Community Detection on Networks with Ricci Flow

Chien-Chun Ni, Yu-Yao Lin, Feng Luo & Jie Gao
(Scientific Reports 2019)

Presenter: Yue Xiang

2022.4.7

中國人民大學
RENMIN UNIVERSITY OF CHINA

Outlines

Introduction

- Many complex networks in the real world have community structures.

Introduction

- Many complex networks in the real world have community structures.
  - Nodes in the same community are densely connected while nodes from different communities are sparsely connected.

Introduction

- Many complex networks in the real world have community structures.
  - Nodes in the same community are densely connected while nodes from different communities are sparsely connected.
- Recognition of community structures plays an important role in supporting processes on networks, *e.g.*, spread of diseases, information or behaviors.

Introduction

- Many complex networks in the real world have community structures.
  - Nodes in the same community are densely connected while nodes from different communities are sparsely connected.
- Recognition of community structures plays an important role in supporting processes on networks, *e.g.*, spread of diseases, information or behaviors.

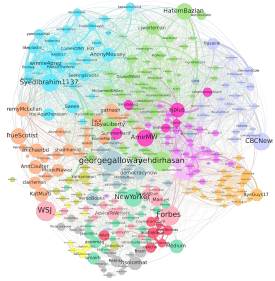

Figure 1: Twitter EGO Network

Current Work on Community Detection

Most current work on community detection try to recognize dense clusters in a graph by

- randomized algorithms, *e.g.* label propagation or random walks;
- optimized centrality, *e.g.*, betweenness centrality; etc.

This work introduces a discrete Ricci flow on weighted graphs (networks) for identifying communities in a network.

The Optimal Transportation

The goal is to find a transportation plan γ that attains the infimum cost

$$W(\mu,\nu) = \inf\left\{ \int_{X\times Y} c(x,y)\mathrm{d}\gamma(x,y) \mid \gamma \in \Gamma(\mu,\nu) \right\}$$

The Optimal Transportation

The goal is to find a transportation plan γ that attains the infimum cost

$$W(\mu,\nu) = \inf\left\{ \int_{X\times Y} c(x,y)\mathrm{d}\gamma(x,y) \mid \gamma \in \Gamma(\mu,\nu) \right\}$$

The Optimal Transportation

The goal is to find a transportation plan γ that attains the infimum cost

$$W(\mu,\nu) = \inf\left\{\int_{X\times Y} c(x,y)\mathrm{d}\gamma(x,y) \mid \gamma \in \Gamma(\mu,\nu)\right\}$$

- The cost function c is usually taken to be the distance $d(x,y)$ if $X = Y$ and the cost of transportation per-unit distance is constant.

The Optimal Transportation

The goal is to find a transportation plan γ that attains the infimum cost

$$W(\mu,\nu) = \inf\left\{ \int_{X\times Y} c(x,y)\mathrm{d}\gamma(x,y) \mid \gamma \in \Gamma(\mu,\nu) \right\}$$

- The cost function c is usually taken to be the distance $d(x,y)$ if $X = Y$ and the cost of transportation per-unit distance is constant.
- γ is a probability measure on $X \times Y$ satisfying $\gamma(A \times Y) = \mu(A)$ and $\gamma(X \times B) = \nu(B)$ for all measurable subsets A and B.

The Optimal Transportation

The goal is to find a transportation plan γ that attains the infimum cost

$$W(\mu,\nu) = \inf\left\{ \int_{X\times Y} c(x,y)\mathrm{d}\gamma(x,y) \mid \gamma \in \Gamma(\mu,\nu) \right\}$$

- The cost function c is usually taken to be the distance $d(x,y)$ if $X = Y$ and the cost of transportation per-unit distance is constant.
- γ is a probability measure on $X \times Y$ satisfying $\gamma(A \times Y) = \mu(A)$ and $\gamma(X \times B) = \nu(B)$ for all measurable subsets A and B.
- $\Gamma(\mu,\nu)$ denotes the collection of all possible transportation plans.

The Optimal Transportation

The goal is to find a transportation plan γ that attains the infimum cost

$$W(\mu,\nu) = \inf\left\{ \int_{X\times Y} c(x,y)\mathrm{d}\gamma(x,y) \mid \gamma \in \Gamma(\mu,\nu) \right\}$$

- The cost function c is usually taken to be the distance $d(x,y)$ if $X = Y$ and the cost of transportation per-unit distance is constant.
- γ is a probability measure on $X \times Y$ satisfying $\gamma(A \times Y) = \mu(A)$ and $\gamma(X \times B) = \nu(B)$ for all measurable subsets A and B.
- $\Gamma(\mu,\nu)$ denotes the collection of all possible transportation plans.
- $W(\mu,\nu)$ is called the Wasserstein distance between two probability measures μ,ν on X.

