) \equiv \sum_{i \in S} d(i)$. Therefore, the conductance of S is

$$\phi(S) \equiv \frac{|\delta(S)|}{\min\{\mathrm{vol}(S), \mathrm{vol}(V-S)\}}.$$

The edge expansion of S is

$$\alpha(S) \equiv \frac{|\delta(S)|}{|S|}, \quad \text{for } |S| \leq \frac{n}{2}.$$

The sparsity of S is

$$\rho(S) \equiv \frac{|\delta(S)|}{|S||V-S|}.$$

Definition 3.2:

$$\phi(G) \equiv \min_{S \subset V} \phi(S),$$

$$\alpha(G) \equiv \min_{S \subset V : |S| \leq \frac{n}{2}} \alpha(S),$$

$$\rho(G) \equiv \min_{S \subset V} \rho(S).$$

Theorem 3.6: Let λ_2 be the smallest eigenvalue of $\mathscr{L}$, then

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

Intuition from Algebraic Connectivity

- We assume normalized algebraic connectivity is related to algebraic connectivity.
- For algebraic connective, $x_2^T L_G x_2 = \sum_{(i,j)\in E} (x_2(i) - x_2(j))^2$.
- Consider the linear embedding problem:

  Assign node coordinates x1, . . . , xn,
  with 0 mean and unit variance,
  so as to minimize the sum of squares
  of distances between adjacent nodes.

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{(i,j)\in E}(x_i - x_j)^2 \\ \text{subject to} \quad & \mathbf{1}^T x = 0, \\ & x^T x = 1. \end{aligned}$$

→ $\lambda 2(L)$ as optimal value

- This implies that adding an edge between the nodes that are farthest from each other in the linear embedding might help

Intuition from Algebraic Connectivity

- If we look at the Rayleigh Quotients (following as example, L is the unnormalized Laplacian matrix, i.e., LG):
- Previous constraints no longer hold, but give a similar flavor

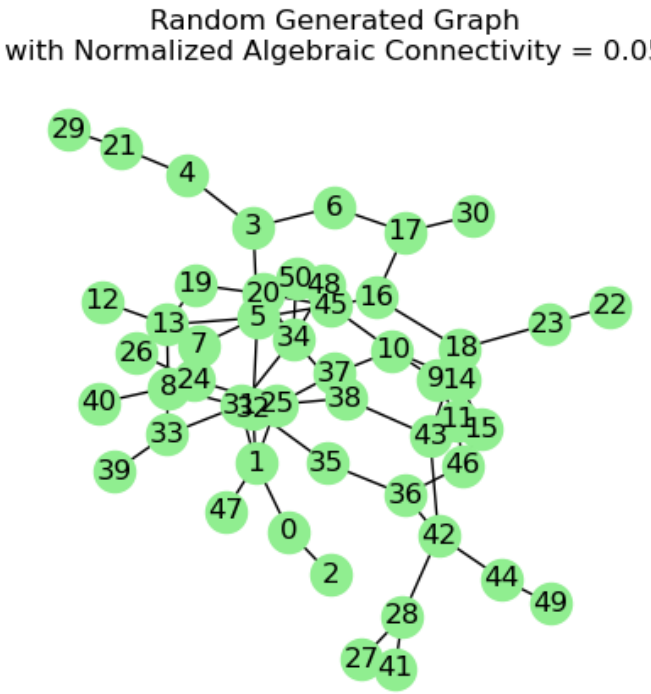
$$\lambda_2 = \min_{\mathbf{1}^T D\mathbf{y}=0,\ \mathbf{y}\neq\mathbf{0}} \frac{\mathbf{y}^T L\mathbf{y}}{\mathbf{y}^T D\mathbf{y}}.$$

λ_2 *is obtained when* $\mathbf{y} = D^{-1/2}\mathbf{v}_2$ *where* $\mathbf{v}_2$ *is an eigenvector to* λ_2.

