CSE 6740: Computational Data Analysis

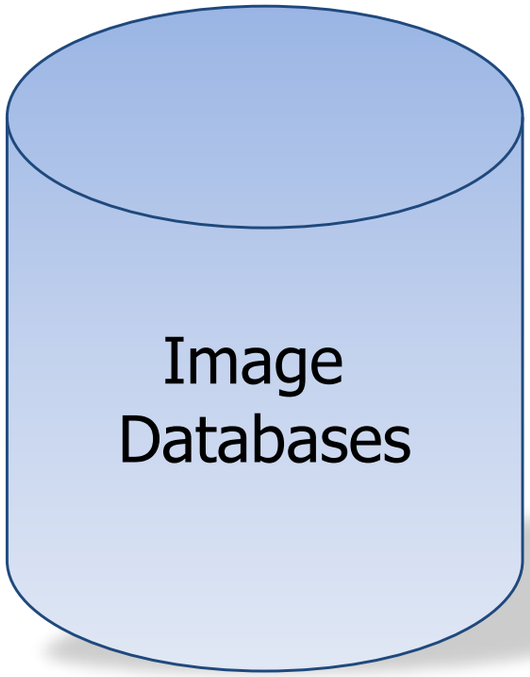
Spring 2026

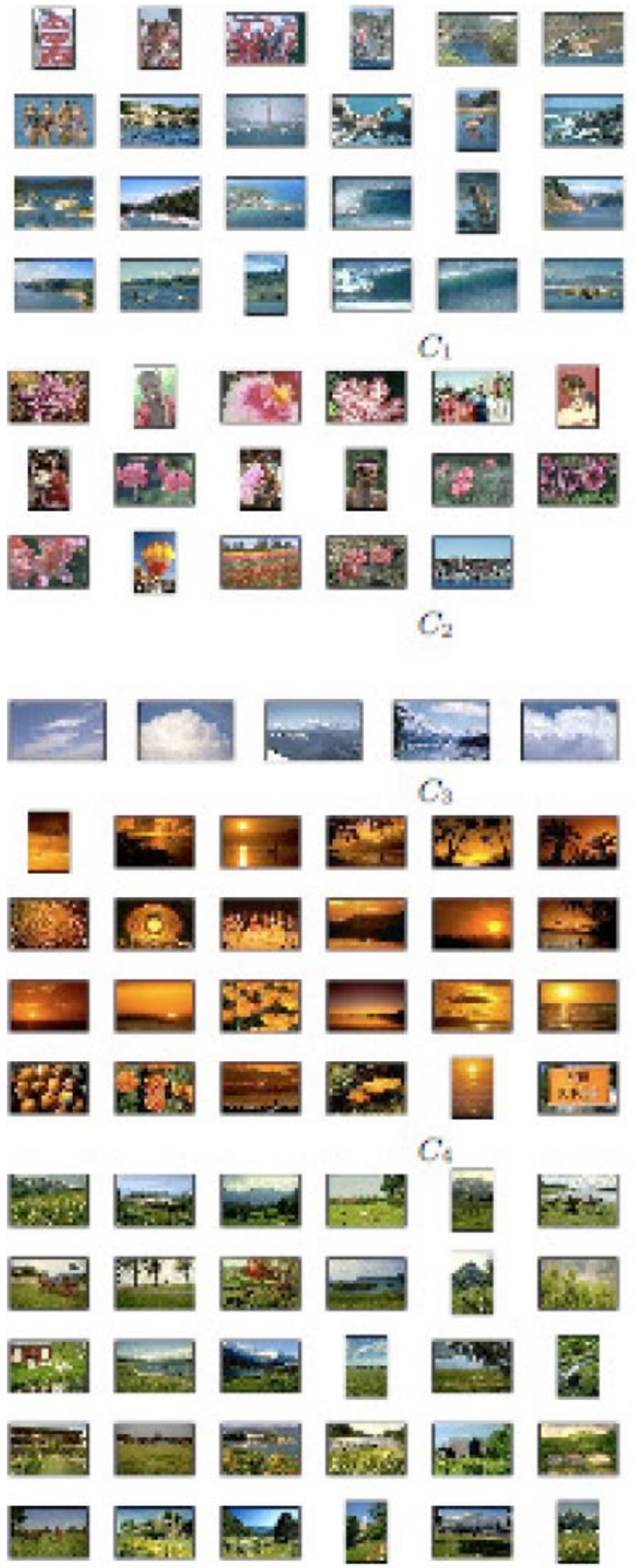
Clustering Nodes in Graphs

Anqi Wu

01/20

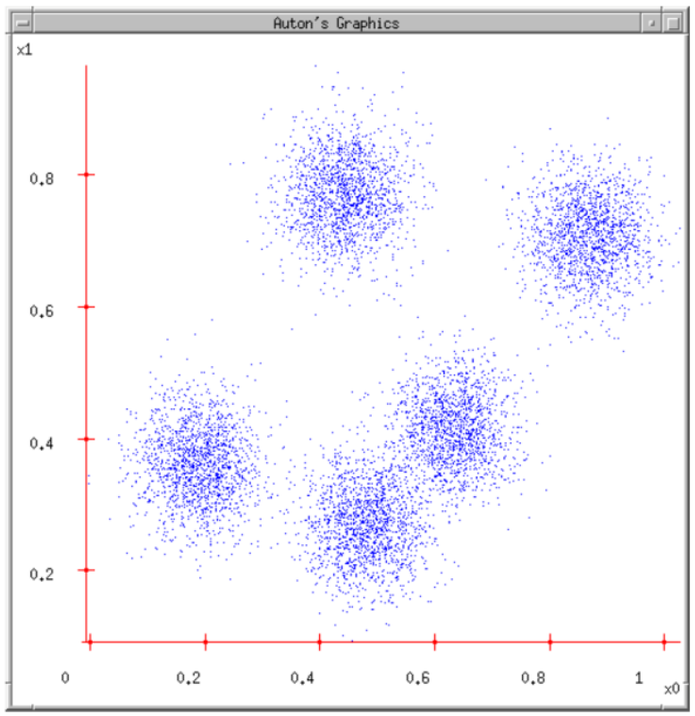
Clustering images





Goal of clustering:

Divide object into groups, and objects within a group are more similar than those outside the group



Formal statement of clustering problem

- Given m data points, $\{x^1, x^2, \dots x^m\} \in R^n$
- Find k cluster centers, $\{c^1, c^2, \dots, c^k\} \in R^n$
- And assign each data point i to one cluster, $\pi(i) \in \{1, \dots, k\}$
- Such that the averaged square distances from each data point to its respective cluster center is small

$$\min_{c,\pi} \frac{1}{m} \sum_{i=1}^{m} \left\| x^i - c^{\pi(i)} \right\|^2$$

K-means algorithm

- Initialize k cluster centers, $\{c^1, c^2, \dots, c^k\}$, randomly
- Do
  - Decide the cluster memberships of each data point, x^i, by assigning it to the nearest cluster center (cluster assignment)

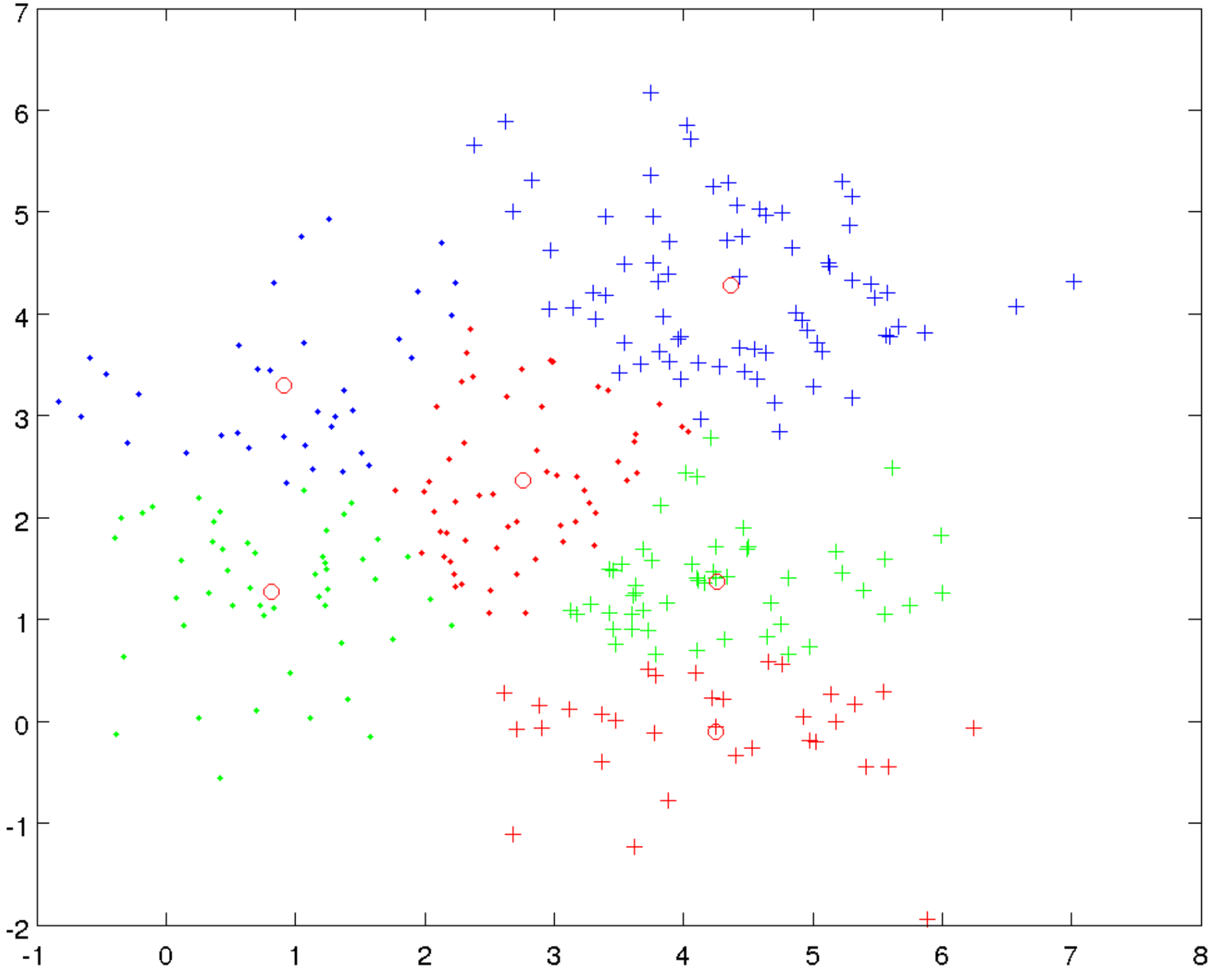
$$\pi(i) = argmin_{j=1,\dots,k} \left\| x^i - c^j \right\|^2$$