An Intuitive Understanding of Curvature

Curvature quantitatively measures how space is curved: the amount by which a curve deviates from being a straight line, or a surface deviates from being a plane.

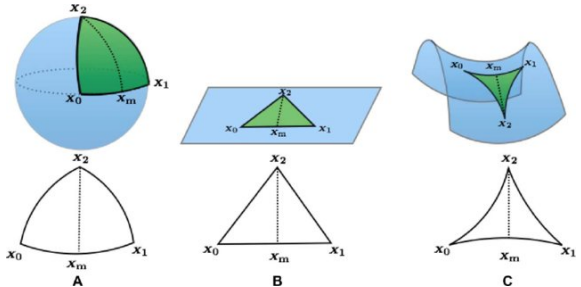

Figure 2: Geodesic triangles on different surfaces: (A) sphere: positive curvature, (B) planar surface: zero, and (C) hyperbolic paraboloid of one sheet: negative

Connecting Curvature to Community Detection

- A positive sectional curvature space tends to have a small diameter and is geometrically crowded (*e.g.*, a sphere).
- A negative sectional curvature closed Riemannian manifold is geometrically spreading out like a tree in large scale.

Connecting Curvature to Community Detection

- A positive sectional curvature space tends to have a small diameter and is geometrically crowded (*e.g.*, a sphere).
- A negative sectional curvature closed Riemannian manifold is geometrically spreading out like a tree in large scale.

Thus, a positively curved region behaves more like a 'community' than negatively curved regions.

Ricci Flow

Given a smooth manifold M and an open real interval (a, b), a Ricci flow assigns to each t in the interval (a, b) a Riemannian metric g_t on M such that

$$\frac{\partial}{\partial t} g_t = -2R^{g_t}.$$

R^{g_t}: Ricci tensor (Ricci curvature).
The Ricci tensor is often thought of as an average value of the sectional curvatures.

Ricci Flow

Given a smooth manifold M and an open real interval (a, b), a Ricci flow assigns to each t in the interval (a, b) a Riemannian metric g_t on M such that

$$\frac{\partial}{\partial t} g_t = -2R^{g_t}.$$

R^{g_t}: Ricci tensor (Ricci curvature).
The Ricci tensor is often thought of as an average value of the sectional curvatures.

- If R^{g_t} is positive, the Ricci flow "contracts".
- If R^{g_t} is negative, the Ricci flow "expands".

Ollivier's Ricci Curvature

Let (M^n, d) be an n-dimensional Riemannian manifold with Riemannian distance d whose Ricci curvature is k and Riemannian volume measure is μ.

Ollivier's Ricci Curvature

Let (M^n, d) be an n-dimensional Riemannian manifold with Riemannian distance d whose Ricci curvature is k and Riemannian volume measure is μ. Fix $\varepsilon > 0$. Let $m_X = \frac{\mu|_{B(x,\varepsilon)}}{\mu(B(x,\varepsilon))}$ be the **probability measure** associated to $x \in M$.

Ollivier's Ricci Curvature

Given a metric space (X, d) equipped with a probability measure m_x for each $x \in X$, the Ollivier's Ricci curvature along the shortest path xy is

$$k(x, y) = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

where $W(m_x, m_y)$ is the Wasserstein distance with respect to the cost function $c(x, y) = d(x, y)$.

Discrete Ollivier Ricci Curvature

- For each node x on a metric graph $G = (V, E, w)$, we define a **mass distribution** m_x on x's neighbor nodes.

Discrete Ollivier Ricci Curvature

- For each node x on a metric graph $G = (V, E, w)$, we define a **mass distribution** m_x on x's neighbor nodes.
- A **discrete transport plan** is a map $A : V \times V \to [0, 1]$ such that $A(u, v)$ is the amount of mass at vertex v to be moved to vertex u. It satisfies $\sum_{v' \in V} A(u, v') = m_x(u)$ and $\sum_{u' \in V} A(u', v) = m_y(v)$.