- Remember that

$$x_2^T L_G x_2 = \sum_{(i,j)\in E} (x_2(i) - x_2(j))^2$$

Experiment: Dataset



Preprocessing:

1. Convert the graph to undirected
2. Choose the largest connected component as input

Dataset	Number of nodes	Number of edges
Random	51	64
Random2	29	45
Random3	124	258
Random4	256	1012
CORA	2485	5069

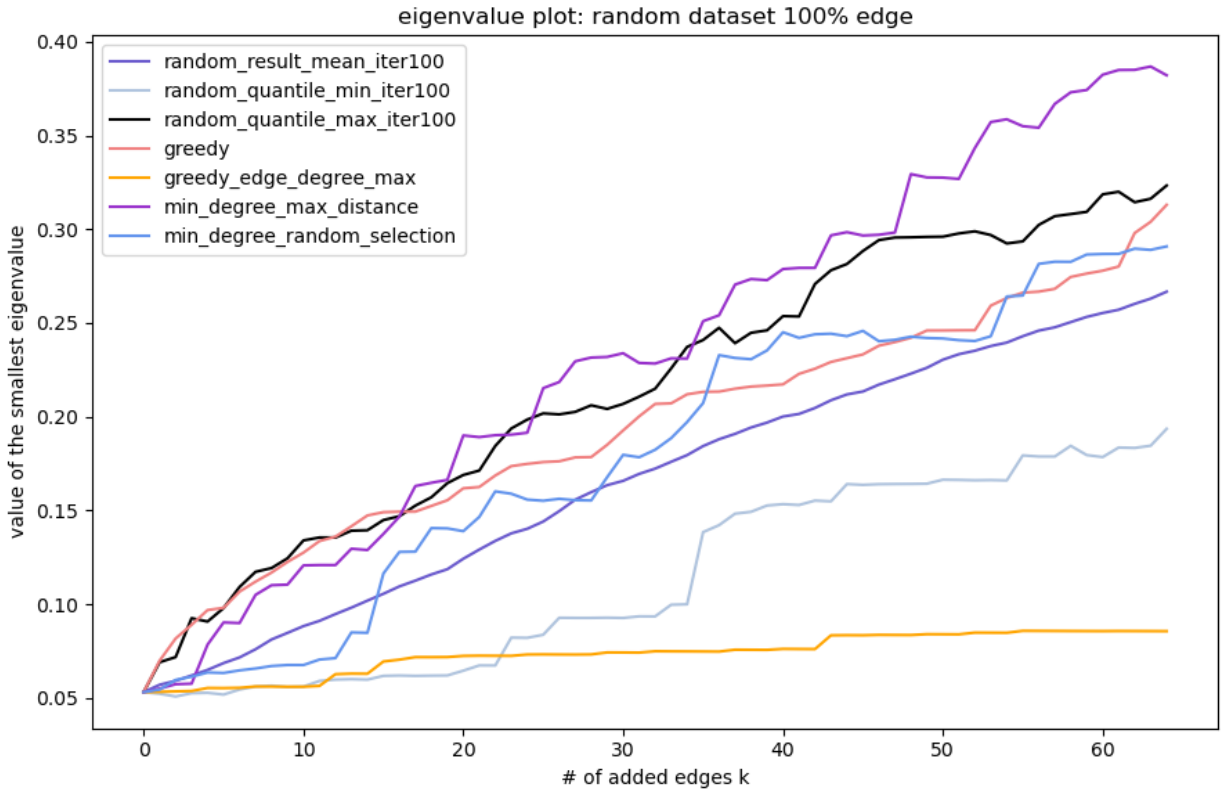
TABLE II: Datasets After Preprocessing

Experiment: Running Time Comparison

