

Community Detection: Spectral Clustering

CS224W: Analysis of Networks
Jure Leskovec, Stanford University
<http://cs224w.stanford.edu>

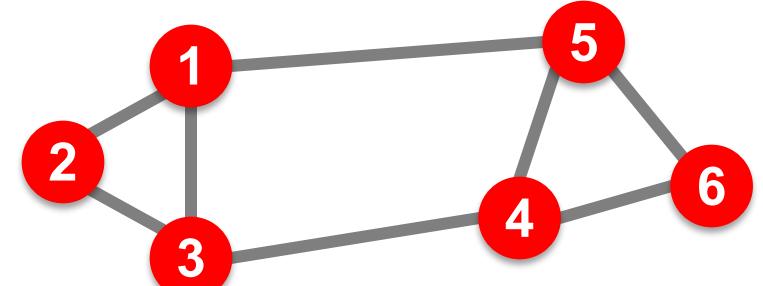


Spectral Clustering Algorithms

- Three basic stages:
 - 1) Pre-processing
 - Construct a matrix representation of the graph
 - 2) Decomposition
 - Compute eigenvalues and eigenvectors of the matrix
 - Map each point to a lower-dimensional representation based on one or more eigenvectors
 - 3) Grouping
 - Assign points to two or more clusters, based on the new representation
- But first, let's define the problem

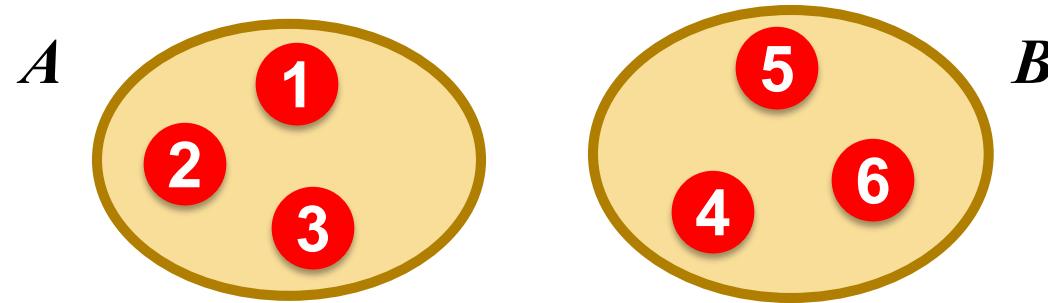
Graph Partitioning

- Undirected graph $G(V, E)$:



- Bi-partitioning task:

- Divide vertices into two disjoint groups A, B



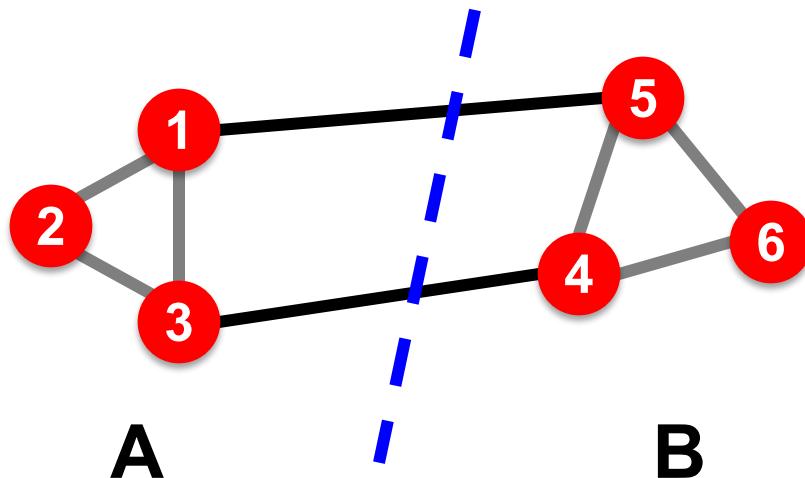
- Questions:

- How can we define a “good” partition of G ?
 - How can we efficiently identify such a partition?

Graph Partitioning

■ What makes a good partition?

- Maximize the number of within-group connections
- Minimize the number of between-group connections

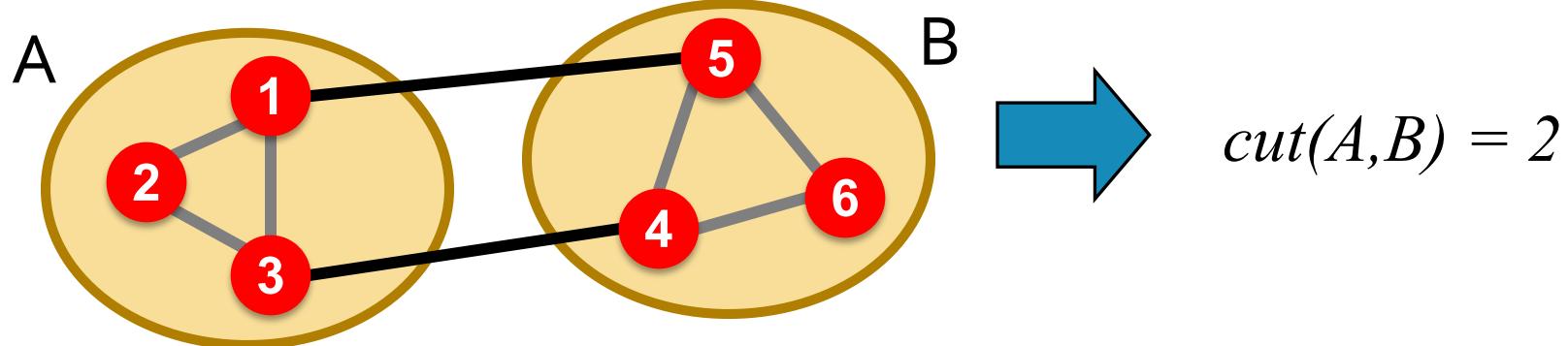


Graph Cuts

- Express partitioning objectives as a function of the “edge cut” of the partition
- **Cut:** Set of edges with only one vertex in a group:

$$cut(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

If the graph is weighted w_{ij} is the weight, otherwise, all $w_{ij}=\{0,1\}$



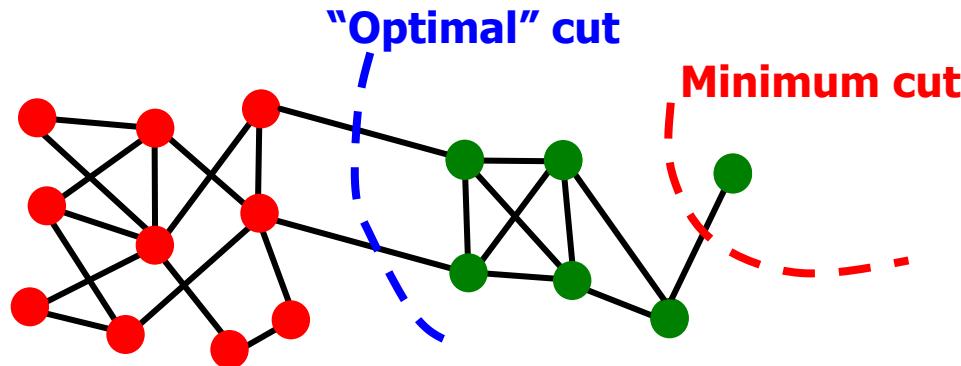
Graph Cut Criterion

- Criterion: **Minimum-cut**

- Minimize weight of connections between groups

$$\arg \min_{A,B} \text{cut}(A,B)$$

- Degenerate case:



- Problem:

- Only considers external cluster connections
 - Does not consider internal cluster connectivity

Graph Cut Criterion

■ Criterion: **Conductance** [Shi-Malik, '97]

- Connectivity between groups relative to the density of each group

$$\phi(A, B) = \frac{cut(A, B)}{\min(vol(A), vol(B))}$$

$vol(A)$: total weighted degree of the nodes in

A : $vol(A) = \sum_{i \in A} k_i$ (number of edge end points in A)

■ Why use this criterion?

- Produces more balanced partitions

■ How do we efficiently find a good partition?

- **Problem:** Computing best conductance cut is NP-hard

Spectral Graph Partitioning

- A : adjacency matrix of undirected G
 - $A_{ij} = 1$ if (i, j) is an edge, else 0
- x is a vector in \mathbb{R}^n with components (x_1, \dots, x_n)
 - Think of it as a label/value of each node of G
- **What is the meaning of $A \cdot x$?**

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$
$$y_i = \sum_{j=1}^n A_{ij} x_j = \sum_{(i,j) \in E} x_j$$

- Entry y_i is a sum of labels x_j of neighbors of i

What is the meaning of Ax ?

- **j^{th} coordinate of $A \cdot x$:**

- Sum of the x -values of neighbors of j
 - Make this a new x -value at node j

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$$A \cdot x = \lambda \cdot x$$

- **Spectral Graph Theory:**

- Analyze the “spectrum” of matrix representing G
 - **Spectrum:** Eigenvectors $x^{(i)}$ of a graph, ordered by the magnitude (strength) of their corresponding eigenvalues λ_i : $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$

$$\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$$

Note: We sort λ_i in ascending (not descending) order!