  - Adjust the cluster centers (center adjustment)

$$c^j = \frac{1}{|\{i: \pi(i) = j\}|} \sum_{i:\pi(i)=j} x^i$$

- While any cluster center has been changed

Run kmeans_animation.m

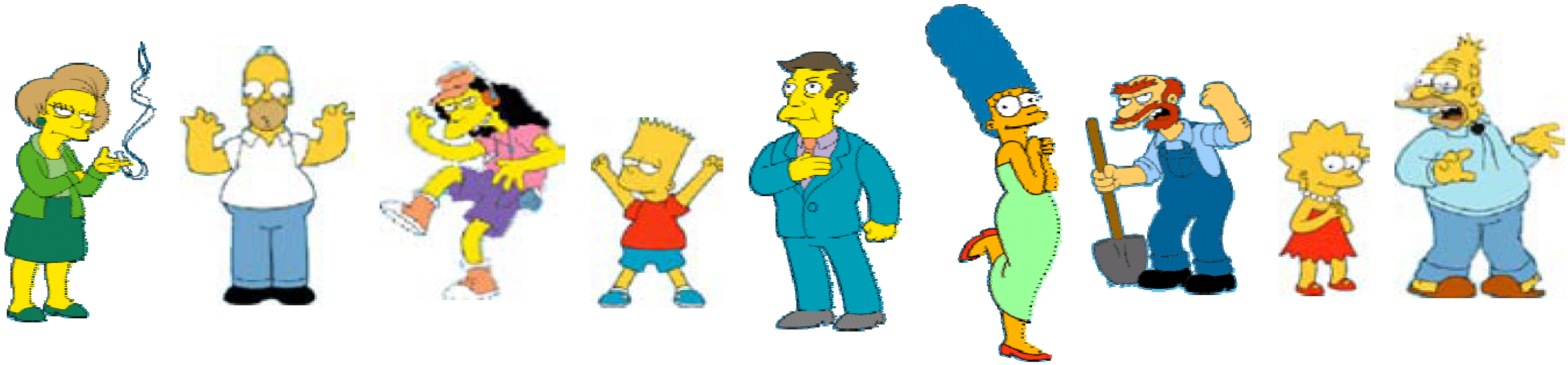


Property of K-means

- Will different initializations lead to different results?
  - Yes
  - No
  - Sometimes

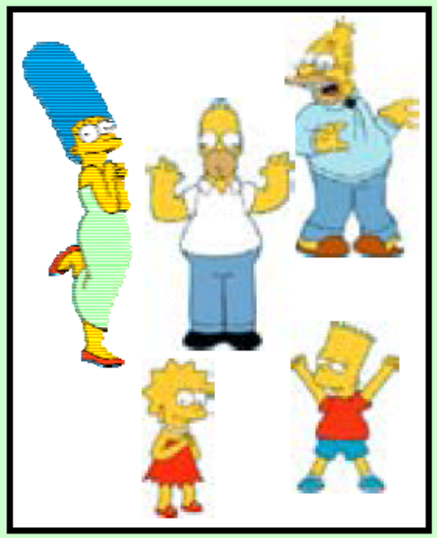
- Will the algorithm always stop after some iteration?
  - Yes
  - No (we have to set a maximum number of iterations)
  - Sometimes

Clustering is a subjective task



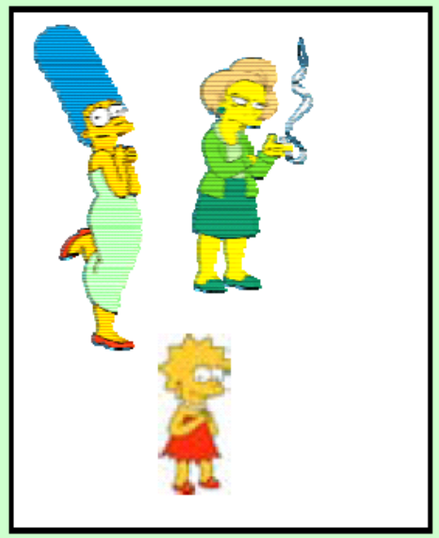
What is consider similar/dissimilar?

Clustering is subjective



Simpson's Family

School Employees



Females

Males

You pick your similarity/dissimilarity

General formulation of clustering

- Given m data points, $\{x^1, x^2, \dots x^m\} \in R^n$
- Find k cluster centers, $\{c^1, c^2, \dots, c^k\} \in R^n$
- And assign each data point i to one cluster, $\pi(i) \in \{1, \dots, k\}$
- Such that the sum of the squared distances from each data point to its respective cluster center is minimized

$$\min_{c,\pi} \sum_{i=1}^{m} d\left(x^i, c^{\pi(i)}\right)^2$$

K-means algorithm

- Initialize k cluster centers, $\{c^1, c^2, \ldots, c^k\}$, randomly
- Do
  - Decide the cluster memberships of each data point, x^i, by assigning it to the nearest cluster center (cluster assignment)

$$\pi(i) = argmin_{j=1,\ldots,k}\, d(x^i, c^j)$$

  - Adjust the cluster centers (center adjustment)

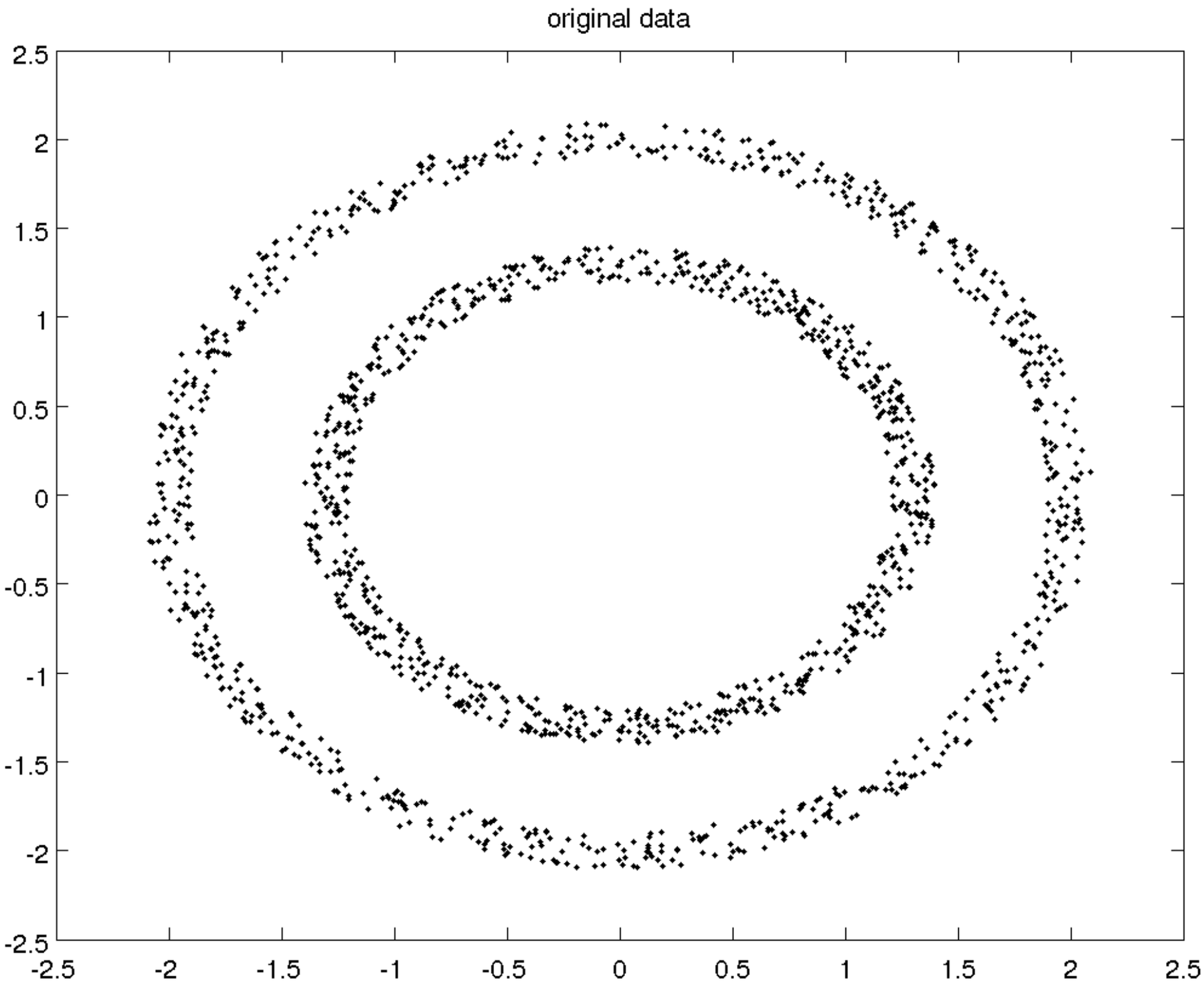
$$c^j = argmin_{v \in R^n} \sum_{i:\pi(i)=j} d(x^i, v)^2$$