Computed on GPU, which is much faster than parallel CPU computing

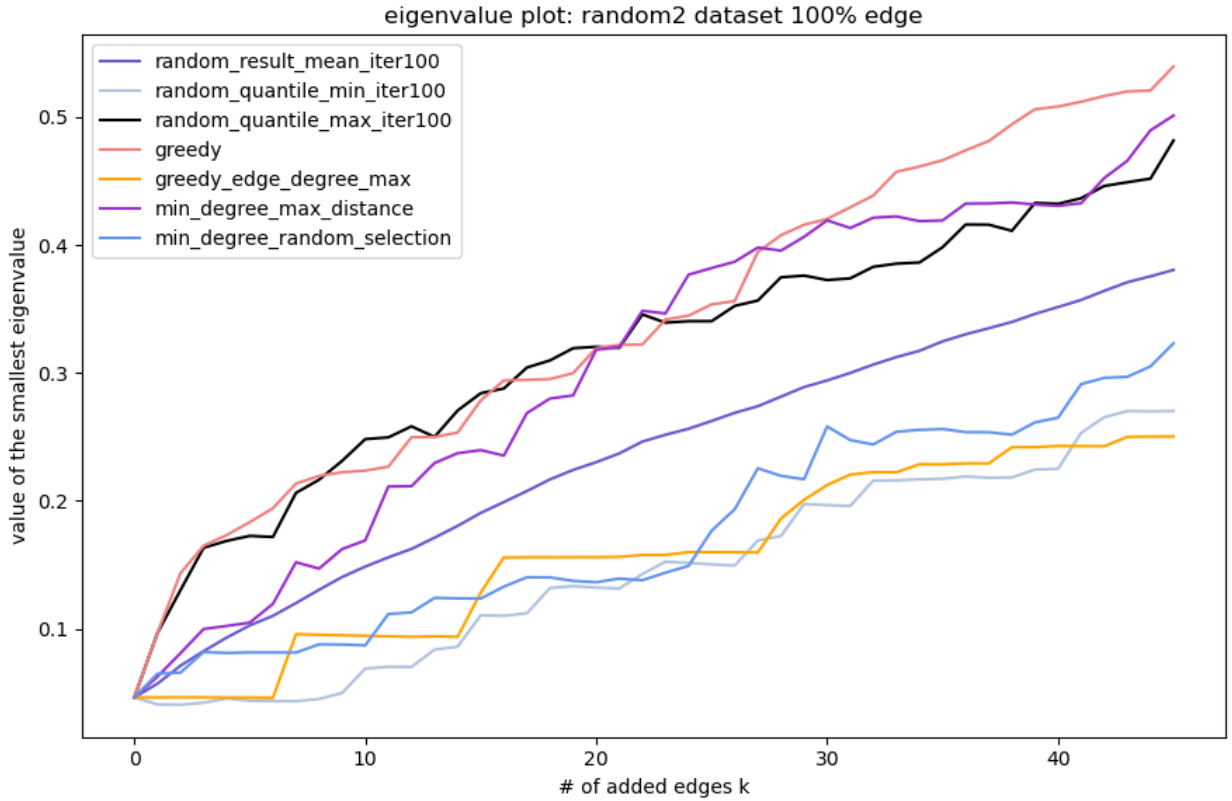
Algorithm	Random Graph	Cora
Local Greedy Search	99.53*s*	*N/A*
Random Selection	10.46*s*	1138.68*s*
Edge Rank	2.08*s*	375.44*s*
Smallest Degree	2.05*s*	45.80*s*

Algorithm	Random2	Random3	Random4
Local Greedy Search	11.66*s*	470.125*s*	10152.21*s*
Random Selection	4.92*s*	8.48*s*	33.89*s*
Edge Rank	2.06*s*	2.25*s*	3.05*s*
Smallest Degree	1.98*s*	2.18*s*	2.44*s*



