Geometric Aspects of Mining Complex Networks

Leo Torres
PhD candidate
Network Science Institute, Northeastern University
Geometric Aspects of Mining Complex Networks
What concepts and procedures can we take from geometry and topology and apply to mining and learning from complex networks?
Distances

\[ d \left( \begin{array}{c}
\text{A} \\
\text{B}
\end{array} \right) \]
Embeddings
Distances

$$d\left(\begin{array}{c} A \end{array} , \begin{array}{c} B \end{array}\right)$$

Non-backtracking cycles: length spectrum theory and graph mining applications


Embeddings

GLEE: Geometric Laplacian Eigenmap Embedding

NBD: Non-Backtracking Distance


Supported by NSF CNS-1314603, NSF IIS-1741197, and DTRA HDTRA1-10-1-0120.
**Spoiler Alert!**

**Networks:** the non-backtracking eigenvalues track important descriptors like degree distribution and triangles.

Machine Learning: non-backtracking eigenvalues are a great way of measuring distance.

Mathematics: the length spectrum of an unweighted graph characterizes its 2-core uniquely up to isomorphism.
Networks: the non-backtracking eigenvalues track important descriptors like degree distribution and triangles.

**Machine Learning:** non-backtracking eigenvalues are a great way of measuring distance.

Mathematics: the length spectrum of an unweighted graph characterizes its 2-core uniquely up to isomorphism.
Spoiler Alert!

Networks: the non-backtracking eigenvalues track important descriptors like degree distribution and triangles.

Machine Learning: non-backtracking eigenvalues are a great way of measuring distance.

Mathematics: the length spectrum of an unweighted graph characterizes its 2-core uniquely up to isomorphism.
The Length Spectrum

1. Given a graph $G = (V, E)$ and a node $v$, 

![Graph Diagram]
The Length Spectrum

1. Given a graph $G = (V, E)$ and a node $v$, consider the set of all closed walks that start and end at $v$. 

![Diagram showing a graph $G$ with a highlighted node $v$ and various closed walks]
The Length Spectrum

2. **Walks are equivalent** if they are equal save for **tree-like parts that don’t go through the basepoint**...
The Length Spectrum

2. ... and retain the shortest walk in each subset.
2. This set is the fundamental group of $G$ with basepoint $v$. 

$\pi_1(G, v)$
3. **Walks are equivalent** if they are equal *save for tree-like parts* that don’t go through the basepoint.

\[ \pi_1(G, v) \]
The Length Spectrum

3. This is the set of non-backtracking cycles (NBCs) of G.
The Length Spectrum

4. $\mathcal{L}$ is defined on $\pi_1(G, v)$ and assigns each walk the length of its “shaved” version.
The Length Spectrum

4. $\mathcal{L}$ is defined on $\pi_1(G, v)$ and assigns each walk the length of its “shaved” version.

\[ \mathcal{L} : \pi_1(G, v) \rightarrow \mathbb{R} \]

\[ \mathcal{L}(\text{walk}) = 3 \]
The Length Spectrum

The Length Spectrum of a graph characterizes its 2-core uniquely up to isomorphism.

Modifying the Length Spectrum

$$d(G, H) = d(\mathcal{L}_G, \mathcal{L}_H)$$
Modifying the Length Spectrum

\[ d(G, H) = d(\mathcal{L}_G, \mathcal{L}_H) \]

<table>
<thead>
<tr>
<th>Two assumptions</th>
<th>Two problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G \rightarrow \mathcal{L}_G )</td>
<td>How to compute?</td>
</tr>
<tr>
<td>( d(\mathcal{L}_G, \mathcal{L}_H) )</td>
<td>How to compare?</td>
</tr>
</tbody>
</table>
Modifying the Length Spectrum

\[ d(G, H) = d(\mathcal{L}_G, \mathcal{L}_H) \]

<table>
<thead>
<tr>
<th>Two assumptions</th>
<th>Two problems</th>
<th>Two solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G \rightarrow \mathcal{L}_G )</td>
<td>How to compute?</td>
<td>Outputs instead of inputs</td>
</tr>
<tr>
<td>( d(\mathcal{L}_G, \mathcal{L}_H) )</td>
<td>How to compare?</td>
<td>Partition the set of outputs</td>
</tr>
</tbody>
</table>
## Modifying the Length Spectrum

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Graphs G" /></td>
<td><img src="image" alt="Graphs H" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Graphs G" /></td>
<td><img src="image" alt="Graphs H" /></td>
</tr>
</tbody>
</table>
## Modifying the Length Spectrum