Discrete Ollivier Ricci Curvature

- For each node x on a metric graph $G = (V, E, w)$, we define a **mass distribution** m_x on x's neighbor nodes.
- A **discrete transport plan** is a map $A : V \times V \to [0, 1]$ such that $A(u, v)$ is the amount of mass at vertex v to be moved to vertex u. It satisfies $\sum_{v' \in V} A(u, v') = m_x(u)$ and $\sum_{u' \in V} A(u', v) = m_y(v)$.
- $W(m_x, m_y)$ is the minimum total weighted travel distance to move m_x to m_y, i.e., $W(m_x, m_y) = \inf \left\{ \sum_{u,v \in V} A(u, v) d(u, v) \right\}$.

Discrete Ollivier Ricci Curvature

- For each node x on a metric graph $G = (V, E, w)$, we define a **mass distribution** m_x on x's neighbor nodes.
- A **discrete transport plan** is a map $A : V \times V \to [0, 1]$ such that $A(u, v)$ is the amount of mass at vertex v to be moved to vertex u. It satisfies $\sum_{v' \in V} A(u, v') = m_x(u)$ and $\sum_{u' \in V} A(u', v) = m_y(v)$.
- $W(m_x, m_y)$ is the minimum total weighted travel distance to move m_x to m_y, i.e., $W(m_x, m_y) = \inf \left\{ \sum_{u,v \in V} A(u, v) d(u, v) \right\}$.
- The discrete Ricci curvature on a network edge $xy \in E$ is defined as

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

where $d(x, y)$ is the length of the shortest path between nodes x and y.

Inter-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

Inter-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

If two nodes x and y are from different communities, their neighbor nodes tend to have fewer common neighbors.

Inter-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

If two nodes x and y are from different communities, their neighbor nodes tend to have fewer common neighbors.
$\Rightarrow$ The best way to move m_x from x's neighbors to m_y in y's neighbors is to travel along the edge xy.

Inter-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

If two nodes x and y are from different communities, their neighbor nodes tend to have fewer common neighbors.
$\Rightarrow$ The best way to move m_x from x's neighbors to m_y in y's neighbors is to travel along the edge xy.
$\Rightarrow$ The Wasserstein distance is necessarily larger than the length of xy.
$\Rightarrow$ Inter-community edges have **negative** Ricci curvature!

Intra-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

Intra-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

Nodes within the same community tend to share neighbors or have shortcut between neighbors.

Intra-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

Nodes within the same community tend to share neighbors or have shortcut between neighbors.
$\Rightarrow$ The Wasserstein distance is no greater than $d(x, y)$.

Intra-community Ricci Curvature

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)}$$

Nodes within the same community tend to share neighbors or have shortcut between neighbors.
$\Rightarrow$ The Wasserstein distance is no greater than $d(x, y)$.
$\Rightarrow$ Intra-community edges are mostly **positively** curved.

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $C = \sum_{x_i \in \pi(x)} \exp\left(-d(x,x_i)^p\right)$ is a normalization factor.

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $C = \sum_{x_i \in \pi(x)} \exp\left(-d(x,x_i)^p\right)$ is a normalization factor.
- $\pi(x)$ is the set of neighbors of x.

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $C = \sum_{x_i \in \pi(x)} \exp\left(-d(x,x_i)^p\right)$ is a normalization factor.
- $\pi(x)$ is the set of neighbors of x.
- The parameter α determines the probability to remain at x.

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $C = \sum_{x_i \in \pi(x)} \exp\left(-d(x,x_i)^p\right)$ is a normalization factor.
- $\pi(x)$ is the set of neighbors of x.
- The parameter α determines the probability to remain at x.
- The power parameter p determines how much we want to discount the neighbor x_i of x with respect to the weight $d(x,x_i)$:

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $C = \sum_{x_i \in \pi(x)} \exp\left(-d(x,x_i)^p\right)$ is a normalization factor.
- $\pi(x)$ is the set of neighbors of x.
- The parameter α determines the probability to remain at x.
- The power parameter p determines how much we want to discount the neighbor x_i of x with respect to the weight $d(x,x_i)$:
  - $p = 0$: the probability measure is uniform on all neighbors of x.