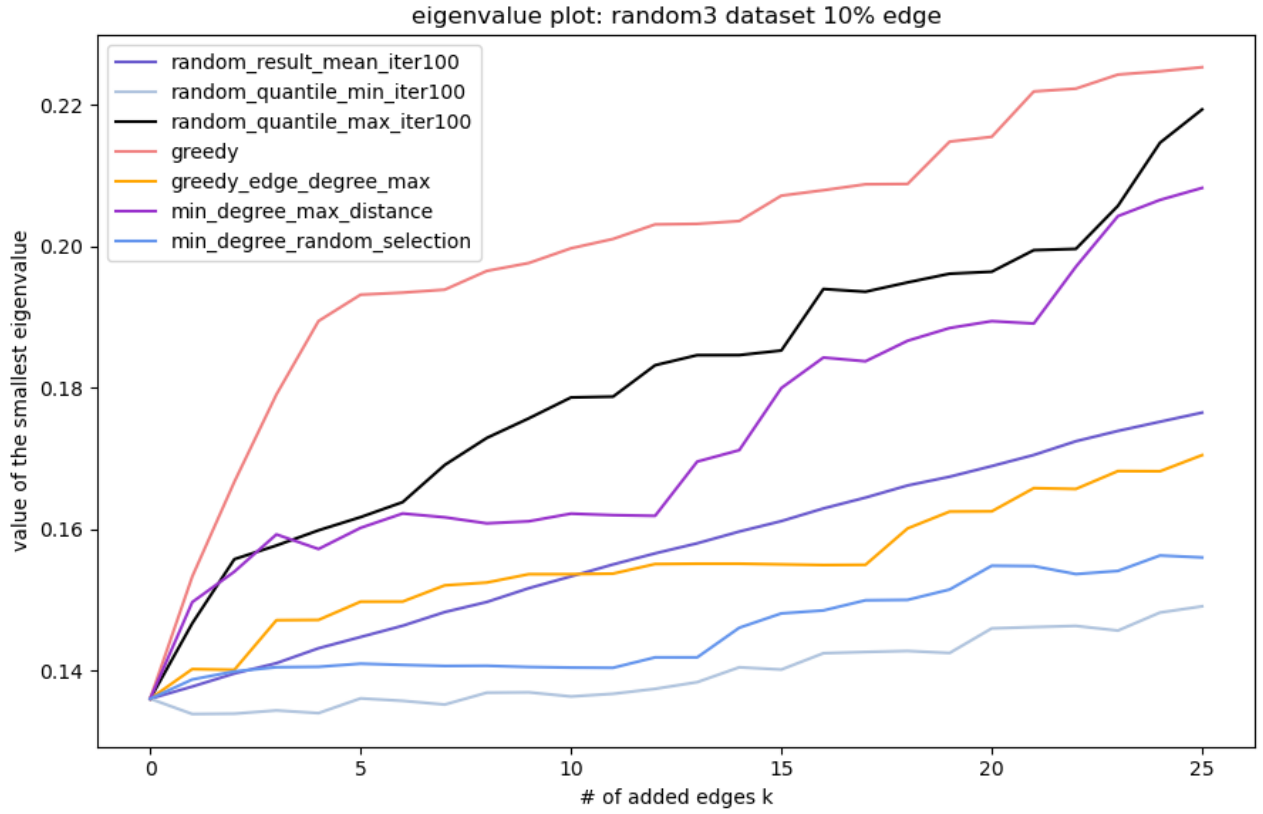
Performance Comparison: Random data

Iter_num = 100
Gamma = 1



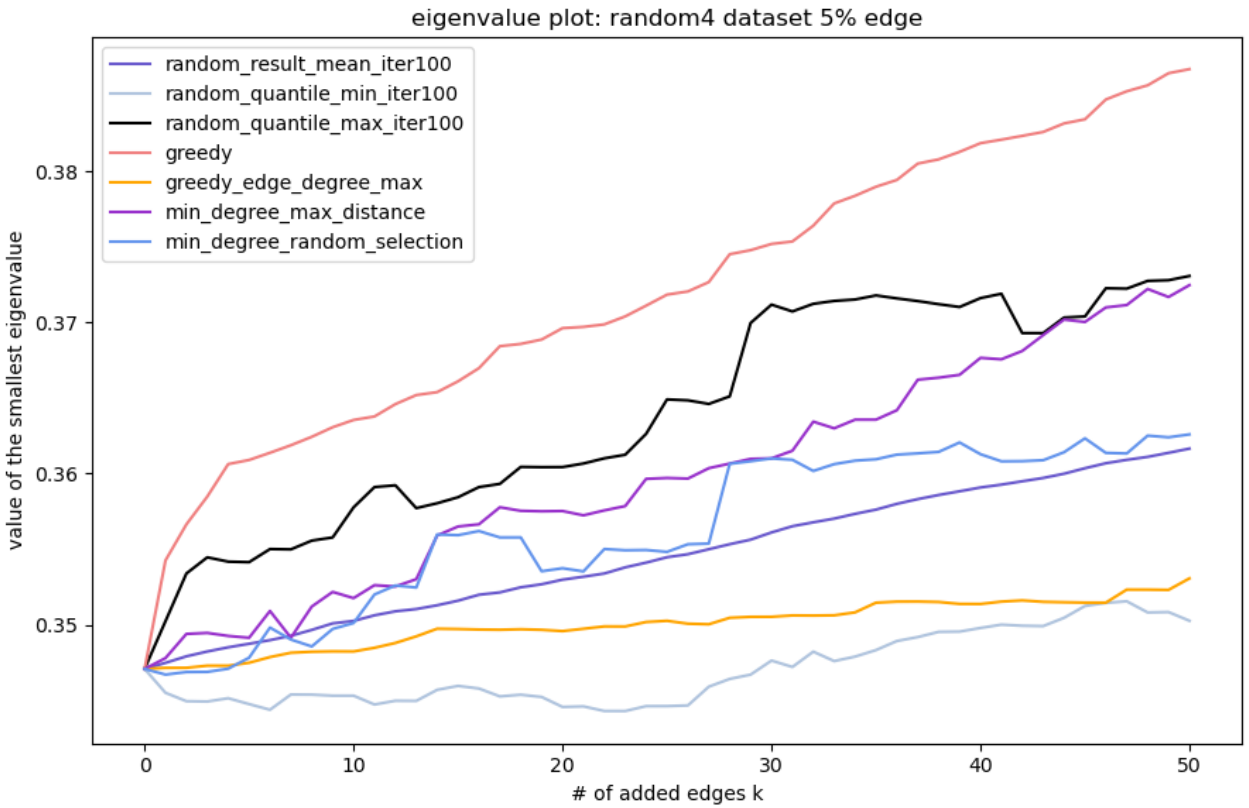
Performance Comparison: Random data 2

Iter_num = 100
Gamma = 1



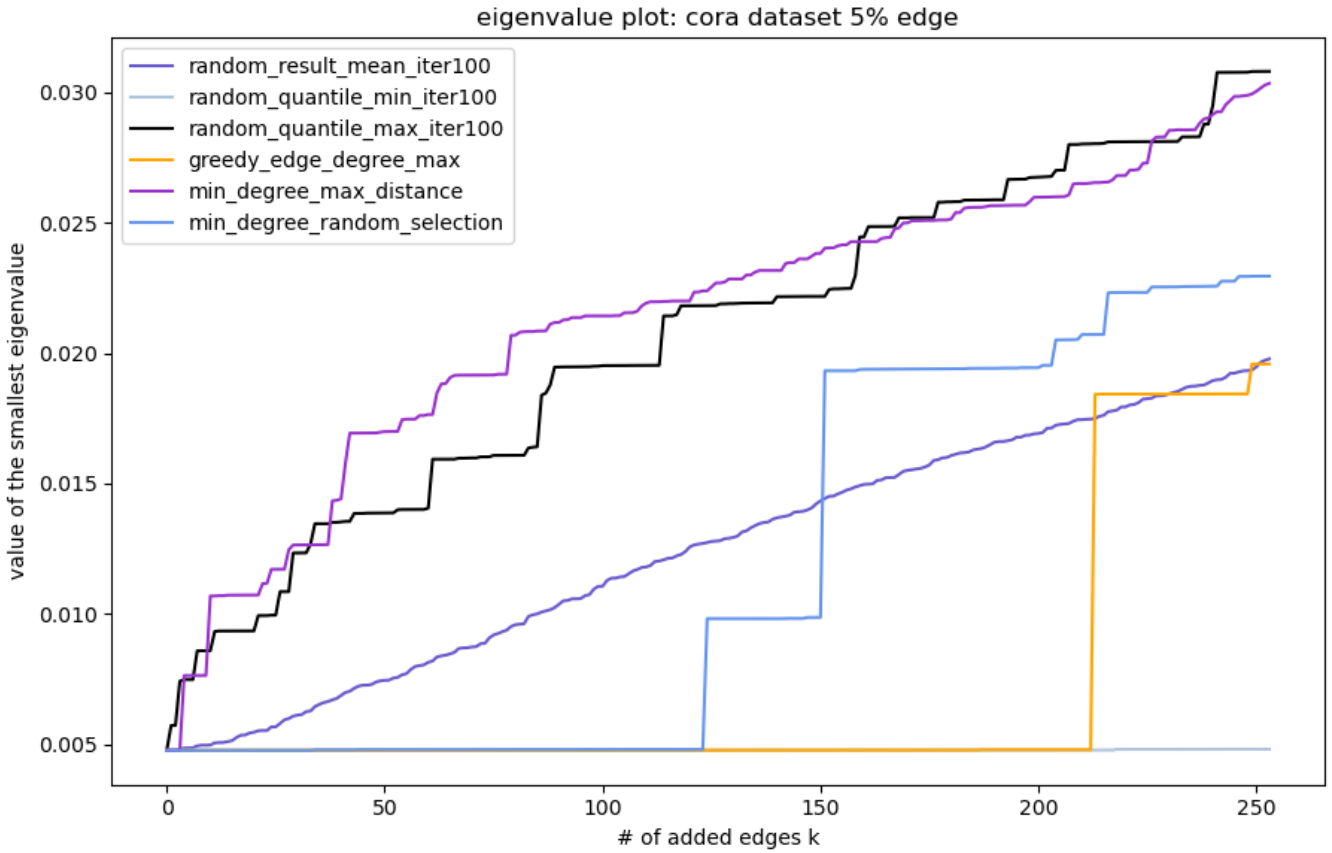
Performance Comparison: Random data 3

Iter_num = 100
Gamma = 0.1



Performance Comparison: Random data 4

Iter_num = 100
Gamma = 0.05



Performance Comparison: CORA

Iter_num = 500
Gamma = 0.05

Result Analysis

1. Degree based algorithms is most efficient in terms of time complexity

2. Random selection gives a tradeoff between effectiveness and efficiency, where increasing the iteration number can help to extremize the max / min bound, while the computation time might be longer

3. For small proportion of added edges, the local greedy selection gives the best performance. But the corresponding computation time grows non-linearly as the graph size increases.

4. The Min Degree Max Distance algorithm performs better than the average of random selection, and sometimes better than the local greedy selection, balancing efficiency and effectiveness.

Future Improvement

- The minimum degree strategy does not give best performance, which might be due to the degree normalization property for normalized Laplacian Matrix
- More effective method with high time efficiency should be considered
- Mathematical proof: try to form upper and lower bounds for the defined optimization problem
- May incorporate properties of corresponding eigenvector

Q&A

THANK YOU!