### Inputs

<table>
<thead>
<tr>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Graph G" /></td>
<td><img src="image2" alt="Graph H" /></td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>count</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>count</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td></td>
</tr>
</tbody>
</table>
# Modifying the Length Spectrum

## Inputs

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image1" alt="Graph G" /></td>
<td><img src="image2" alt="Graph H" /></td>
</tr>
</tbody>
</table>

## Outputs

<table>
<thead>
<tr>
<th></th>
<th>Count of Length 3</th>
<th>Count of Length 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td><img src="chart1" alt="Bar Chart 3" /></td>
<td><img src="chart2" alt="Bar Chart 3" /></td>
</tr>
<tr>
<td>4</td>
<td><img src="chart3" alt="Bar Chart 4" /></td>
<td><img src="chart4" alt="Bar Chart 4" /></td>
</tr>
</tbody>
</table>
Modifying the Length Spectrum

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Graph G]</td>
<td>![Graph H]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>length</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>length</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>
Modifying the Length Spectrum

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>H</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>length</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>length</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

...
Non-backtracking matrix

\[ G = (V, E) \]

\[ |E| = m \]
Graph Distance

\[ tr(B^3) \quad tr(B^5) \quad tr(B^4) \]
Given two graphs $G, H$ and an integer $r$, write $\lambda_k, \mu_k$ for the eigenvalues of their corresponding non-backtracking matrices, $k = 1, 2, \ldots, r$. Let $\Lambda$ and $M$ be the cumulative density function of the respective spectral densities. Define the distance between $G$ and $H$ by

$$d_r(G, H) = \sqrt{\iint |\Lambda(x, y) - M(x, y)|^2 \, dx \, dy}$$
Properties: hubs

Configuration model \((n = 10k, \langle k \rangle = 10, \gamma = 2.1)\)

fewer hubs
Properties: hubs are imaginary

Configuration model ($n = 10k, \langle k \rangle = 10, \gamma = 2.1$)

fewer hubs
Properties: triangles

ER graph (n = 10k, p = 0.001)

more triangles
Properties: triangles

ER graph (n = 10k, p = 0.001)

more triangles
Examples: clustering

1 dot = 1 eigenvalue

1 dot = 1 graph
Examples: clustering

graphs showing the relationship between completeness and homogeneity for different datasets and methods.
Summary

The non-backtracking eigenvalues track descriptors like degree distribution and triangles and can find patterns and anomalies; THEREFORE they are a great way of measuring distance BECAUSE they contain similar information to the length spectrum, which characterizes the 2-core of an unweighted graph uniquely.

Supported by NSF CNS-1314603, NSF IIS-1741197, and DTRA HDTRA1-10-1-0120.
Summary: geometry

- The derivation and algorithm are based on **algebraic topology**.
  - Intrinsic topology/geometry of each graph.

- The set of eigenvalues can be considered as a form of “**graph embedding**”.
  - Geometry of the set of all graphs, as represented by their eigenvalues.

Supported by **NSF CNS-1314603**, **NSF IIS-1741197**, and **DTRA HDTRA1-10-1-0120**.
GLEE: Geometric Laplacian Eigenmap Embedding


Work supported by NSF CNS-1314603, NSF IIS-1741197, Army Research Laboratory Cooperative Agreement W911NF-13-2-0045.
Spoiler Alert!

Mathematics: there is a bijection between undirected graphs on n nodes and n-1 dimensional simplices.

Networks: we can encode graph structure in geometric terms using the simplex geometry of the Laplacian.

Machine Learning: some embedding methods perform well only when clustering coefficient is high.
**Spoiler Alert!**

Mathematics: there is a bijection between undirected graphs on n nodes and n-1 dimensional simplices.

**Networks:** we can **encode graph structure in geometric terms** using the simplex geometry of the Laplacian.

Machine Learning: some embedding methods perform well only when clustering coefficient is high.
**Spoiler Alert!**

Mathematics: there is a bijection between undirected graphs on n nodes and n-1 dimensional simplices.

Networks: we can encode graph structure in geometric terms using the simplex geometry of the Laplacian.