Probability Distribution $m_x^{\alpha,p}$

We suggest a more general family of probability distributions $m_x^{\alpha,p}$, with two parameters: $\alpha \in [0,1]$ and power $p \geq 0$:

$$m_x^{\alpha,p}(x_i) = \begin{cases} \alpha & \text{if } x_i = x \\ \frac{1-\alpha}{C} \cdot \exp\left(-d(x,x_i)^p\right) & \text{if } x_i \in \pi(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $C = \sum_{x_i \in \pi(x)} \exp\left(-d(x,x_i)^p\right)$ is a normalization factor.
- $\pi(x)$ is the set of neighbors of x.
- The parameter α determines the probability to remain at x.
- The power parameter p determines how much we want to discount the neighbor x_i of x with respect to the weight $d(x,x_i)$:
  - $p = 0$: the probability measure is uniform on all neighbors of x.
  - For a large p, the neighbors that are far away from x are aggressively discounted.

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.
- In each iteration, we update all edge weights simultaneously by the following flow process:

$$w_{xy}^{(i+1)} = d^{(i)}(x, y) - \epsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x, y)$$

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.
- In each iteration, we update all edge weights simultaneously by the following flow process:

$$w_{xy}^{(i+1)} = d^{(i)}(x, y) - \epsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x, y)$$

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.
- In each iteration, we update all edge weights simultaneously by the following flow process:

$$w_{xy}^{(i+1)} = d^{(i)}(x, y) - \epsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x, y)$$

  - Initially $w_{xy}^{(0)} = w_{xy}$ and $d_{xy}^{(0)} = d_{xy}$.

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.
- In each iteration, we update all edge weights simultaneously by the following flow process:

$$w_{xy}^{(i+1)} = d^{(i)}(x, y) - \epsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x, y)$$

  - Initially $w_{xy}^{(0)} = w_{xy}$ and $d_{xy}^{(0)} = d_{xy}$.
  - $w_{xy}^{(i)}$ is the weight of the edge xy at the i-th iteration.

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.
- In each iteration, we update all edge weights simultaneously by the following flow process:

$$w_{xy}^{(i+1)} = d^{(i)}(x,y) - \epsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x,y)$$

  - Initially $w_{xy}^{(0)} = w_{xy}$ and $d_{xy}^{(0)} = d_{xy}$.
  - $w_{xy}^{(i)}$ is the weight of the edge xy at the i-th iteration.
  - $\kappa_{xy}^{(i)}$ is the Ricci curvature at the edge xy at the i-th iteration.

Update Edge Weights via Discrete Ricci Flow

- The discrete Ricci flow algorithm on a network is an **evolving process**.
- In each iteration, we update all edge weights simultaneously by the following flow process:

$$w_{xy}^{(i+1)} = d^{(i)}(x, y) - \epsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x, y)$$

  - Initially $w_{xy}^{(0)} = w_{xy}$ and $d_{xy}^{(0)} = d_{xy}$.
  - $w_{xy}^{(i)}$ is the weight of the edge xy at the i-th iteration.
  - $\kappa_{xy}^{(i)}$ is the Ricci curvature at the edge xy at the i-th iteration.
  - $d^{(i)}(x, y)$ is the shortest path distance on the graph induced by the weights $w_{xy}^{(i)}$.

Network ’Surgery’

- The discrete Ricci flow process expands negatively curved edges and shrinks positively curved edges.

Network ’Surgery’

- The discrete Ricci flow process expands negatively curved edges and shrinks positively curved edges.
- Eventually, nodes connected by intra-community edges are condensed and inter-community edges are stretched.

Network 'Surgery'

- The discrete Ricci flow process expands negatively curved edges and shrinks positively curved edges.
- Eventually, nodes connected by intra-community edges are condensed and inter-community edges are stretched.
- A simple thresholding procedure can easily separate different communities: **network 'surgery'**:
  - remove edges of large weights (likely inter-community edges) after several Ricci flow iterations (usually 10 to 15 iterations).

Sketch of Ricci Flow for Community Detection

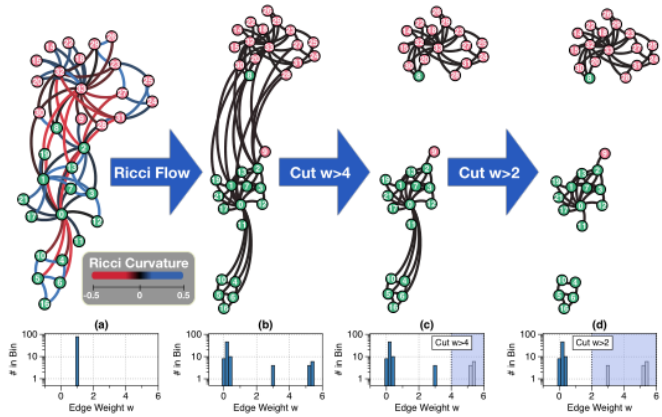

Figure 3: Ricci flow for community detection on the Karate club graph.