Example: d-regular graph

- Suppose all nodes in G have degree d (G is d -regular) and G is connected
- **What are some eigenvalues/vectors of G ?**

$$A \cdot x = \lambda \cdot x \quad \text{What is } \lambda? \text{ What } x?$$

- Let's try: $x = (1, 1, \dots, 1)$
- Then: $A \cdot x = (d, d, \dots, d) = \lambda \cdot x$. So: $\lambda = d$
- We found an eigenpair of G :
 $x = (1, 1, \dots, 1), \lambda = d$
- d is the largest eigenvalue of A (see next slide)

Remember the meaning of $y = A \cdot x$:

Note, this is just one eigenpair. An n by n matrix can have up to n eigenpairs.

$$y_i = \sum_{j=1}^n A_{ij}x_j = \sum_{(i,j) \in E} x_j$$

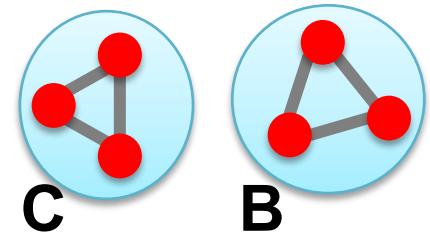
d is the largest eigenvalue of A

- G is d -regular connected, A is its adjacency matrix
- **Claim:**
 - (1) d has multiplicity of 1 (there is only 1 eigenvector associated with eigenvalue d)
 - (2) d is the largest eigenvalue of A
- **Proof:**
 - To obtain value eigval d we needed $x_i = x_j$ for every i, j
 - This means $x = c \cdot (1, 1, \dots, 1)$ for some const. c
 - **Define:** Set S = nodes i with maximum value of x_i
 - Then consider some vector y which is not a multiple of vector $(1, \dots, 1)$. So not all nodes i (with labels y_i) are in S
 - Consider some node $j \in S$ and a neighbor $i \notin S$ then node j gets a value strictly less than d
 - **So y is not eigenvector! And so d is the largest eigenvalue!**

Example: Graph on 2 components

- **What if G is not connected?**

- G has 2 components, each d -regular



- **What are some eigenvectors?**

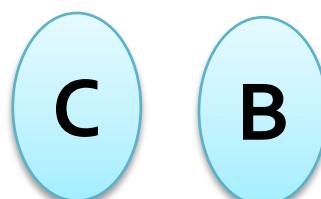
- $x = \text{Put all } 1\text{s on } C \text{ and } 0\text{s on } B \text{ or vice versa}$

- $x' = (\underbrace{1, \dots, 1}_{|C|}, \underbrace{0, \dots, 0}_{|B|})^T$ then $A \cdot x' = (d, \dots, d, 0, \dots, 0)^T$

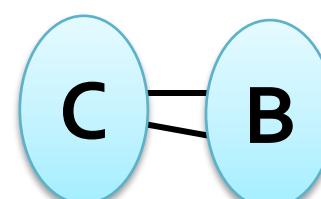
- $x'' = (0, \dots, 0, \underbrace{1, \dots, 1}_{|B|})^T$ then $A \cdot x'' = (0, \dots, 0, d, \dots, d)^T$

- And so in both cases the corresponding $\lambda = d$

- **A bit of intuition:**



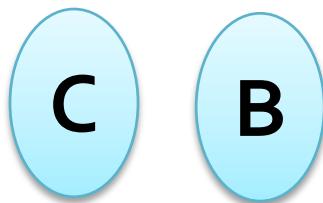
$$\lambda_n = \lambda_{n-1}$$



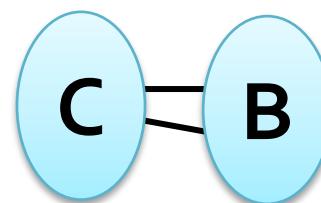
$$\lambda_n - \lambda_{n-1} \approx 0$$

2nd largest eigval.
 λ_{n-1} now has
value very close
to λ_n

More Intuition



$$\lambda_n = \lambda_{n-1}$$



$$\lambda_n - \lambda_{n-1} \approx 0$$

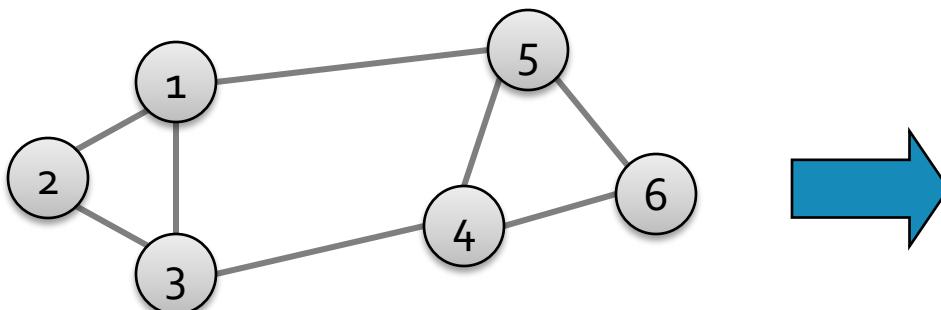
2nd largest eigval.
 λ_{n-1} now has
value very close
to λ_n

- If the graph is connected (right example) then we already know that $x_n = (1, \dots, 1)$ is an eigenvector
- Eigenvectors are orthogonal so then the components of x_{n-1} must sum to 0
 - Why? $x_n \cdot x_{n-1} = 0$ then $\sum_i x_n[i] \cdot x_{n-1}[i] = \sum_i x_n[i]$
 - x_{n-1} “splits” the nodes into two groups
 - $x_{n-1}[i] > 0$ vs. $x_{n-1}[i] < 0$
 - So we in principle could look at the eigenvector of the 2nd largest eigenvalue and declare nodes with positive label in **C** and negative label in **B**. (but there are still many details for us to figure out here)

Matrix Representations

■ Adjacency matrix (A):

- $n \times n$ matrix
- $A = [a_{ij}]$, $a_{ij} = 1$ if edge between node i and j



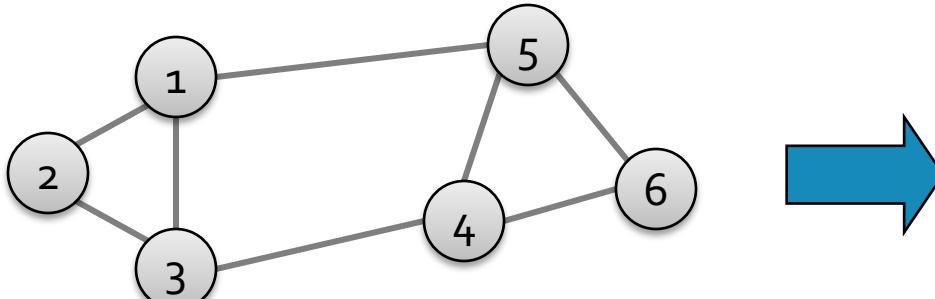
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

■ Important properties:

- Symmetric matrix
- Has n real eigenvalues
- Eigenvectors are real-valued and orthogonal

Matrix Representations

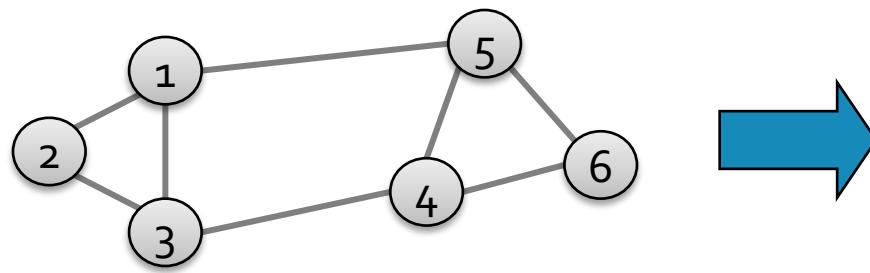
- Degree matrix (D):
 - $n \times n$ diagonal matrix
 - $D = [d_{ii}]$, d_{ii} = degree of node i



	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

Matrix Representations

- Laplacian matrix (L):
 - $n \times n$ symmetric matrix



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

$$L = D - A$$

- What is trivial eigenpair?
 - $x = (1, \dots, 1)$ then $L \cdot x = 0$ and so $\lambda = \lambda_1 = 0$
- Important properties of L :
 - Eigenvalues are non-negative real numbers
 - Eigenvectors are real (and always orthogonal)

3 Facts about the Laplacian L

- (a) All eigenvalues are ≥ 0
 - (b) $x^T L x = \sum_{ij} L_{ij} x_i x_j \geq 0$ for every x
 - (c) $L = N^T \cdot N$
 - That is, L is positive semi-definite
- **Proof: (the 3 facts are saying the same thing)**
- (c) \Rightarrow (b): $x^T L x = x^T N^T N x = (xN)^T (Nx) \geq 0$
 - As it is just the square of length of Nx
 - (b) \Rightarrow (a): Let λ be an eigenvalue of L . Then by (b) $x^T L x \geq 0$ so $x^T L x = x^T \lambda x = \lambda x^T x \Rightarrow \lambda \geq 0$
 - (a) \Rightarrow (c): is also easy! Do it yourself.