References

[1] D. P. Williamson, “Spectral graph theory,” https://people.orie.cornell.edu/dpw/orie6334/Fall2016/, 2016.

[2] A. Ghosh and S. Boyd, “Growing well-connected graphs,” in Proceedings of the 45th IEEE Conference on Decision and Control. IEEE, 2006, pp. 6605–6611.

[3] G. Li, Z. F. Hao, H. Huang, and H. Wei, “Maximizing algebraic connectivity via minimum degree and maximum distance,” IEEE Access, vol. 6, pp. 41 249–41 255, 2018.

Appendix: Proof of Theorem 3.1

Proof: To prove this theorem, we can consider the subspace containing all vectors orthogonal to the span of $x_1, x_2, \cdots, x_k$. Then $x_{k+1}, \cdots, x_n$ will be the basis of this subspace, which means any candidate $x \perp \text{span}(x_1, \cdots, x_k)$ can be expressed as $x = \alpha_{k+1}x_{k+1} + \cdots + \alpha_n x_n$. Therefore,

$$
\begin{aligned}
\frac{x^T A x}{x^T x} &= \frac{x^T \left(\alpha_{k+1}\lambda_{k+1}x_{k+1} + \cdots + \alpha_n \lambda_n x_n\right)}{x^T x} \\
&= \frac{\left(\alpha_{k+1}x_{k+1} + \cdots + \alpha_n x_n\right)^T \left(\alpha_{k+1}\lambda_{k+1}x_{k+1} + \cdots + \alpha_n \lambda_n x_n\right)}{\left(\alpha_{k+1}x_{k+1} + \cdots + \alpha_n x_n\right)^T \left(\alpha_{k+1}x_{k+1} + \cdots + \alpha_n x_n\right)} \\
&= \frac{\alpha_{k+1}^2\lambda_{k+1} + \cdots + \alpha_n^2\lambda_n}{\alpha_{k+1}^2 + \cdots + \alpha_n^2} \\
&\geq \lambda_{k+1}\frac{\alpha_{k+1}^2 + \cdots + \alpha_n^2}{\alpha_{k+1}^2 + \cdots + \alpha_n^2} \\
&= \lambda_{k+1}.
\end{aligned}
$$

Appendix: Proof of Theorem 3.3

Proof: To prove this, we first prove that a disconnected graph will lead λ_2 to be zero. Let G_1 and G_2 be a partition of the disconnected graph G such that there are no edges in between. In this case, by reindexing the nodes, the Laplacian matrix can be rewritten as

$$L_G = \begin{bmatrix} L_{G_1} & 0 \\ 0 & L_{G_2} \end{bmatrix}.$$

Given this, we can easily identify two vectors $[\mathbf{0}\ \mathbf{1}]$ and $[\mathbf{1}\ \mathbf{0}]$ of length n orthogonal to each other, and both of them are eigenvectors of L_G with associated eigenvalue being zero, which implies that $\lambda_2 = 0$.

Appendix: Proof of Theorem 3.3 (2)

To prove the reversed statement, let x_2 be an eigenvector of eigenvalue λ_2 and assume x_2 is orthogonal to $x_1 = e$. If $\lambda_2 = 0$, we have $x_2^T G x_2 = x_2^T (\lambda_2 x_2) = 0$. Therefore,

$$x_2^T L_G x_2 = \sum_{(i,j)\in E} (x_2(i) - x_2(j))^2 = 0,$$

which means each of the above squared items is equal to zero. Then we have $x_2(i) = x_2(j)$ for all $(i, j) \in E$. Consider two node sets $V_1 = \{i \in V : x_2(i) \geq 0\}$ and $V_2 = \{i \in V : x_2(i) < 0\}$. It's clear there are no edges between V_1 and V_2. Moreover, since we first assumed x_2 is orthogonal to vector e, we have $\langle x_2, e\rangle = 0$. This means there should be both positive and negative entries in x_2 which gives that $V_1 \neq \emptyset$ and $V_2 \neq \emptyset$. Hence G is disconnected.