- While any cluster center has been changed

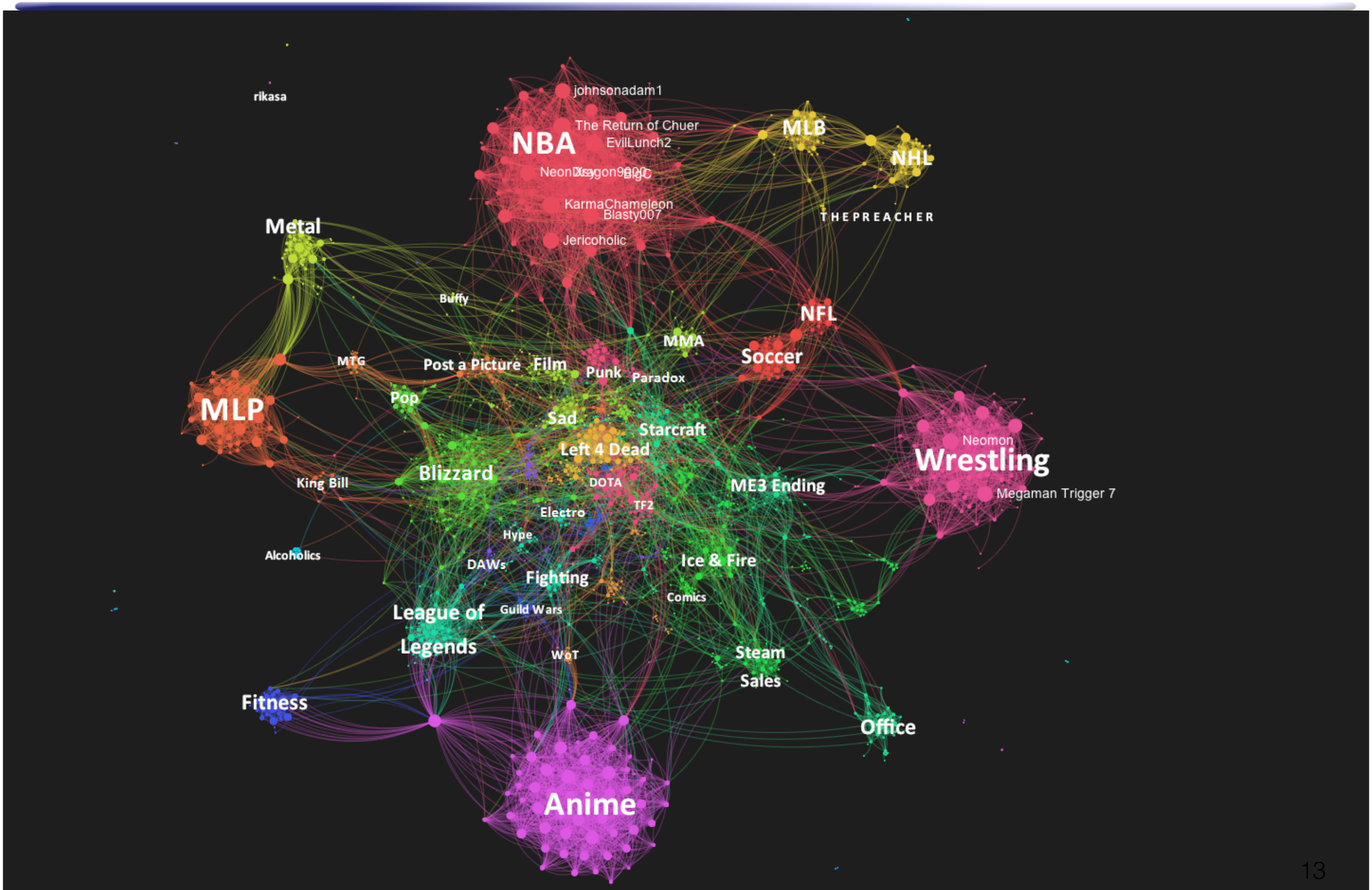
Kmeans vs Hierarchal clustering

	Kmeans	HC
Number of clusters	If there is a specific number of clusters in the dataset, but the group they belong to is unknown	It is easier to determine the number of clusters by hierarchical clustering's dendrogram
Cluster result	unstructured	more interpretable and informative
Time complexity	$O(n \times k \times t)$	$O(n^3)$
Space complexity	$O(n(d + k))$	$O(n^2)$
With a large number of data points	compute faster	compute slower

How about this dataset? (Run test_tworings.m)



How about clustering nodes in social networks



Run demo test_football.m

COLLEGE FOOTBALL SCHEDULES

FBS | FCS

By Week: 1 · 2 · 3 · 4 · 5 · 6 · 7 · 8 · 9 · 10 · 11 · 12 · 13 · 14 · 15 · 16

Buy College Football Tickets

WEEK 1

SATURDAY, AUG. 23

GAME	TIME/SCORE	TV	LOCATION/TICKETS
Sam Houston St. at E. Washington	Eastern Washington 56-35	ESPN	Woodward Stadium

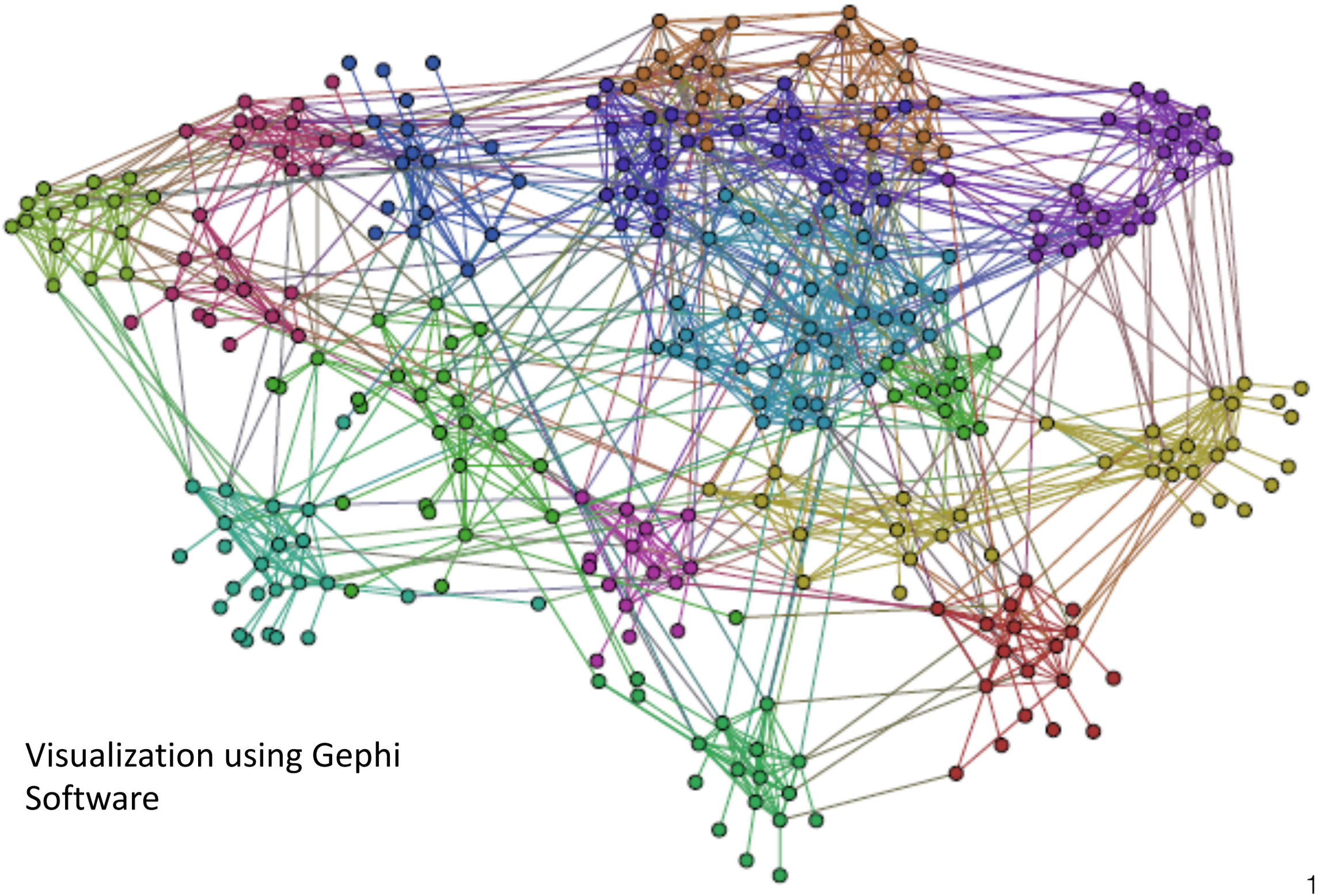
WEDNESDAY, AUG. 27

GAME	TIME/SCORE	TV	LOCATION/TICKETS
Abil Chr. at Georgia State	Georgia State 38-37	ESPNU	Georgia Dome