λ_2 as optimization problem

- Fact: For symmetric matrix M :

$$\lambda_2 = \min_{x : x^T w_1 = 0} \frac{x^T M x}{x^T x}$$

(w_1 is eigenvector corresponding to λ_1)

- What is the meaning of $\min x^T L x$ on G ?

- $x^T L x = \sum_{i,j=1}^n L_{ij} x_i x_j = \sum_{i,j=1}^n (D_{ij} - A_{ij}) x_i x_j$
- $= \sum_i D_{ii} x_i^2 - \sum_{(i,j) \in E} 2x_i x_j$
- $= \sum_{(i,j) \in E} (\underbrace{x_i^2 + x_j^2}_{\text{Node } i \text{ has degree } d_i} - 2x_i x_j) = \sum_{(i,j) \in E} (x_i - x_j)^2$

Node i has degree d_i . So, value x_i^2 needs to be summed up d_i times.
But each edge (i,j) has two endpoints so we need $x_i^2 + x_j^2$

See next slide
for the proof.
Deriving this is
a HW problem.

Details!

Proof:

$$\lambda_2 = \min_{x : x^T w_1 = 0} \frac{x^T M x}{x^T x}$$

- Write x in basis of eigenvecs w_1, w_2, \dots, w_n of M . So,

$$x = \sum_i^n \alpha_i w_i$$

- Then we get: $Mx = \sum_i \alpha_i \underbrace{Mw_i}_{\lambda_i w_i} = \sum_i \alpha_i \lambda_i w_i$
- So, what is $x^T M x$?

$$\begin{aligned} x^T M x &= (\underbrace{\sum_i \alpha_i w_i}_x)^T (\underbrace{\sum_i \alpha_i \lambda_i w_i}_M) = \sum_{ij} \alpha_i \lambda_j \alpha_j w_i^T w_j \\ &= \sum_i \alpha_i^2 \lambda_i w_i^T w_i = \sum_i \lambda_i \alpha_i^2 \end{aligned}$$

$= 0$ if $i \neq j$
 1 otherwise

- Want minimize this over all unit vectors w :
 $w = \min$ over choices of $(\alpha_1, \dots, \alpha_n)$ so that:
where $\sum \alpha_i^2 = 1$ (unit length) $\sum \alpha_i = 0$ (orthogonal to w_1)
- To minimize this, set $\alpha_2 = 1$ and so $\sum_i \lambda_i \alpha_i^2 = \lambda_2$

Finding x

$$\lambda_2 = \min_{x : x^T w_1 = 0} \frac{x^T M x}{x^T x}$$

■ What else do we know about x ?

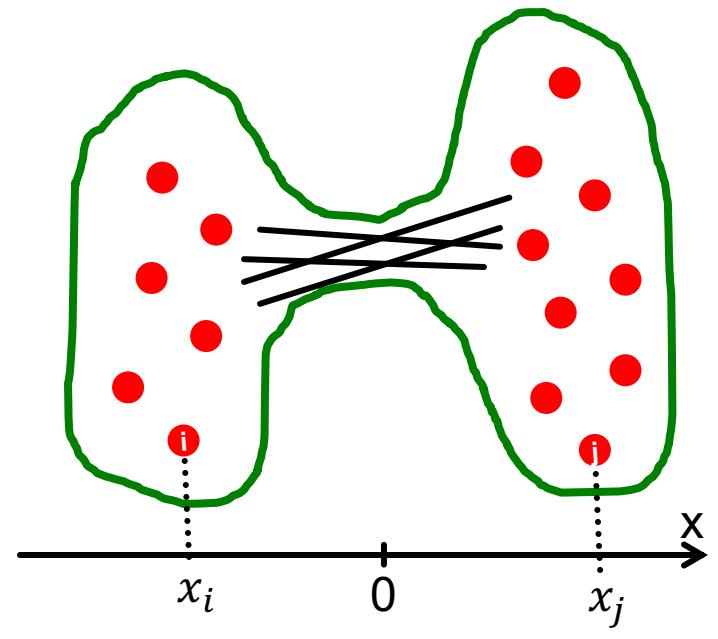
- x is unit vector: $\sum_i x_i^2 = 1$
- x is orthogonal to 1st eigenvector $(1, \dots, 1)$ thus:
 $\sum_i x_i \cdot 1 = \sum_i x_i = 0$

■ Remember:

$$\lambda_2 = \min_{\substack{\text{All labelings} \\ \text{of nodes } i \text{ so} \\ \text{that } \sum x_i = 0}} \frac{\sum_{(i,j) \in E} (x_i - x_j)^2}{\sum_i x_i^2}$$

We want to assign values x_i to nodes i such that few edges cross 0.

(we want x_i and x_j to subtract each other)



Find Optimal Cut [Fiedler'73]

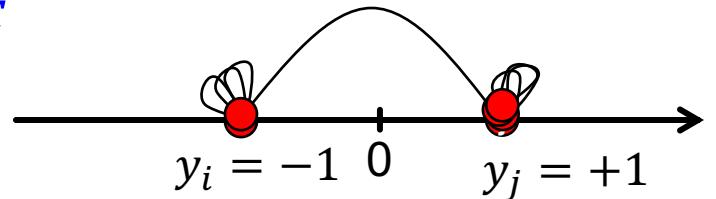
- Back to finding the optimal cut
- Express partition (A, B) as a vector

$$y_i = \begin{cases} +1 & \text{if } i \in A \\ -1 & \text{if } i \in B \end{cases}$$

- Enforce that $|A| = |B| \rightarrow \sum_i y_i = 0$
 - Equivalent to being orthogonal to the trivial eigenvector $(1, \dots, 1)$
- We can minimize the cut of the partition by finding a vector y that **minimizes**:

$$\arg \min_{y \in \{-1, +1\}^n} f(y) = \sum_{(i,j) \in E} (y_i - y_j)^2$$

Can't solve exactly. Let's relax y and allow it to take any real value.

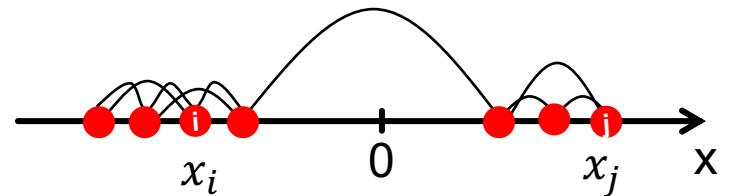


Rayleigh Theorem

$$\lambda_2 = \min_{x : x^T w_1 = 0} \frac{x^T M x}{x^T x}$$

Slide 18

$$\min_{y \in \mathbb{R}^n : \sum_i y_i = 0} f(y) = \sum_{(i,j) \in E} (y_i - y_j)^2 = y^T L y$$
$$\sum_i y_i^2 = 1$$



- $\lambda_2 = \min_y f(y)$: The minimum value of $f(y)$ is given by the 2nd smallest eigenvalue λ_2 of the Laplacian matrix L
- $x = \arg \min_y f(y)$: The optimal solution for y is given by the corresponding eigenvector x , referred to as the **Fiedler vector**
- Can use sign of x_i to determine cluster assignment of node i

Approx. Guarantee of Spectral

- Suppose there is a partition of \mathbf{G} into \mathbf{A} and \mathbf{B} where $|A| \leq |B|$, s.t. “conductance” of the cut (A, B) is $\alpha = \frac{(\# \text{ edges from } A \text{ to } B)}{|A|}$ then $\lambda_2 \leq 2\alpha$

Note: $|A| < |B|$

- This is the approximation guarantee of the spectral clustering: Spectral finds a cut that has at most **twice the conductance** as the optimal one of conductance α .

Proof:

- Let: $a = |A|$, $b = |B|$ and $e = \# \text{ edges from } \mathbf{A} \text{ to } \mathbf{B}$
- Enough to choose some x_i based on \mathbf{A} and \mathbf{B} such that:

$$\lambda_2 \leq \underbrace{\frac{\sum(x_i - x_j)^2}{\sum_i x_i^2}}_{\lambda_2 \text{ is only smaller}} \leq 2\alpha \quad (\text{while also } \sum_i x_i = 0)$$

Approx. Guarantee of Spectral

■ Proof (continued):

■ 1) Let's set: $x_i = \begin{cases} -\frac{1}{a} & \text{if } i \in A \\ +\frac{1}{b} & \text{if } i \in B \end{cases}$

Note: $|A| < |B|$

■ Let's quickly verify that $\sum_i x_i = 0$: $a\left(-\frac{1}{a}\right) + b\left(\frac{1}{b}\right) = 0$

■ 2) Then: $\frac{\sum(x_i - x_j)^2}{\sum_i x_i^2} = \frac{\sum_{i \in A, j \in B} \left(\frac{1}{b} + \frac{1}{a}\right)^2}{a\left(-\frac{1}{a}\right)^2 + b\left(\frac{1}{b}\right)^2} = \frac{e \cdot \left(\frac{1}{a} + \frac{1}{b}\right)^2}{\frac{1}{a} + \frac{1}{b}} =$

$$e \left(\frac{1}{a} + \frac{1}{b}\right) \leq e \left(\frac{1}{a} + \frac{1}{a}\right) = e \frac{2}{a} \leq 2\alpha$$

e ... number of edges between A and B

Which proves that the cost achieved by spectral is better than twice the OPT cost

Approx. Guarantee of Spectral

■ Putting it all together: The Cheeger inequality

$$\frac{\alpha^2}{2k_{max}} \leq \lambda_2 \leq 2\alpha$$

- where k_{max} is the maximum node degree in the graph
 - Note we only provide the 1st part: $\lambda_2 \leq 2\alpha$
 - We did not prove $\frac{\alpha^2}{2k_{max}} \leq \lambda_2$
- Overall this always certifies that λ_2 always gives a useful bound

So far...