**Machine Learning:** some embedding methods perform well only when clustering coefficient is high.
GLEE: Geometric Laplacian Eigenmap Embedding

\[ L = P \times \Lambda \times P^T \]
Singular Value Decomposition says that eliminating the rows and columns corresponding to the lowest singular values give a good approximation of $L$. 
However, the last eigenvalue of $L$ is always 0, which implies exact equality.
GLEE: Geometric Laplacian Eigenmap Embedding

\[
L \times n \times n = P \times (n - 1) \times (n - 1) \times P^T
\]
GLEE: Geometric Laplacian Eigenmap Embedding

\[ n \times n \quad \quad = \quad \quad n \times n - 1 \quad \quad \times \quad \quad n - 1 \times n - 1 \quad \quad \times \quad \quad n - 1 \times n \]

\[ L \quad \quad P \quad \quad \sqrt{\Lambda} \quad \sqrt{\Lambda} \quad \quad P^T \]
GLEE: Geometric Laplacian Eigenmap Embedding

\[
\begin{align*}
L_{n \times n} & \quad = \quad S_{n \times n-1} \quad \times \quad S^T_{n-1 \times n} \\
S & = P\sqrt{\Lambda}
\end{align*}
\]
In a connected graph, $L$ has rank $n-1$, and only one eigenvalue equal to 0. This implies that $S$ has full rank, i.e., rank $n-1$. 

GLEE: Geometric Laplacian Eigenmap Embedding
This implies that the rows of $S$ point to the vertices of an $(n-1)$-D simplex.
GLEE: Geometric Laplacian Eigenmap Embedding

Given a graph $G = (V, E)$, define the $d$-dimensional GLEE of a node $i$ as the first $d$ columns of the $i$-th row of $S = P \Lambda^{1/2}$, and is denote it by $s_i$. 
Graph Reconstruction

Given the matrix $S$ whose rows are $s_i$, how do we reconstruct the graph?
**Graph Reconstruction**

Given the matrix $S$ whose rows are $s_i$, how do we reconstruct the graph?

- Assume $d = n-1$. In this case, we simply have $L = SS^T$. 

![Graph Reconstruction Diagram](image-url)
Graph Reconstruction

Given the matrix $S$ whose rows are $s_i$, how do we reconstruct the graph?

- Assume $d = n-1$. In this case, we simply have $L = S S^T$.
- If $d < n$, then $S S^T$ is the best rank-$d$ approximation of $L$. 

![Graph Reconstruction Diagram]
Graph Reconstruction: results

Same embedding dimension, similar network size, but different average clustering.
Link Prediction: common neighbors

In many networks (e.g. social networks), the number of common neighbors is an excellent predictor of links because of triadic closure.
Link Prediction: common neighbors

In many networks (e.g. social networks), the number of common neighbors is an excellent predictor of links because of **triadic closure**.
Link Prediction: common neighbors

In many networks (e.g. social networks), the number of common neighbors is an excellent predictor of links because of **triadic closure**.
Link Prediction: common neighbors

In many networks (e.g. social networks), the number of common neighbors is an excellent predictor of links because of **triadic closure**.
Link Prediction: 3-paths

In other networks with low clustering (e.g. PPI networks), a better predictor is the number of paths of length 3.
**Link prediction: results**

Average clustering coefficient

- 0.04
- 0.14
- 0.56

**Graphs:**
- **PPI**
- **Wiki-Vote**
- **CA-GrQc**

AUC vs. Dimension $d$

- LE
- node2vec
- GLEE
- GLEE-L3
Summary

There is a bijection between undirected graphs and simplices, THEREFORE we can encode graph structure in geometric terms using GLEE. IN CONTRAST, other methods usually make assumptions about the structure of the graph and therefore perform well only when those assumptions hold (e.g. high clustering coefficient).
Summary: Geometry

- What else can we do with the geometry of embeddings?
- How can graphs be encoded geometrically?
Perturbations
Summary

The largest eigenvalue behaves in predictable ways. **THEREFORE,** monitoring it should provide a good **defense against adversarial attack.** **FOR EXAMPLE,** to immunize against certain recurrent state dynamics, first **remove hubs,** then **break up the cliques.**
Perturbations

Node Immunization with Non-backtracking Eigenvalues


Optimizing Graph Structure for Targeted Diffusion

Geometry...?

Work supported by NSF CNS-1314603, NSF IIS-1741197, Army Research Laboratory Cooperative Agreement W911NF-13-2-0045.
Gracias!

Currently on the job market as a postdoc or assistant professor at the intersection of network science, computer science, and mathematics. Please get in touch!

leo@leotrs.com
www.leotrs.com
@_leotrs
/leotrs