THURSDAY, AUG. 28

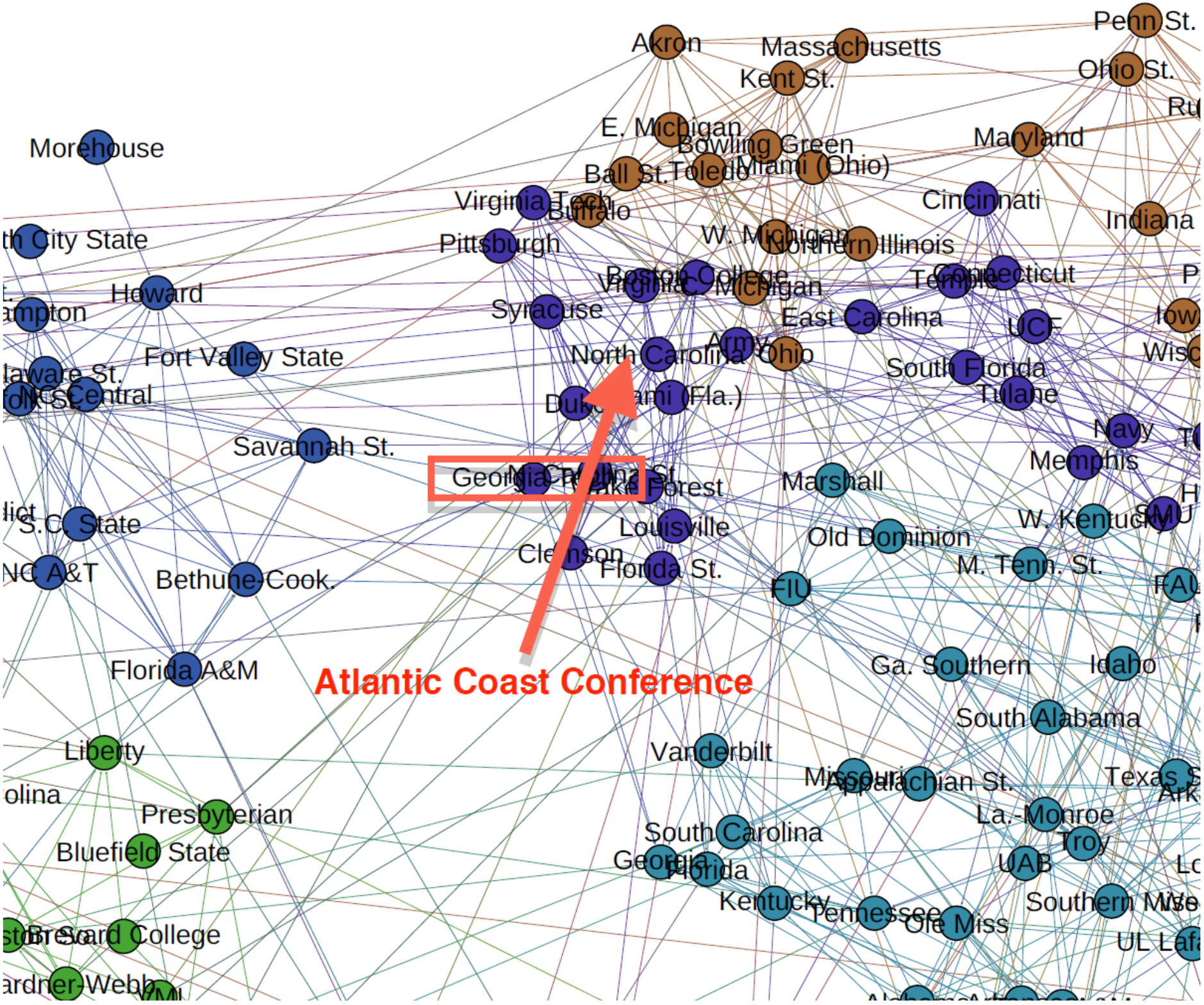
GAME	TIME/SCORE	TV	LOCATION/TICKETS
Texas A&M at South Carolina	Texas A&M 52-28	SEC Network	Williams-Brice Stadium
E. Illinois at Minnesota	Minnesota 42-20	Big Ten Network	TCF Bank Stadium
Presbyterian at Northern Illinois	Northern Illinois 55-3		Huskie Stadium
Missouri St. at Northwestern St.	Missouri State 34-27		Turpin Stadium
Bryant at Stony Brook	Bryant 13-7		
Wake Forest at La.-Monroe	Louisiana-Monroe 17-10	ESPNU	Malone Stadium
Chattanooga at C. Michigan	Central Michigan 20-16		Kelly/Shorts Stadium
Howard at Akron	Akron 41-0		InfoCision Stadium - Summa Field
Charlotte at Campbell	Charlotte 33-9		Barker-Lane Stadium
Reinhardt at Mercer	Mercer 45-42		Moye Complex
E. Kentucky at Robert Morris	Eastern Kentucky 29-10		Joe Walton Stadium
Point U at Charleston So.	Charleston Southern 61-9		CSU Field
Missouri Baptist at SE Missouri St.	Southeast Missouri State 77-0		Houck Stadium
Idaho State at Utah	Utah 56-14	PAC-12 Network	Rice Eccles Stadium
Valparaiso at W. Illinois	Western Illinois 45-6		Hanson Field
Boise St. at Ole Miss	Ole Miss 35-13	ESPN	Georgia Dome
Kentucky Chr. at Tenn. Tech	Tennessee Tech 33-7		Tucker Stadium

Clustering nodes in a network



Visualization using Gephi Software

Clusters make lots of senses (zoom in)

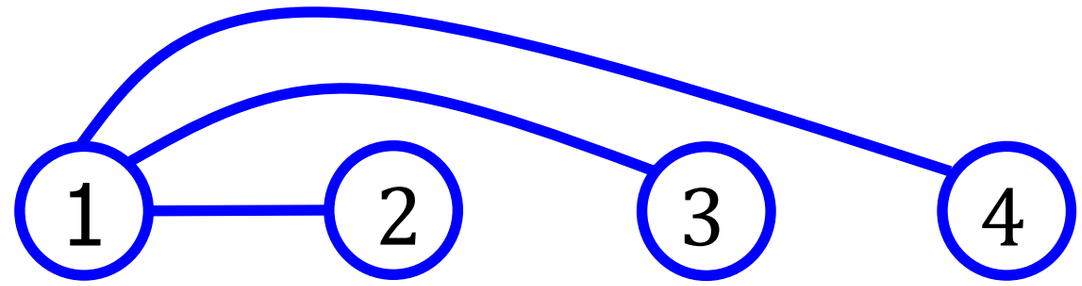


Spectral clustering algorithm

- Step 1: represent graph as adjacency matrix $A \in R^{m \times m}$

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$D = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Step 2: form a special matrix $L = D - A$, the graph Laplacian

$$L = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

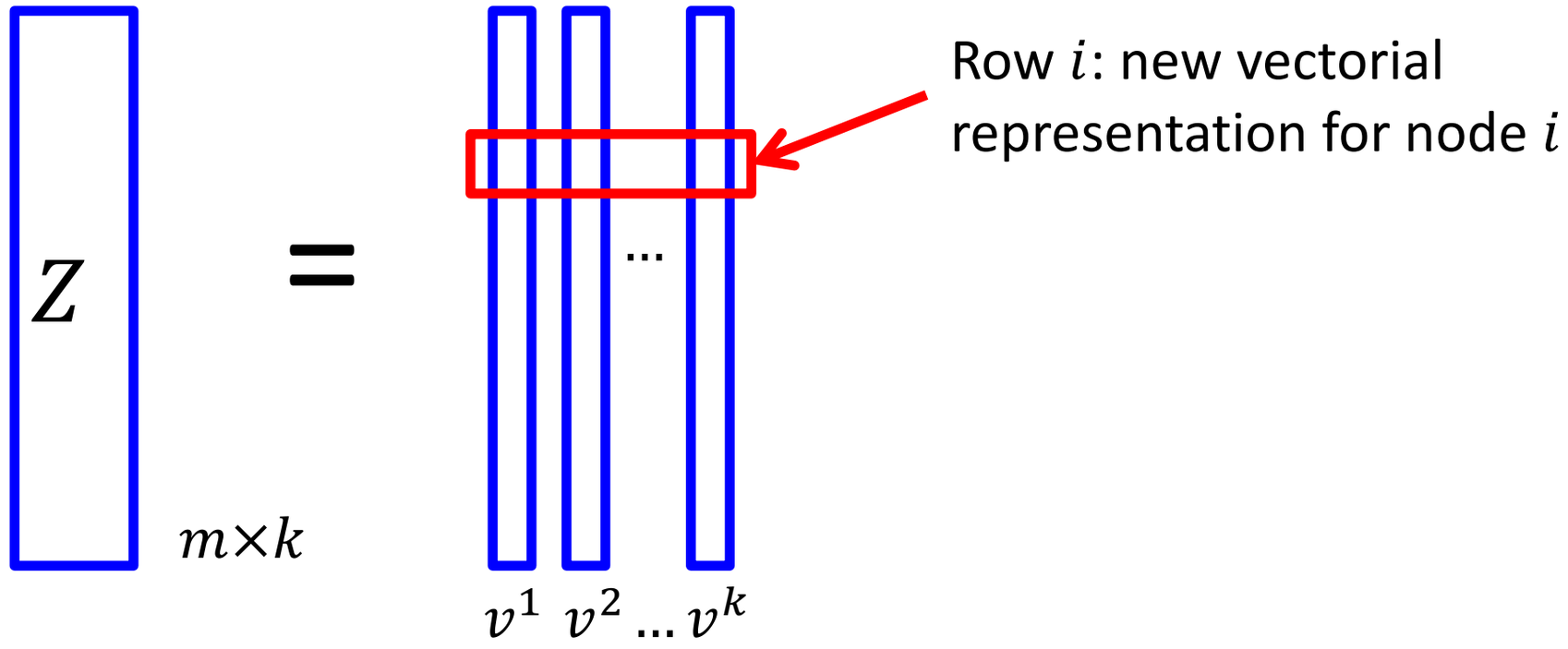
$$L = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$$

Spectral clustering algorithm (cont.)