- **How to define a “good” partition of a graph?**
 - Minimize a given graph cut criterion
- **How to efficiently identify such a partition?**
 - Approximate using information provided by the eigenvalues and eigenvectors of a graph
- **Spectral Clustering**

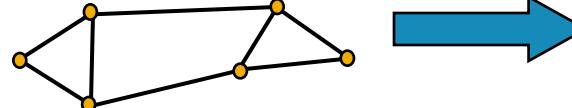
Spectral Clustering Algorithm

- Three basic stages:
 - 1) Pre-processing
 - Construct a matrix representation of the graph
 - 2) Decomposition
 - Compute eigenvalues and eigenvectors of the matrix
 - Map each point to a lower-dimensional representation based on one or more eigenvectors
 - 3) Grouping
 - Assign points to two or more clusters, based on the new representation

Spectral Partitioning Algorithm

■ 1) Pre-processing:

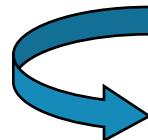
- Build Laplacian matrix L of the graph



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

■ 2) Decomposition:

- Find eigenvalues λ and eigenvectors x of the matrix L



$$\lambda =$$

0.0
1.0
3.0
3.0
4.0
5.0

$$X =$$

0.4	0.3	-0.5	-0.2	-0.4	-0.5
0.4	0.6	0.4	-0.4	0.4	0.0
0.4	0.3	0.1	0.6	-0.4	0.5
0.4	-0.3	0.1	0.6	0.4	-0.5
0.4	-0.3	-0.5	-0.2	0.4	0.5
0.4	-0.6	0.4	-0.4	-0.4	0.0

- Map vertices to corresponding components of x_2

1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

How do we now find the clusters?

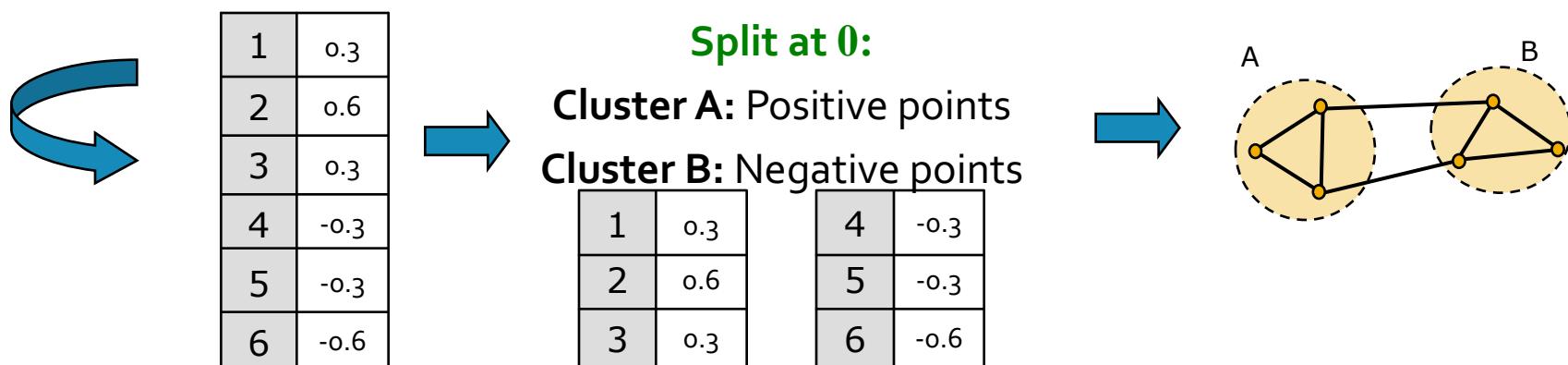
Spectral Partitioning

■ 3) Grouping:

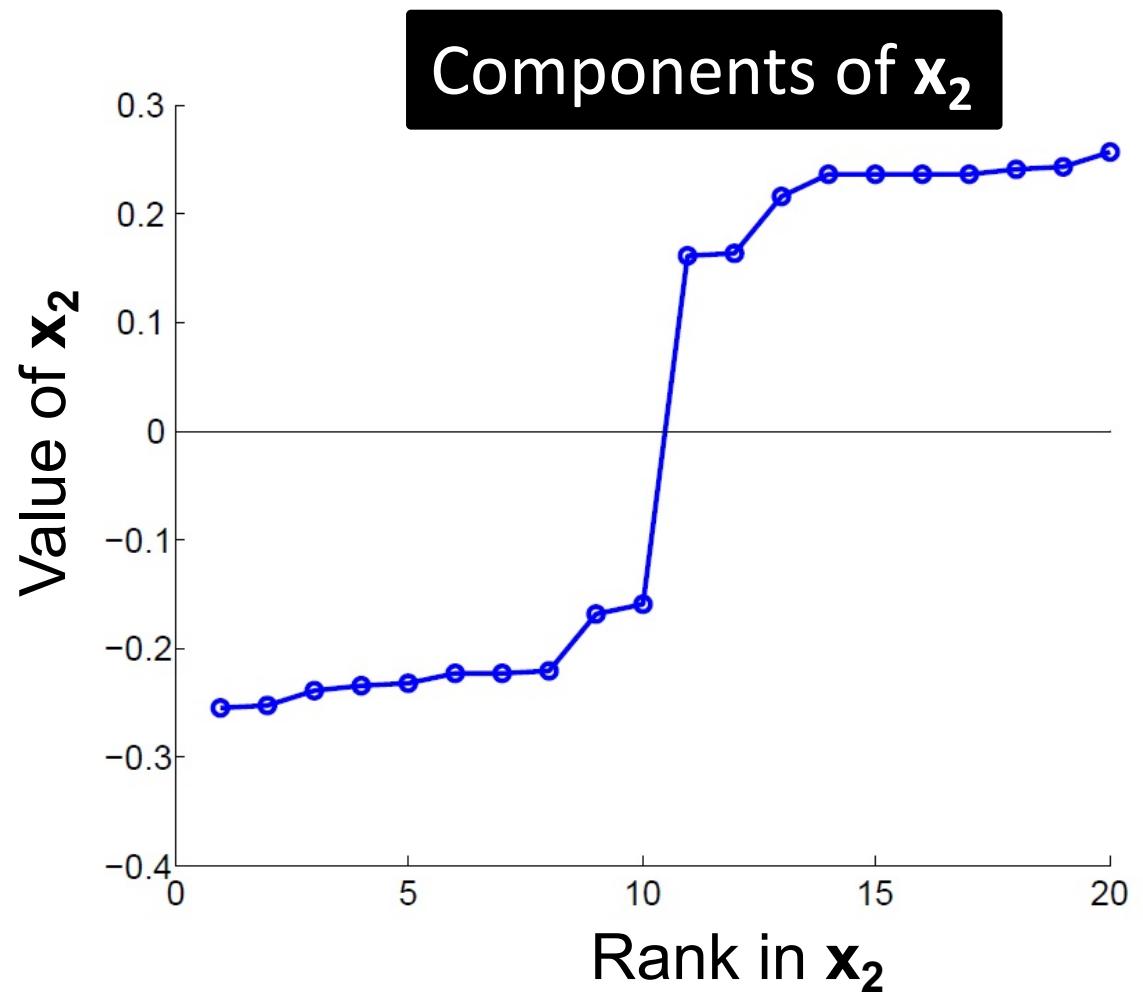
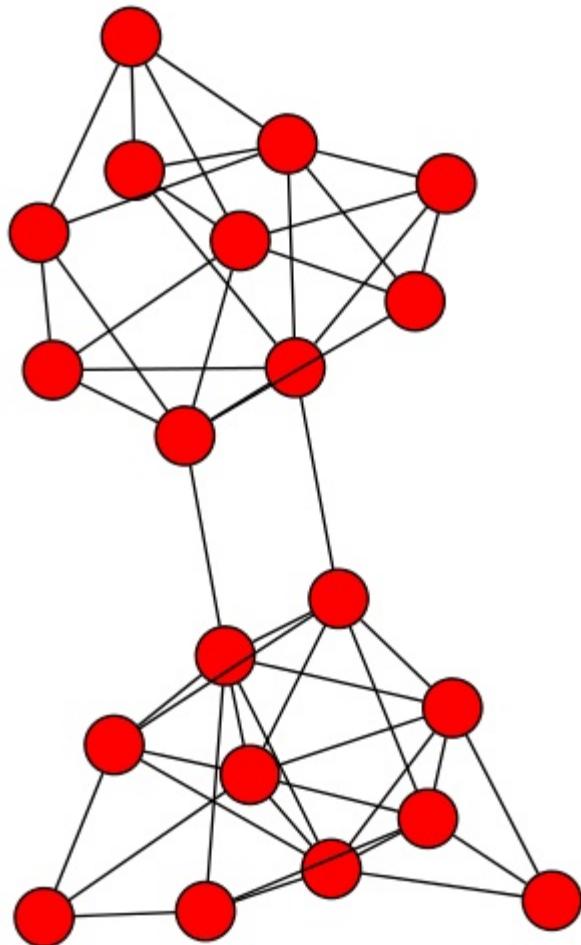
- Sort components of reduced 1-dimensional vector
- Identify clusters by splitting the sorted vector in two

■ How to choose a splitting point?