Appendix: Eigenvalue Interlacing Problem

C. Eigenvalue Interlacing Theorem

The eigenvalue interlacing theorem states the eigenvalue relationships between a symmetric matrix and its principal submatrix. By proving this theorem, we can better understand the power of eigenvalues and eigenvectors in graph problems. Note that the theorem also serves as the foundation of many research problems stated in section 2.

Theorem 3.2: Let $A \in \mathbb{R}^{n\times n}$ be symmetric. $B \in \mathbb{R}^{m\times m}$ with $m < n$ is a principal submatrix of A. (Here the principal submatrix is obtained by deleting both i-th row and i-th column for some values of i). Suppose A has eigenvalues $\lambda_1 \le \cdots \le \lambda_n$ and B has eigenvalues $\beta_1 \le \cdots \le \beta_m$. Then

$$\lambda_k \le \beta_k \le \lambda_{k+n-m} \quad \text{for} \quad k = 1, \cdots, m.$$

Typically, if $m = n = 1$,

$$\lambda_1 \le \beta_1 \le \lambda_2 \le \beta_2 \le \cdots \le \beta_{n-1} \le \lambda_n.$$

Proof: Without loss of generality, we assume $A = \begin{bmatrix} B & X^T \\ X & Z \end{bmatrix}$, $\{x_1, \cdots, x_n\}$ are eigenvectors of A, and $\{y_1, \cdots, y_m\}$ are eigenvectors of B. To prove the theorem, define vector spaces

$$V = \text{span}\,(x_k, \cdots, x_n), \quad W = \text{span}\,(y_1 \cdots, y_k),$$

$$\widetilde{W} = \left\{ \begin{pmatrix} w \\ 0 \end{pmatrix} \in \mathbb{R}^n, w \in W \right\}.$$

There exists $\widetilde{w} \in V \cap \widetilde{W}$ and $\widetilde{w} = \begin{pmatrix} w \\ 0 \end{pmatrix}$ for some $w \in W$. Therefore,

$$\widetilde{w}^T A \widetilde{w} = \begin{bmatrix} w^T & 0 \end{bmatrix} \begin{bmatrix} B & X^T \\ X & Z \end{bmatrix} \begin{bmatrix} w \\ 0 \end{bmatrix} = w^T B w.$$

From (2) we have $\lambda_k = \min_{x\in V} \frac{x^T A x}{x^T x}$ and $\beta_k = \max_{x\in W} \frac{x^T B x}{x^T x}$. Then we get

$$\lambda_k \le \frac{\widetilde{w}^T A \widetilde{w}}{\widetilde{w}^T \widetilde{w}} = \frac{w^T B w}{w^T w} \le \beta_k.$$

For proof of the other side of the inequality, the construction can be similar. We skip the part at this moment.

Appendix: Linear Embedding

Another interpretation of the perturbation heuristic is via linear embedding. The problem of embedding a graph $G = (V, E)$ into the real line is the following: assign node coordinates $x_1, \ldots, x_n$, with 0 mean and unit variance, so as to minimize the sum of squares of distances between adjacent nodes. That is, the real embedding problem is

$$\begin{array}{ll} \text{minimize} & \sum_{(i,j)\in E}(x_i - x_j)^2 \\ \text{subject to} & \mathbf{1}^T x = 0, \\ & x^T x = 1. \end{array} \tag{13}$$

The unit variance constraint is imposed since otherwise the problem is trivial with optimal solution $x_i = 0$, and optimal value 0.

The objective in (13) is simply $x^T L x$, and so the optimal value is $\lambda_2(L)$, with optimal solution $x = v$, where v is the Fiedler vector. The perturbation heuristic therefore adds an edge between the nodes that are farthest from each other in the linear embedding.

Appendix: Computation Time