- Step 3: compute k eigenvectors, $v^1, v^2, \ldots, v^k$, of L corresponding to the k smallest eigenvalues ($k \ll m$)

$$Lv^1 = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix} v^1 \stackrel{?}{=} \lambda_1 v^1$$

- Step 4: run kmeans algorithm on $Z = \left(v^1, v^2, \ldots v^k\right)$ by treating each row as a new data point



Why Laplacian matrix L?

Step 1: Clustering as a Cut Problem on a Graph

We want to partition nodes into two groups S and $\bar{S}$. For a weighted graph with edge weights w_{ij}, define the cut:

$$\mathrm{Cut}(S, \bar{S}) = \sum_{i \in S, j \in \bar{S}} w_{ij}.$$

Goal: make cross-group connections as weak as possible.

Why Laplacian matrix L?

Step 2: Encode the Partition with a Discrete Indicator

Define a discrete indicator vector $x \in \{+1, -1\}^n$:

$$x_i = \begin{cases} +1 & i \in S \\ -1 & i \in \bar{S} \end{cases}$$

Then for any pair (i, j):

$$(x_i - x_j)^2 = \begin{cases} 0 & \text{same group} \\ 4 & \text{different groups} \end{cases}$$

So $(x_i - x_j)^2$ acts like a "soft switch" that detects boundary edges.

Why Laplacian matrix L?

Step 3: A Cut-Counting Objective (No L Yet)

Consider the objective

$$J(x) \;=\; \frac{1}{2}\sum_{i,j} w_{ij}(x_i - x_j)^2.$$

If $x \in \{+1,-1\}^n$, only cross-group pairs contribute:

$$J(x) \;=\; \frac{1}{2}\sum_{(i,j)\in \mathrm{cut}} w_{ij}\cdot 4 \;=\; 2\,\mathrm{Cut}(S,\bar{S}).$$

Thus, for discrete indicators,

$$\min_{x\in\{+1,-1\}^n} J(x) \quad\Longleftrightarrow\quad \min \mathrm{Cut}(S,\bar{S}).$$

Why Laplacian matrix L?

Step 4: Why Relax to Continuous x?

The discrete optimization

$$\min_{x \in \{+1,-1\}^n} J(x)$$

is combinatorial and hard.
Spectral methods relax the constraint:

$$x \in \mathbb{R}^n,$$

while keeping the same objective form $J(x)$.
We also need constraints to avoid trivial solutions:

$$\|x\|^2 = 1, \qquad x \perp \mathbf{1}.$$

Why Laplacian matrix L?

Step 5: Now Introduce the Laplacian

Let W be the weighted adjacency matrix, and D be the degree matrix:

$$D_{ii} = \sum_j w_{ij}, \qquad L = D - W.$$

A key identity shows the objective can be written compactly:

$$\frac{1}{2}\sum_{i,j} w_{ij}(x_i - x_j)^2 \;=\; x^\top L x.$$

So $J(x) = x^\top L x$. This is not a new objective: it is exactly the same cut-counting form.

$n \times n$ $(Ax)_{n=1}^{n}$

Why Laplacian matrix L?

Step 6: Continuous Minimization Leads to Eigenvectors

We solve the relaxed problem:

$$\min_{x \in \mathbb{R}^n} x^\top L x \quad \text{s.t.} \quad \|x\|^2 = 1, \; x \perp \mathbf{1}.$$

The solution is the eigenvector corresponding to the smallest eigenvalue that is not the trivial constant eigenvector. This is the second smallest eigenvector of L.

Graph Laplacian

- Graph Laplacian $L \in R^{m \times m}$ is a matrix representation of graph

- Capture information on many graph properties (eg. use its eigenvalues to count the number spanning trees)

- Computation $L = D - A$
  - Start with (weighted) adjacency matrix A

$$A_{ij} = \begin{cases} w_{ij} > 0, \text{if node } i \text{ and } j \text{ are neighbors} \\ 0, \quad \text{if } i \text{ and } j \text{ are not direct neighors} \end{cases}$$

  - Diagonal degree matrix $D = diag(A1)$

$$D_{ii} = \sum_{j \in N(i)} w_{ij}$$

$$x^T L x = x^T (D - A) x$$
$$= x^T D x - x^T A x = \sum_{i \in V} d_i x_i^2 - \sum_{\{i,j\} \in E} 2 x_i x_j$$
$$= \sum_{(i,j) \in E} (x_i^2 + x_j^2) - \sum_{(i,j) \in E} 2 x_i x_j$$
$$= \sum_{(i,j) \in E} (x_i - x_j)^2 \geq 0$$

Graph Laplacian example

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$D = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

Property I of Graph Laplacian

- $L = D - A$

- The multiplicity of the eigenvalue 0 corresponds to the number of connected components in the graph

- Example

$$L = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

$$Lv_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

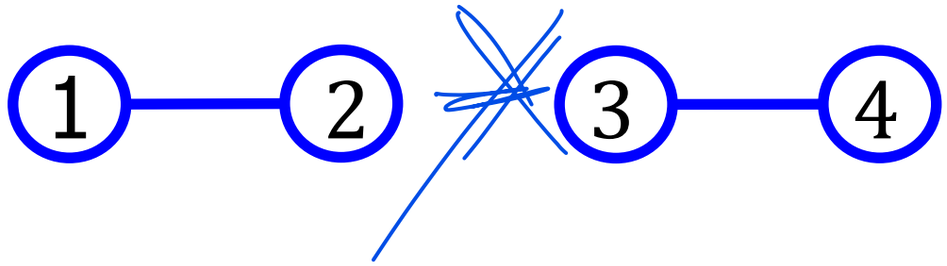
$$Lv_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Property II of Graph Laplacian

- $L = D - A$

- The eigenvectors with eigenvalue 0 contains cluster assignment information

- Example



$$L = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

$$Lv^1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$Lv^2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$x^T L x = \sum_{i,j} (x_i - x_j)^2$

$= \sum_{1,2} (1-1)^2$

$+ \sum_{3,4} (2-2)^2$

What if the graph has only 1 component

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$D = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

Properties of Graph Laplacian

- $L = D - A$

- The smallest eigenvalue of L is 0, corresponding a constant eigenvector $\frac{1}{\sqrt{m}}1$

- Example

$$L = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

$$\frac{1}{\sqrt{4}}L1 = \frac{1}{\sqrt{m}}\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

What if the graph has k components

- If a graph has k connected components (or k clusters)

- The graph Laplacian has k blocks

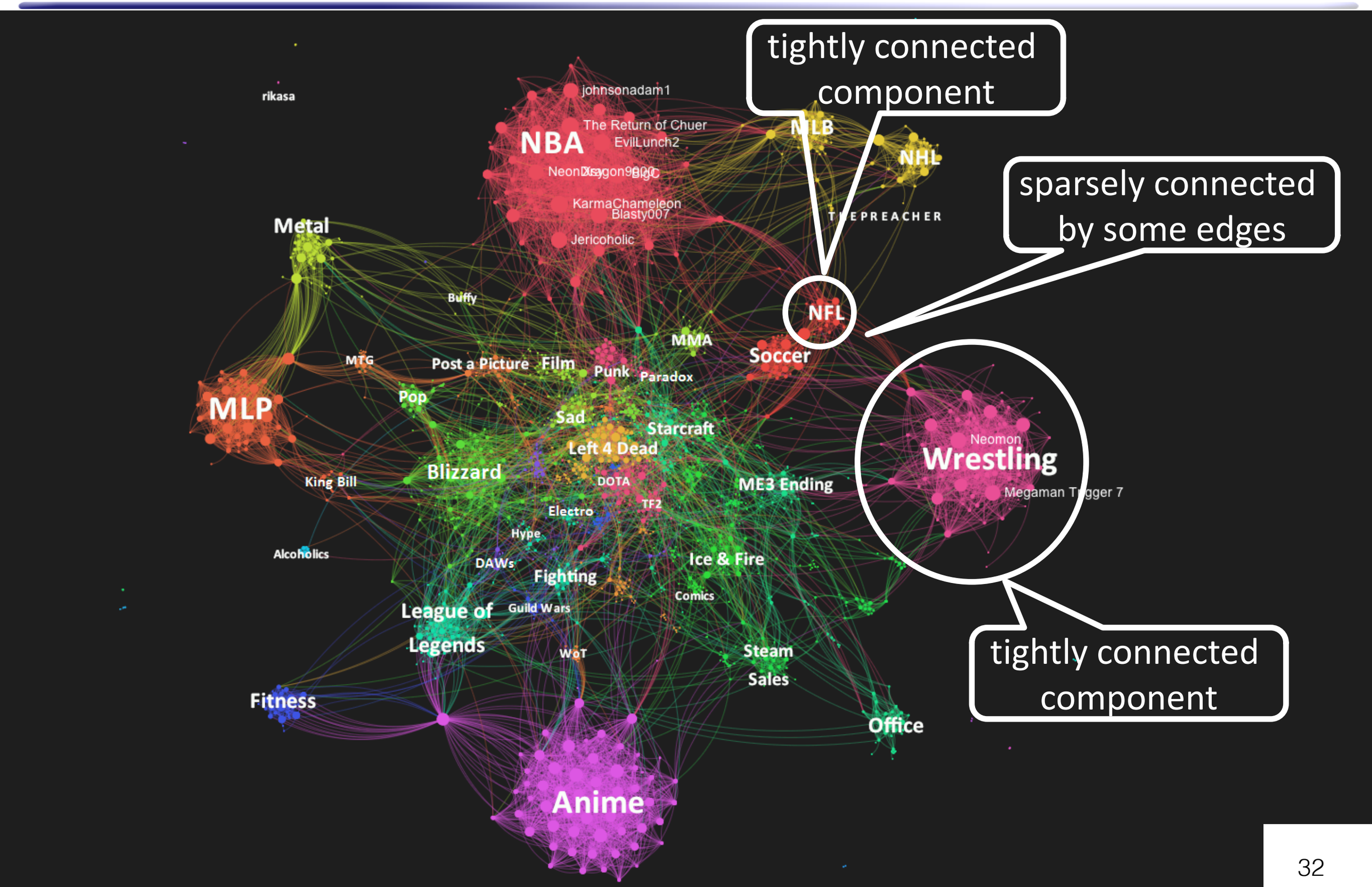
$$L = \begin{pmatrix} L_1 & 0 & 0 & 0 \\ 0 & L_2 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & L_k \end{pmatrix}$$