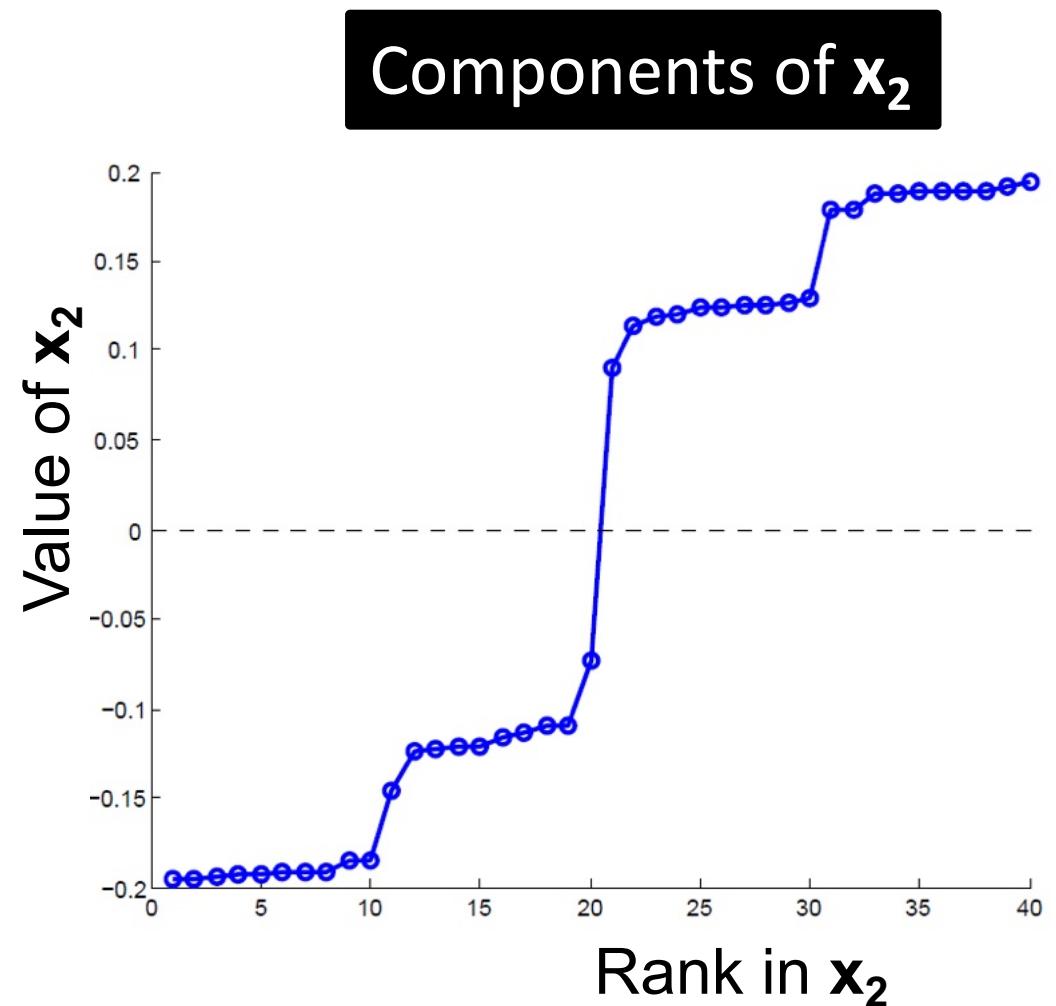
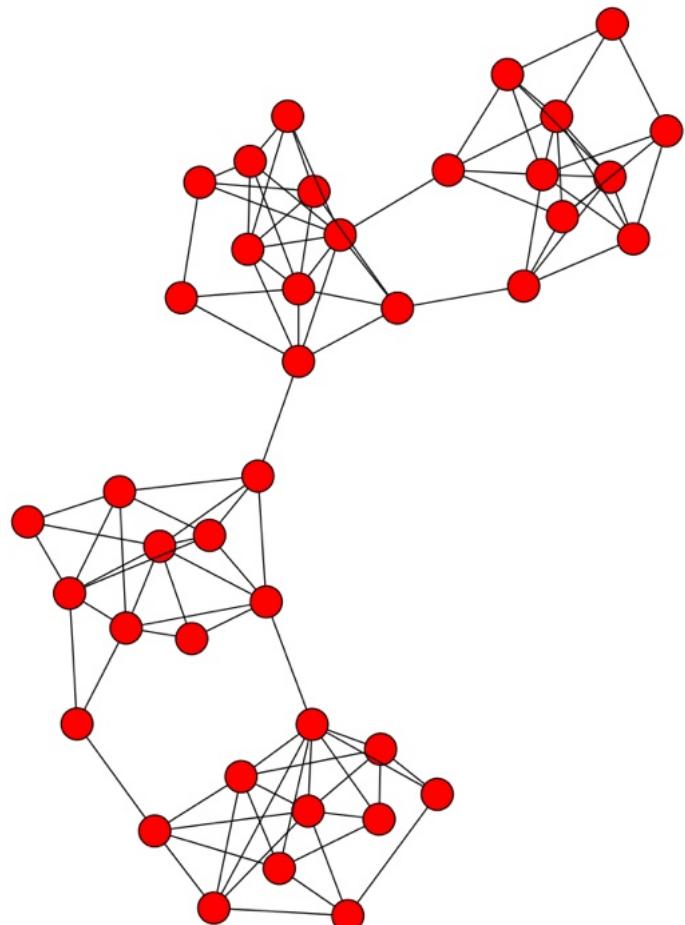
- Naïve approaches:
 - Split at 0 or median value
- More expensive approaches:
 - Attempt to minimize normalized cut in 1-dimension (sweep over ordering of nodes induced by the eigenvector)



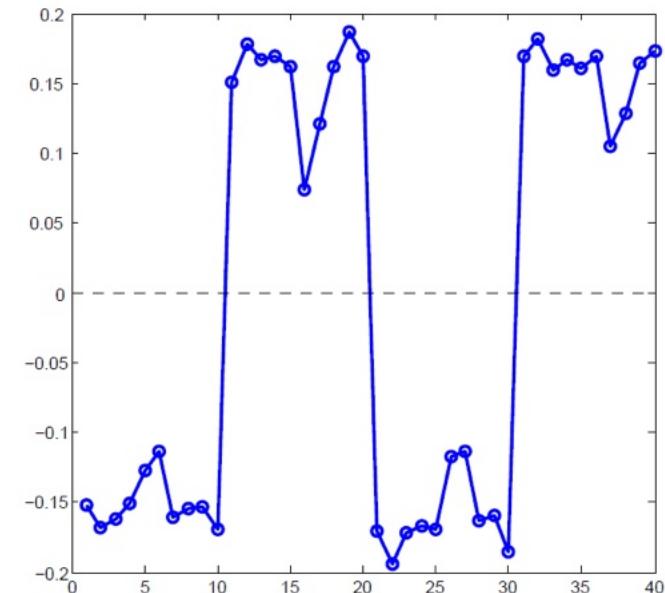
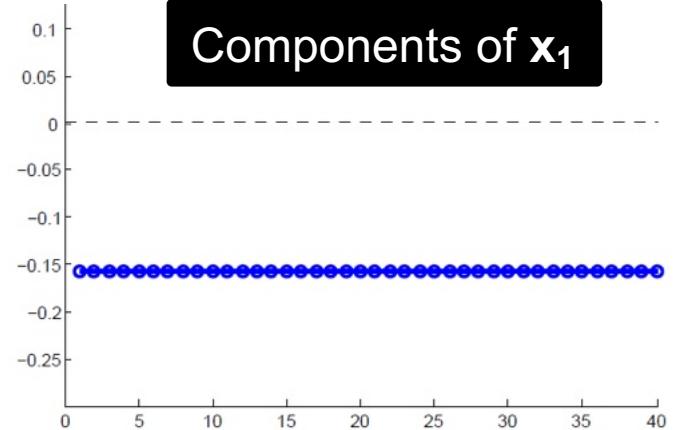
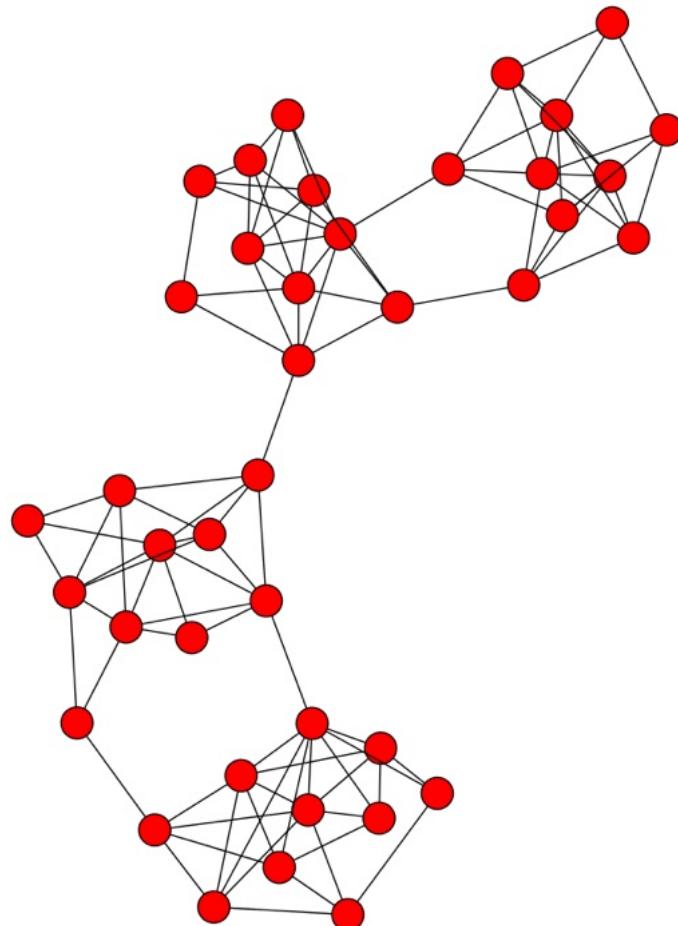
Example: Spectral Partitioning