Discrete Ricci Flow Algorithm

Algorithm 1: Discrete Ricci Flow

Input : An undirected graph G and a real number $\delta > 0$.

Output : A weighted graph G with edge weight as Ricci flow metric on each edge.

1 Normalize the edge weight $w_{xy}^{(i)} \leftarrow d^{(i)}(x,y) \cdot \frac{|E|}{\sum_{xy \in E} d^{(i)}(x,y)}$

2 Compute the Ricci curvature of G

3 Update the edge weight by $w_{xy}^{(i+1)} \leftarrow d^{(i)}(x,y) - \varepsilon \cdot \kappa_{xy}^{(i)} \cdot d^{(i)}(x,y)$

4 Repeat $1-3$ until all $|\kappa_{xy}^{(i)} - \kappa_{xy}^{(i-1)}| < \delta$.

Figure 4: Discrete Ricci Flow Algorithm

Evaluation

1. Adjusted Rand Index (ARI): with the ground truth clustering;
2. modularity: without the ground truth clustering.

ARI on Synthetic Data

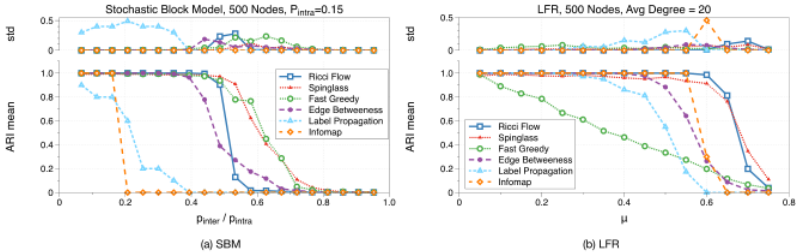

Figure 5: The accuracy of the Ricci flow method for community detection on networks.

(a) Stochastic block model (SBM) with 500 nodes and two communities of the same size.
(b) Lancichinetti-Fortunato-Radicchi (LFR) Model with 500 nodes, average degree of 20 and 38 communities.

ARI on Real-World Data

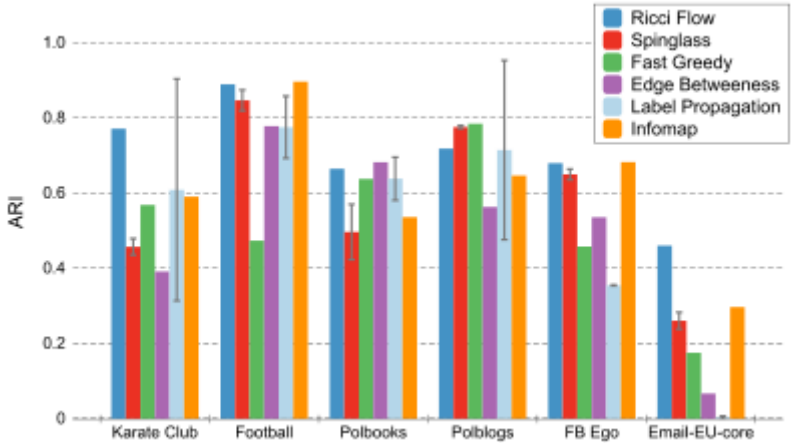

Figure 6: Real networks

Detect Hierarchical Community Structures

A GNet planar model with 1000 nodes. With different cutoff thresholds, we are able to detect communities in a hierarchical manner.

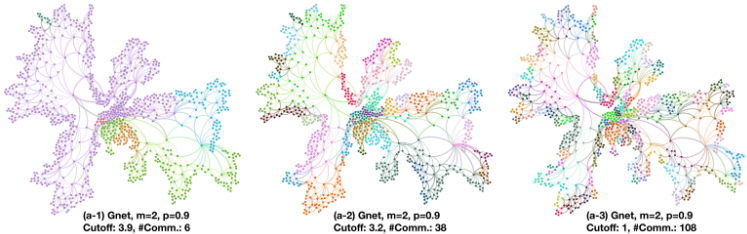

Figure 7: Illustrations of communities detected by discrete Ricci flow with three different cutoff weights

Summary

This paper considers networks as geometric objects and uses curvature and curvature guided flow to decompose networks.