- The graph Laplacian has k eigenvectors with zero eigenvalues

- Eigenvector 1 is constant in block 1, but 0 in other blocks;
  eigenvector 2 is constant in block 2, but 0 in other blocks;

  …

Real world social networks



In most real networks

- If a graph has k tightly connected components (or k clusters) with sparsely connected edges
- The graph Laplacian has approximately k blocks
- The graph Laplancian has k eigenvectors with small eigenvalues
- Eigenvector 1 is approximately constant in block 1, but 0 in other blocks; eigenvector 2 …

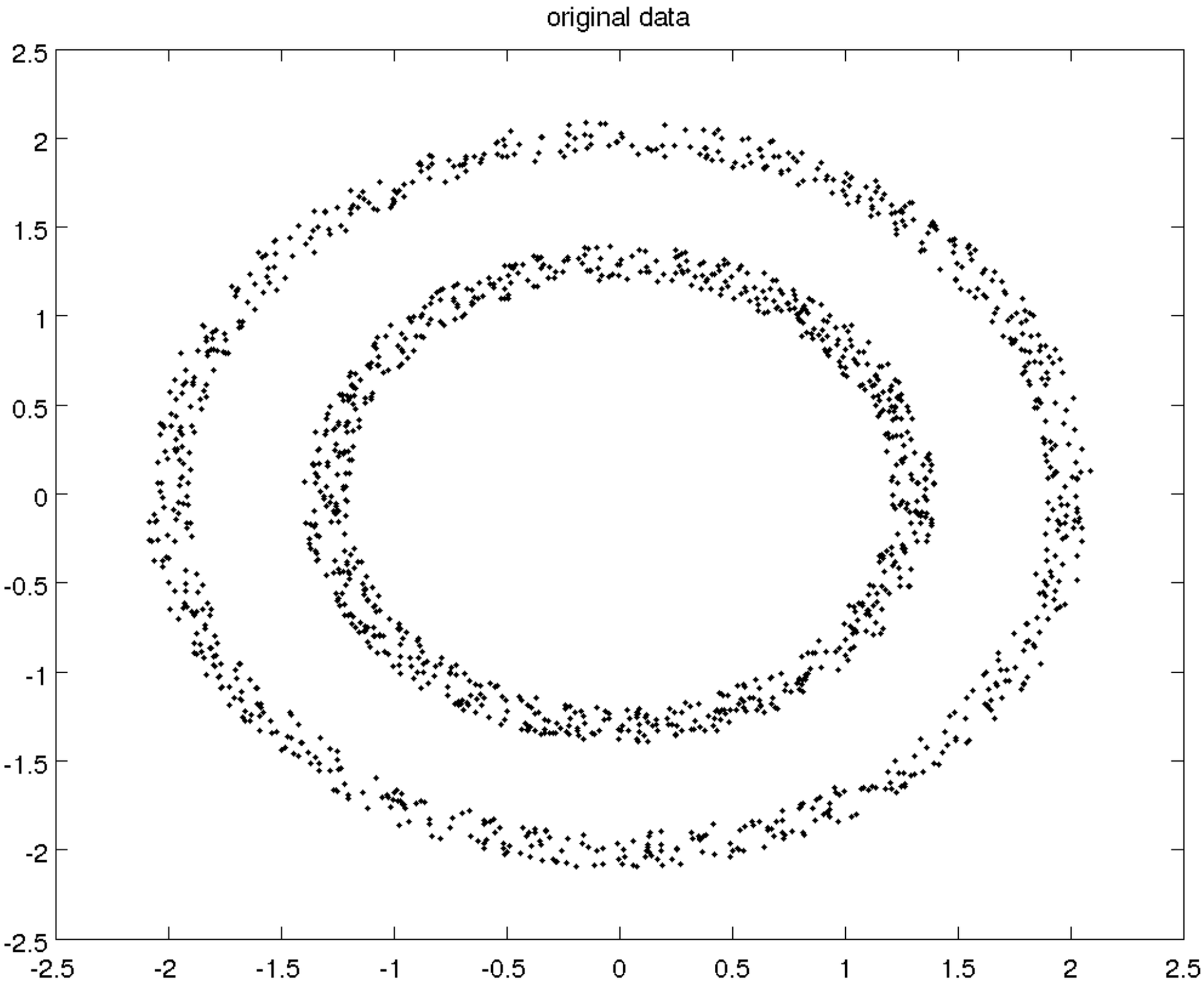
High level idea of spectral clustering

- Examine the properties of graph Laplacian for the perfect cases
  - The number of 0 eigenvalues corresponds to the number of connected components
  - Eigenvectors correspond to cluster assignment
- Then use the intuition from perfect cases to design algorithms for the imperfect case.
  - Eigenvectors not longer correspond exactly cluster indicator
  - Perform post processing to obtain cluster assignment

Summary of spectral clustering

- Step 1: represent graph as adjacency matrix $A \in R^{m \times m}$
- Step 2: form a special matrix $L = D - A$, the graph Laplacian
- Step 3: compute k eigenvectors, $v^1, v^2, \ldots, v^k$, of L corresponding to the k smallest eigenvalues ($k \ll m$)
- Step 4: run kmeans algorithm on $Z = \left(v^1, v^2, \ldots v^k\right)$ by treating each row as a new data point

How about this dataset?

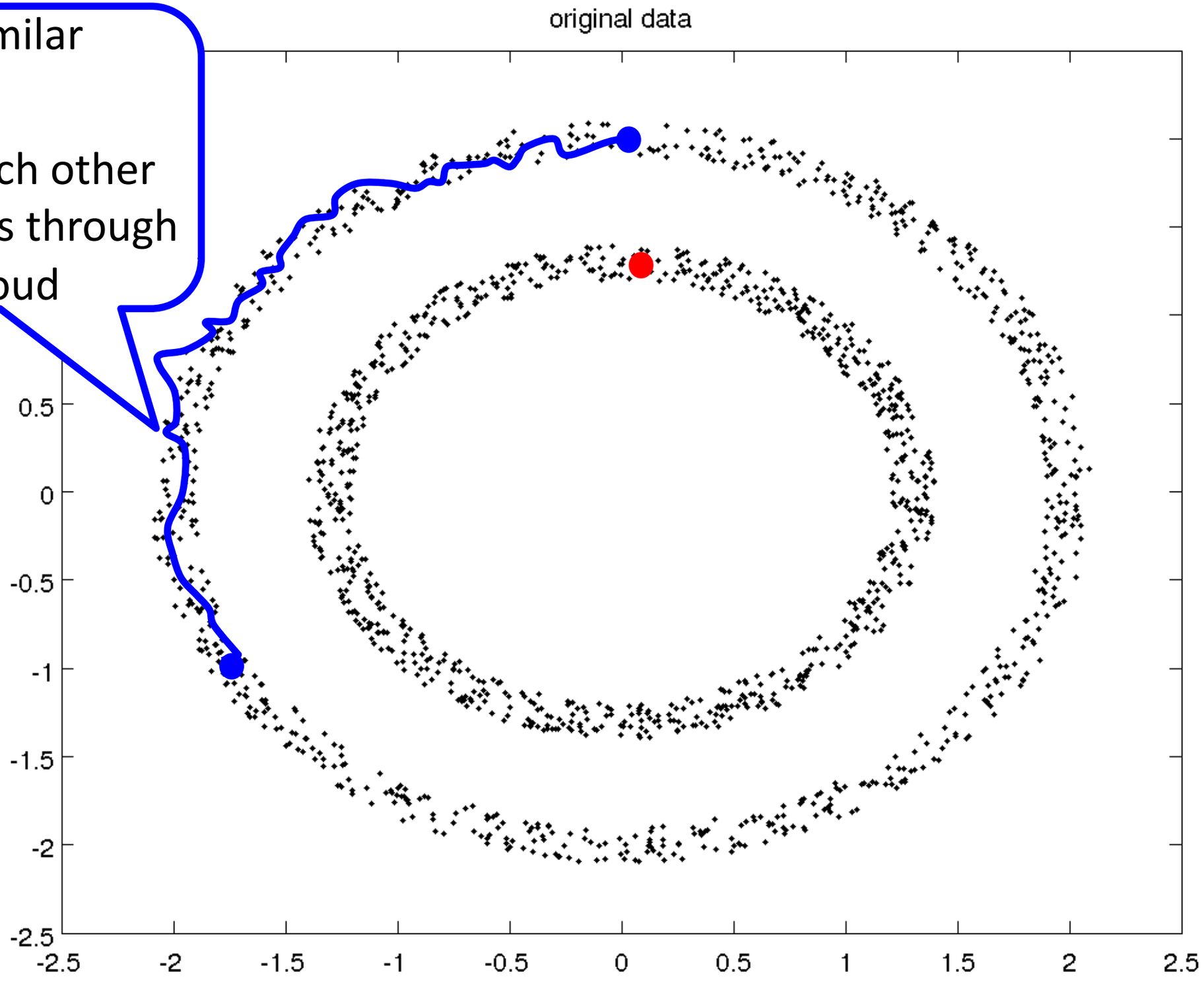


What's a reasonable similarity measure?