Example: Spectral Partitioning



Example: Spectral partitioning



k-Way Spectral Clustering

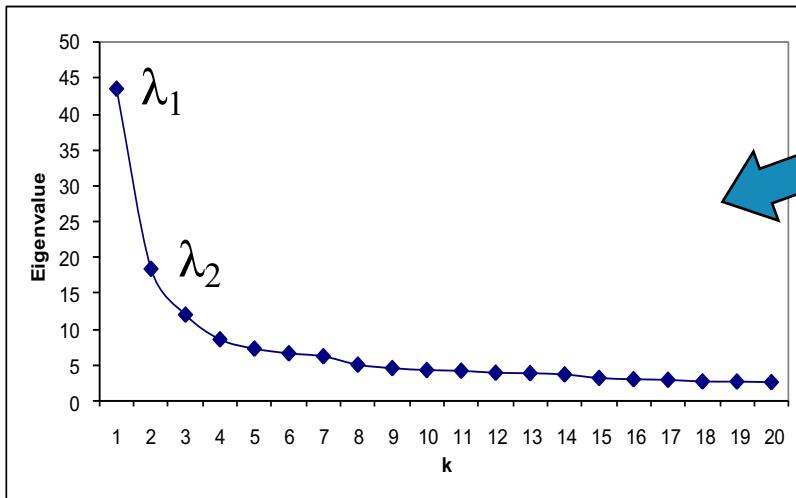
- How do we partition a graph into k clusters?
- Two basic approaches:
 - Recursive bi-partitioning [Hagen et al., '92]
 - Recursively apply bi-partitioning algorithm in a hierarchical divisive manner
 - Disadvantages: Inefficient, unstable
 - Cluster multiple eigenvectors [Shi-Malik, '00]
 - Build a reduced space from multiple eigenvectors
 - Each node is now represented by k numbers
 - We then cluster (apply k-means) the nodes based on their k -dim representation
 - Commonly used in recent papers
 - A preferable approach...

Why use multiple eigenvectors?

- **Approximates the optimal cut** [Shi-Malik, '00]
 - Can be used to approximate optimal k -way normalized cut
- **Emphasizes cohesive clusters**
 - Increases the unevenness in the distribution of the data
 - Associations between similar points are amplified, associations between dissimilar points are attenuated
 - The data begins to “approximate a clustering”
- **Well-separated space**
 - Transforms data to a new “embedded space”, consisting of k orthogonal basis vectors
- Multiple eigenvectors prevent instability due to information loss

How to select k?

- **Eigengap:**
 - The difference between two consecutive eigenvalues
- **Most stable clustering is generally given by the value k that maximizes eigengap Δ_k :**
$$\Delta_k = |\lambda_k - \lambda_{k-1}|$$
- **Example:**

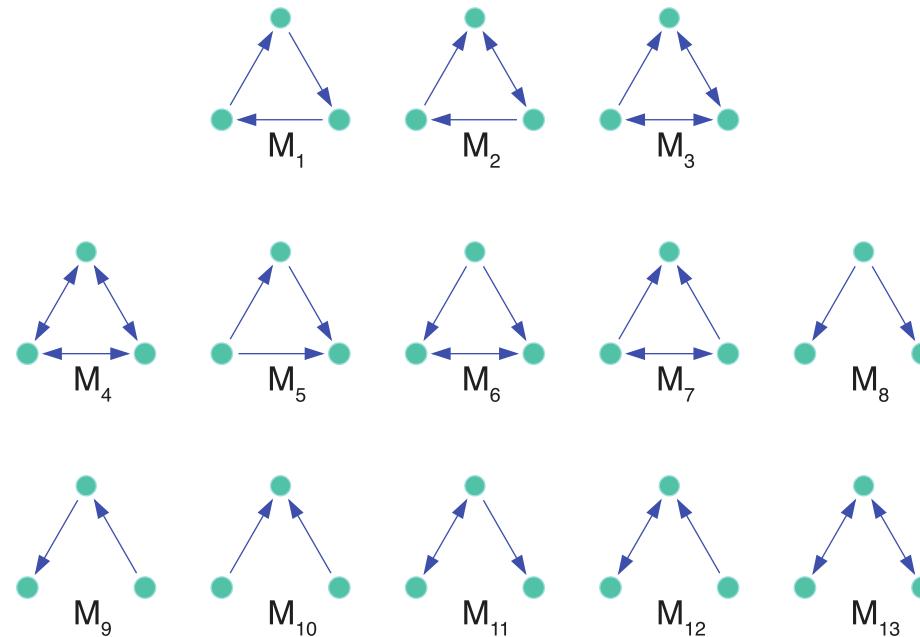


$\max \Delta_k = |\lambda_2 - \lambda_1|$
⇒ Choose
 $k = 2$

Motif-Based Spectral Clustering

Motif-based spectral clustering

- What if we want our clustering based on other patterns (not edges)?

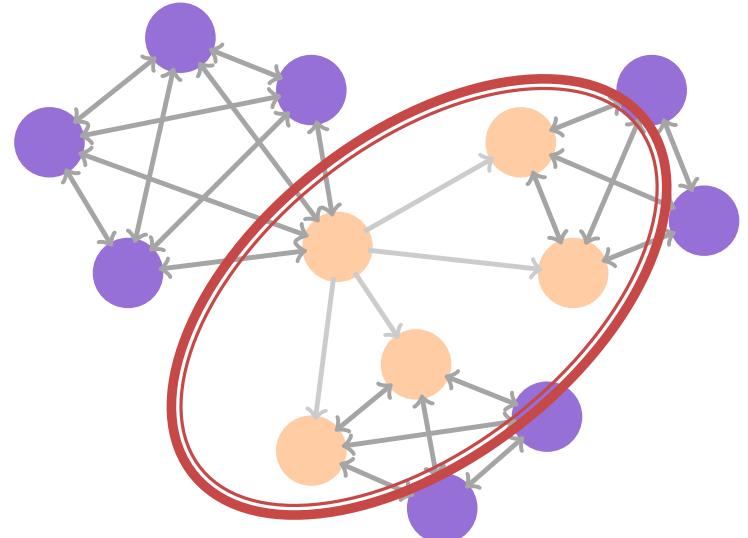
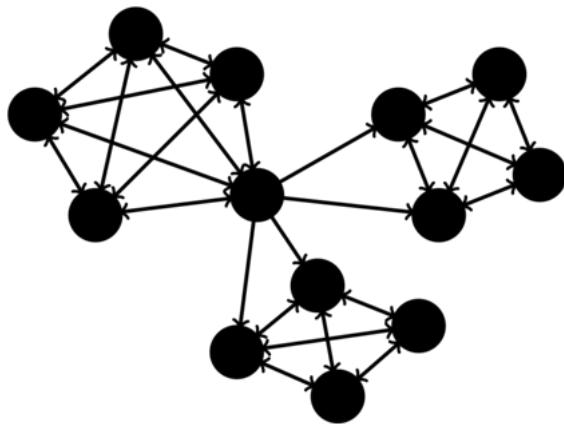


Small subgraphs (motifs, graphlets) are building blocks of networks [Milo et al., '02]

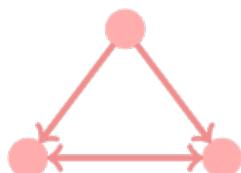
Topic 1: Modules of Motifs

Find modules based on motifs!

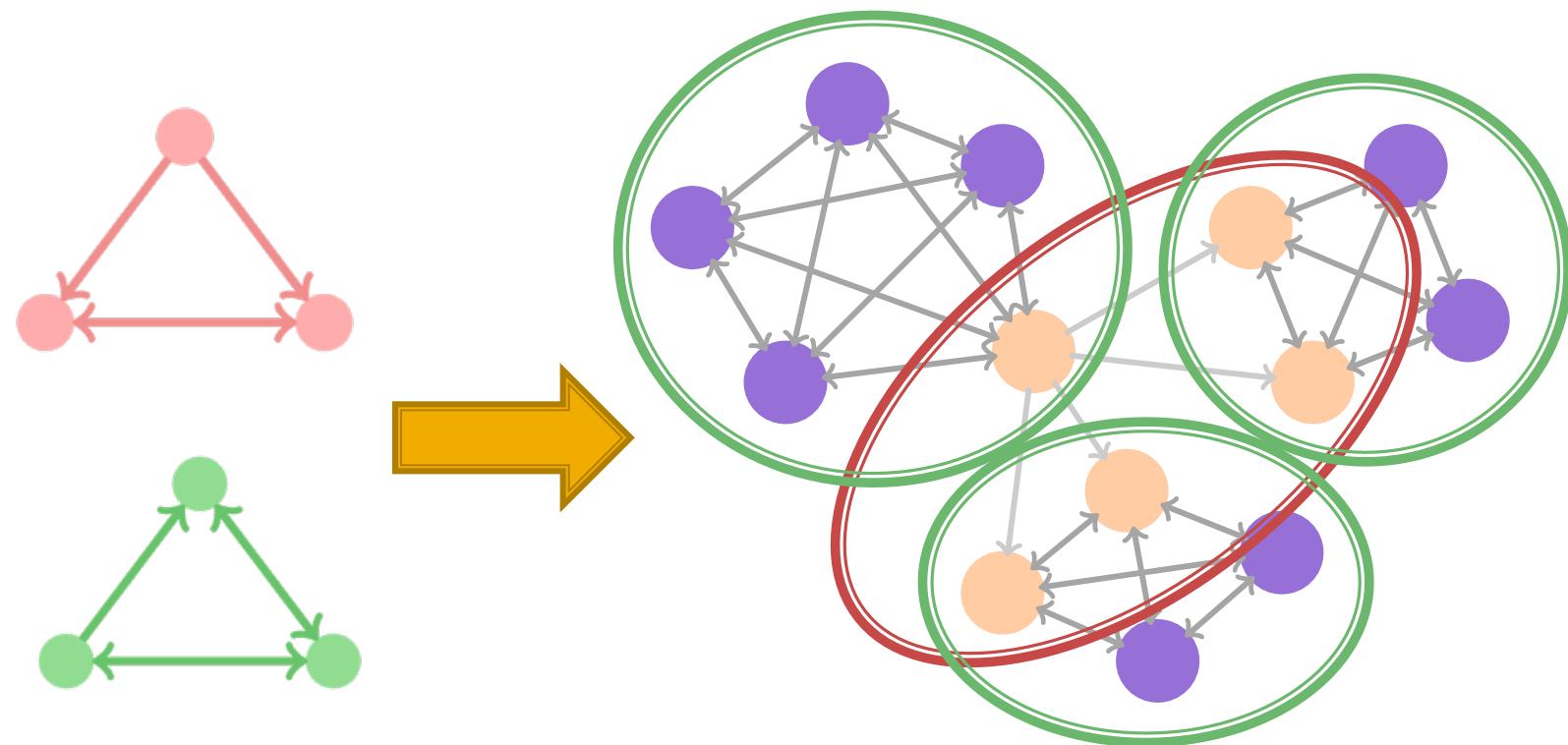
Network:



Motif:



Modules of Motifs

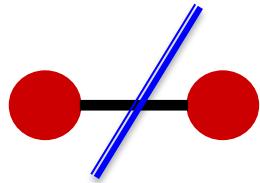


Different motifs reveal different
modular structures!

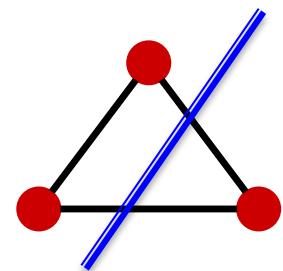
Defining Motif Conductance

Generalize Cut and Volume to motifs:

edges cut



motifs cut



$vol(S) = \#(\text{edge end-points in } S)$

$vol_M(S) = \#(\text{motif end-points in } S)$

$$\phi(S) = \frac{\#(\text{edges cut})}{vol(S)}$$



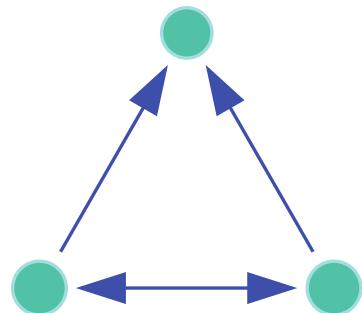
$$\phi(S) = \frac{\#(\text{motifs cut})}{vol_M(S)}$$

[Benson, Gleich, Leskovec, Science, 2016]

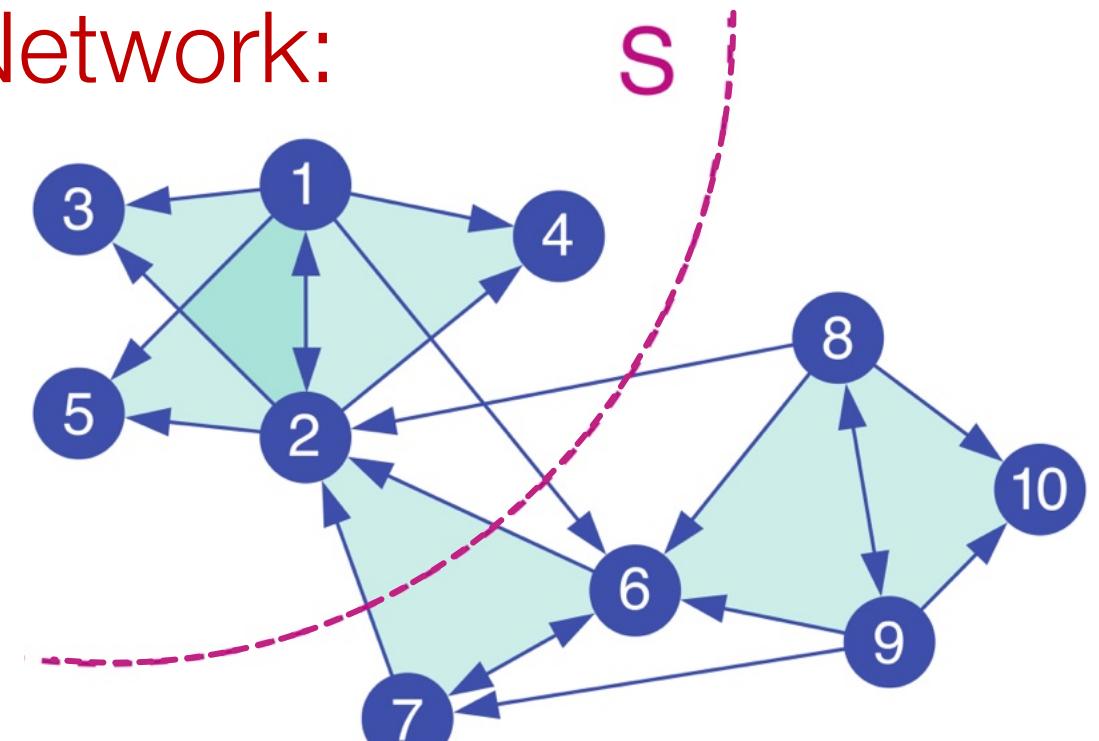
Jure Leskovec, Stanford CS224W: Analysis of Networks, <http://cs224w.stanford.edu>

Motif Conductance: Example

Motif:



Network:

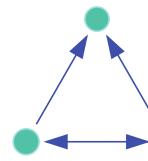


$$\phi_M(S) = \frac{\text{motifs cut}}{\text{motif volume}}$$

Higher-order Partitioning

How do we find clusters of motifs?

- Given a motif M and a graph G
- Find a set of nodes S that minimizes motif conductance

$$\phi_M(S) = \frac{\text{motifs cut}}{\text{motif volume}}$$


Bad news: Finding set S with the minimal motif conductance is NP-hard!