points similar

=

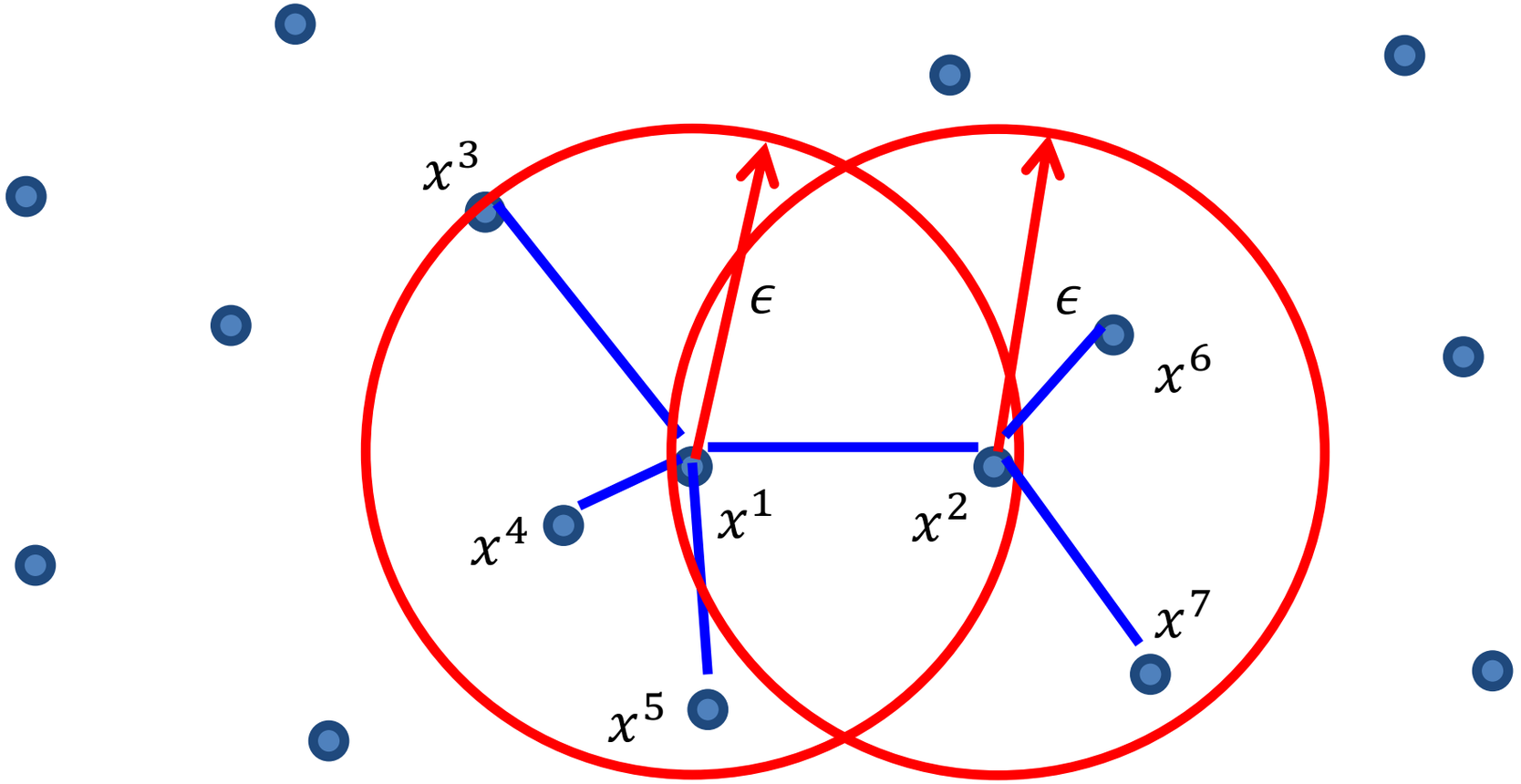
can reach each other by small jumps through data cloud



Neighbor graph

- Given m data points, threshold ϵ, construct matrix $A \in R^{m \times m}$

$$A^{ij} = \begin{cases} 1, \text{if } \left\| x^i - x^j \right\| \leq \epsilon \\ 0, \qquad \text{otherwise} \end{cases}$$



Spectral clustering for vectorial data

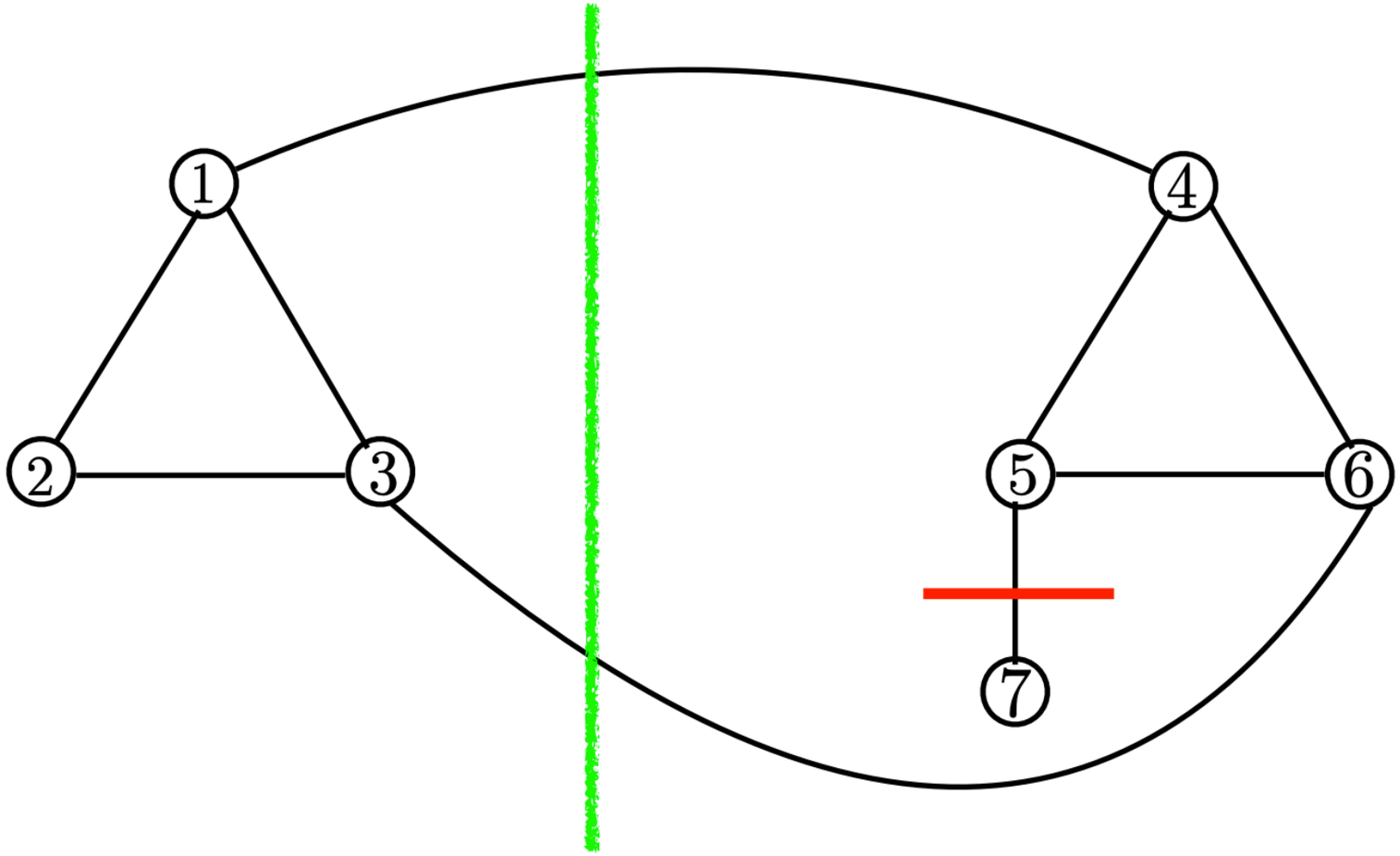
- Given m nodes, $\{x^1, x^2, \dots x^m\} \in R^n$
- Step 1: build an adjacency matrix A using nearest neighbors
- Step 2: represent graph as adjacency matrix $A \in R^{m \times m}$
- Step 3: form a special matrix $L = D - A$, the graph Laplacian
- Step 4: compute k eigenvectors, $v^1, v^2, \dots, v^k$, of L corresponding to the k smallest eigenvalues ($k \ll m$)
- Step 5: run kmeans algorithm on $Z = \left(v^1, v^2, \dots v^k\right)$ by treating each row as a new data point

Variants of spectral clustering (Ng et al.)

- Given m data points (nodes), $\{x^1, x^2, \ldots x^m\} \in R^n$
- Build an adjacency matrix A using kernel functions
- Compute $B = D^{-1/2}AD^{-1/2}$ (or $B = I - D^{-1/2}AD^{-1/2}$), where $D = diag(A1)$
- Compute k eigenvectors, $v^1, v^2, \ldots, v^k$, of B corresponding to the k largest (or smallest) eigenvalues
- Run kmeans algorithm on $Z = \left(v^1, v^2, \ldots v^k\right)$ by treating each row as a new data point

Unnormalized vs normalized

- Unnormalized Spectral clustering aims to cluster based on minimizing cut
- cut: Number of edges that need to be deleted to have no links between the cluster and other nodes outside
- But is cut the right metric?



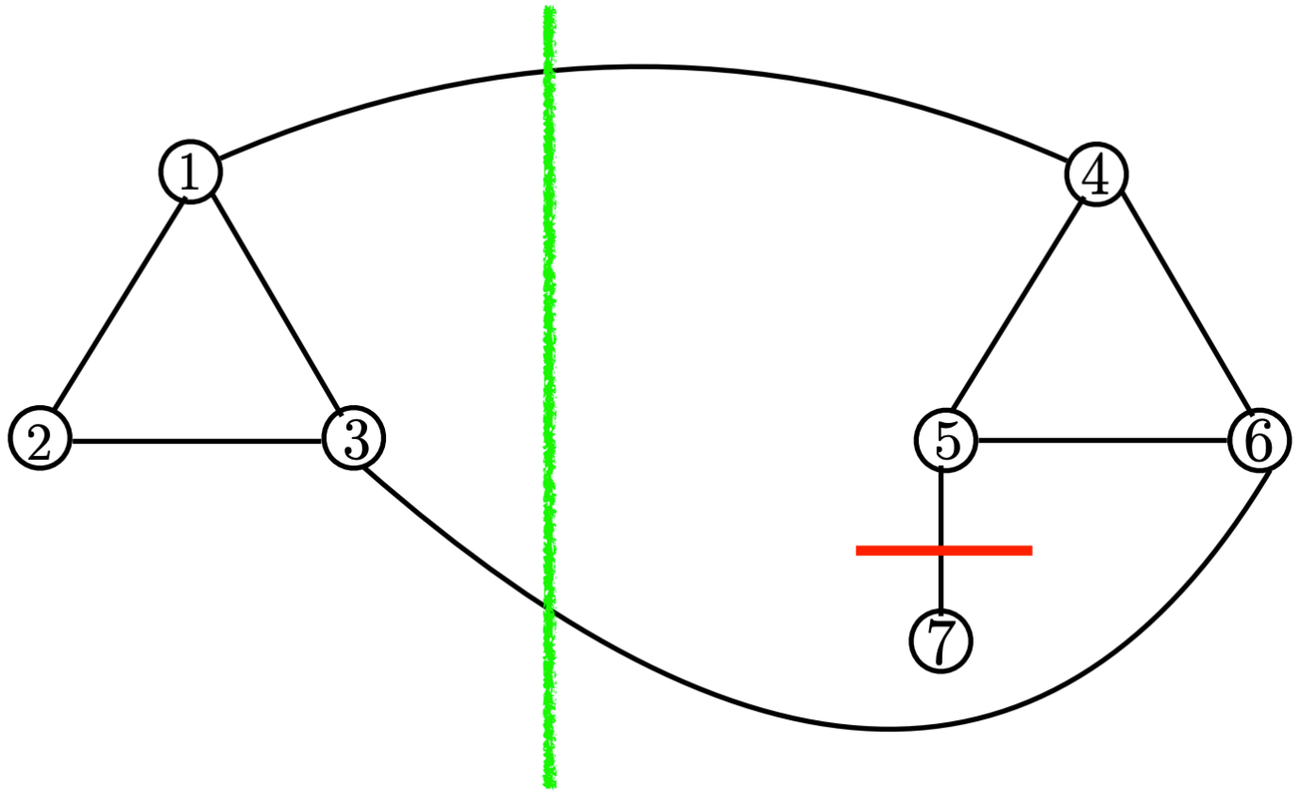
Unnormalized vs normalized

- Normalized cut: Minimize sum of ratio of number of edges cut per cluster and number of edges within cluster

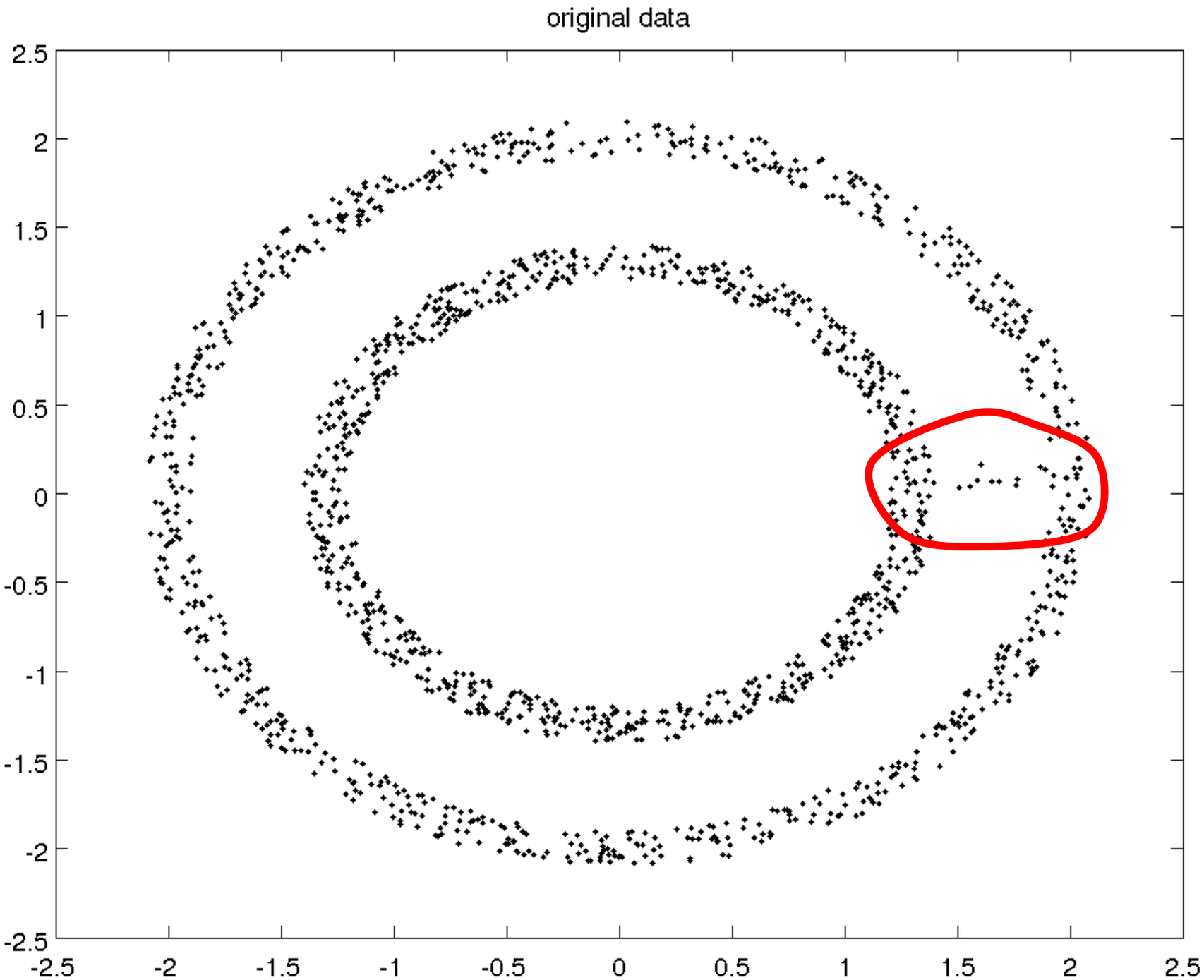
$$\text{NCUT} = \sum_j \frac{\text{CUT}(C_j)}{\text{Edges}(C_j)}$$

- Example K = 2

$$\text{CUT}(C_1, C_2)\left(\frac{1}{\text{Edges}(C_1)} + \frac{1}{\text{Edges}(C_2)}\right)$$



What happens by adding more data points?



What if my graph is large?

- Key challenge:
  - Eigen-decomposition of a large graph Laplacian is expensive
  - How to scale the algorithm up to millions of nodes?

- One solution:
  - Use randomized linear algebra to approximately find top eigenvectors of $B = D^{-1/2}AD^{-1/2} \in R^{m \times m}$ (big)
    - Generate a Gaussian random matrix $\Omega \in R^{d \times m}$ $(d \ll m)$
    - Find a set of orthonormal basis Q for column of

$$Y = B\Omega^{\mathsf{T}}$$

      eg. using Gram-Schmidt orthogonalization
    - Eigendecomposition of $C = \Omega B\Omega^{\mathsf{T}} \in R^{d \times d}$ (tiny)

$$C = U\Lambda U^{\mathsf{T}}$$

    - $Z = QU$
    - References: Halko, Martinsson and Tropp (2009).

Run test_citationgraph.m

- Compare matlab eigs to the randomized eigendecomposition
- Compare matlab implemenation of kmeans with our vectorized implementation
- Our codes run much faster.