Motif Spectral Clustering

Solution: Motif Spectral Clustering

- Input: Graph G and motif M
- Using G form a new weighted graph $W^{(M)}$
- Apply spectral clustering on $W^{(M)}$
- Output the clusters

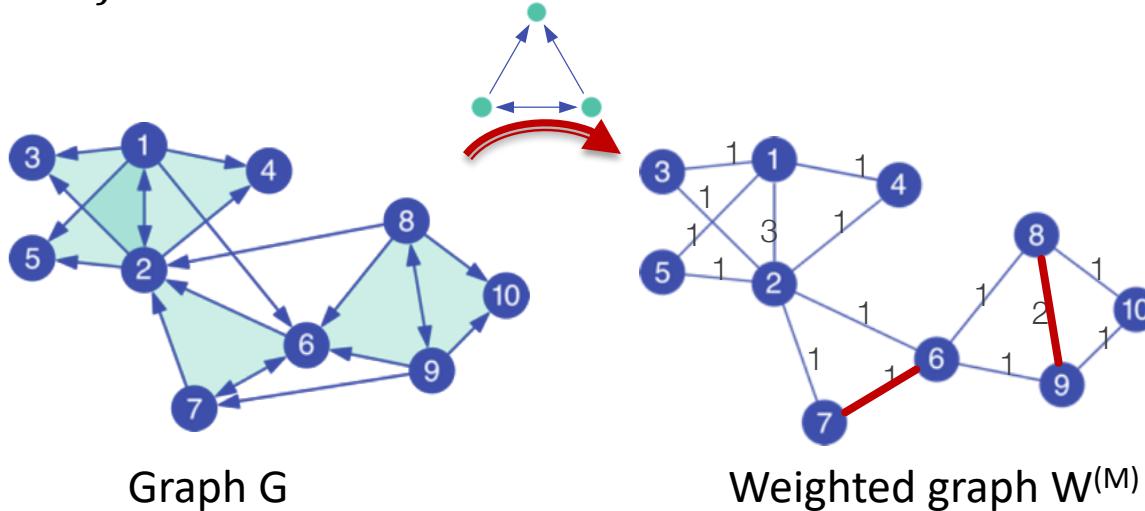
Theorem: Resulting clusters will obtain near optimal motif conductance

Optimizing Motif Conductance

■ Three steps:

■ 1) Pre-processing

- $W_{ij}^{(M)}$ = # times edge (i, j) participates in the motif M



See lecture 5 on motifs and the ESU algorithm for enumerating them

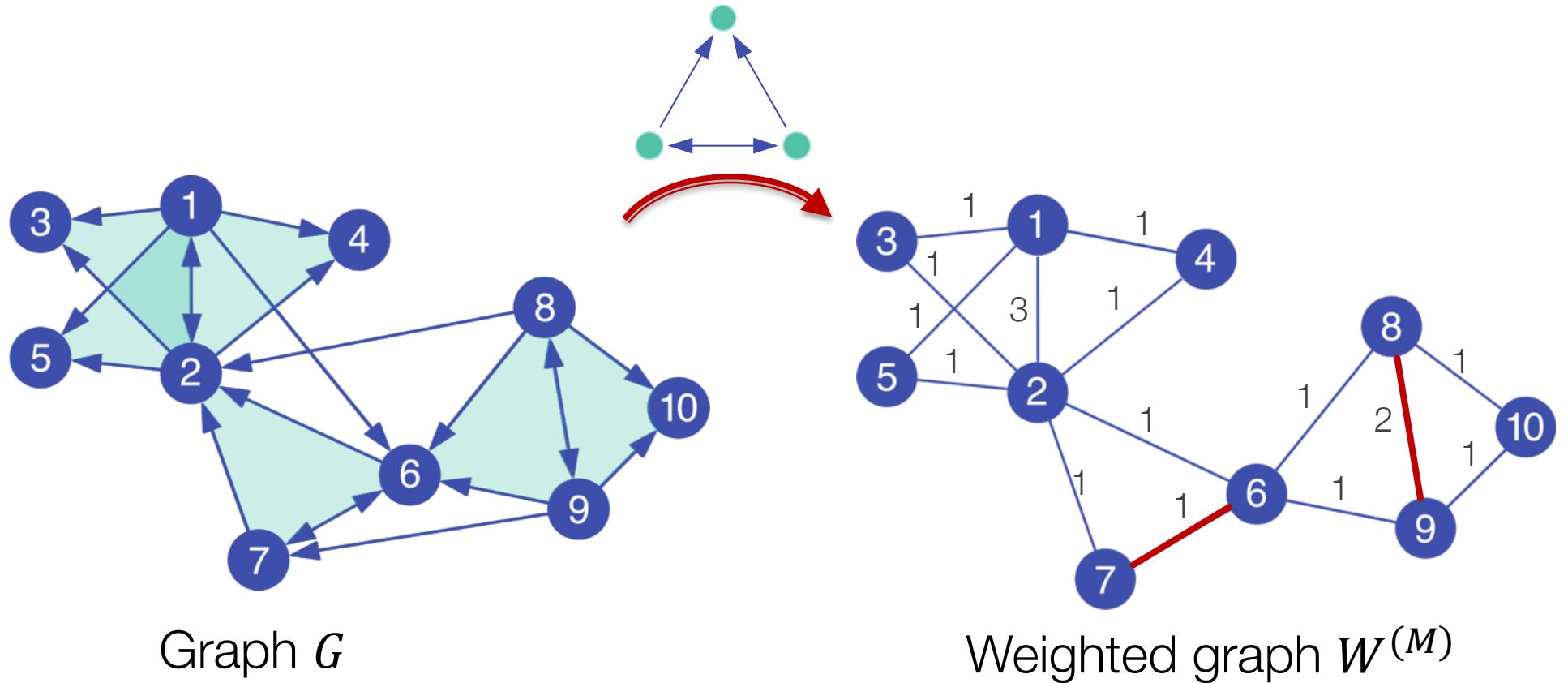
■ 2) Decomposition

- Use standard spectral clustering (but on $W^{(M)}$)

■ 3) Grouping

- Same as standard spectral clustering

Motif Spectral Clustering

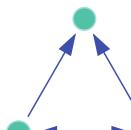


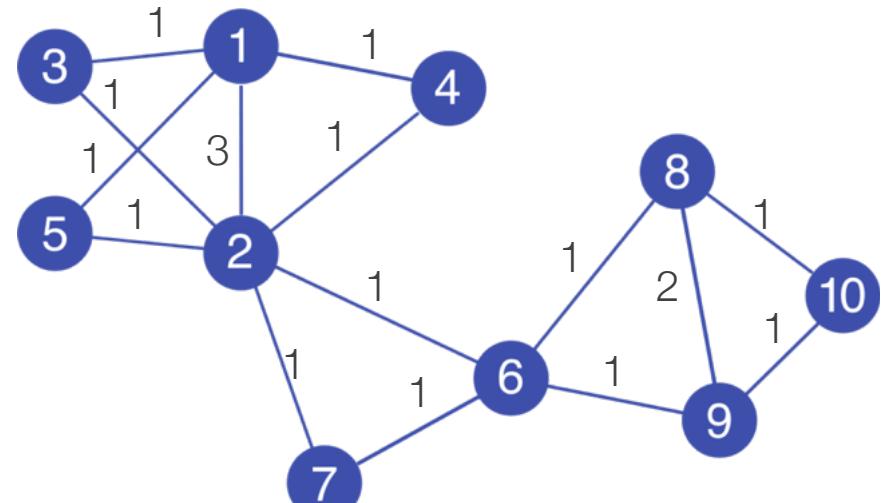
$W_{ij}^{(M)} = \# \text{ of times edge } (i,j) \text{ participates in motif M}$

Motif Spectral Clustering

Insight:

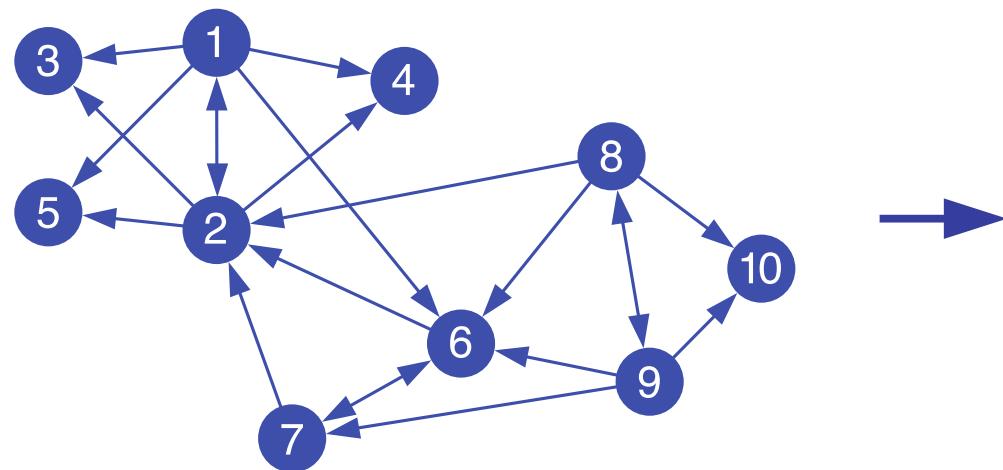
Spectral clustering on weighted graph $W^{(M)}$ finds clusters of low motif conductance:

$$\phi_M(S) = \frac{\text{motifs cut}}{\text{motif volume}}$$




$W_{ij}^{(M)} = \# \text{ of times edge } (i,j) \text{ participates in motif } M$

Step 1: Create W



Nodes	1	2	3	4	5	6	7	8	9	10
1	0	3	1	1	1	0	0	0	0	0
2	3	0	1	1	1	1	1	0	0	0
3	1	1	0	0	0	0	0	0	0	0
4	1	1	0	0	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	1	1	1	0
7	0	1	0	0	0	1	0	0	0	0
8	0	0	0	0	1	0	0	2	1	0
9	0	0	0	0	1	0	2	0	1	0
10	0	0	0	0	0	0	1	1	0	0

Step 1: Given motif M . Form weighted graph $W^{(M)}$

Step 2: Apply Spectral Clust to $W^{(M)}$

Step 2: Apply spectral clustering:

Compute Fiedler vector $f^{(M)}$

associated with λ_2 of the Laplacian of $L^{(M)}$

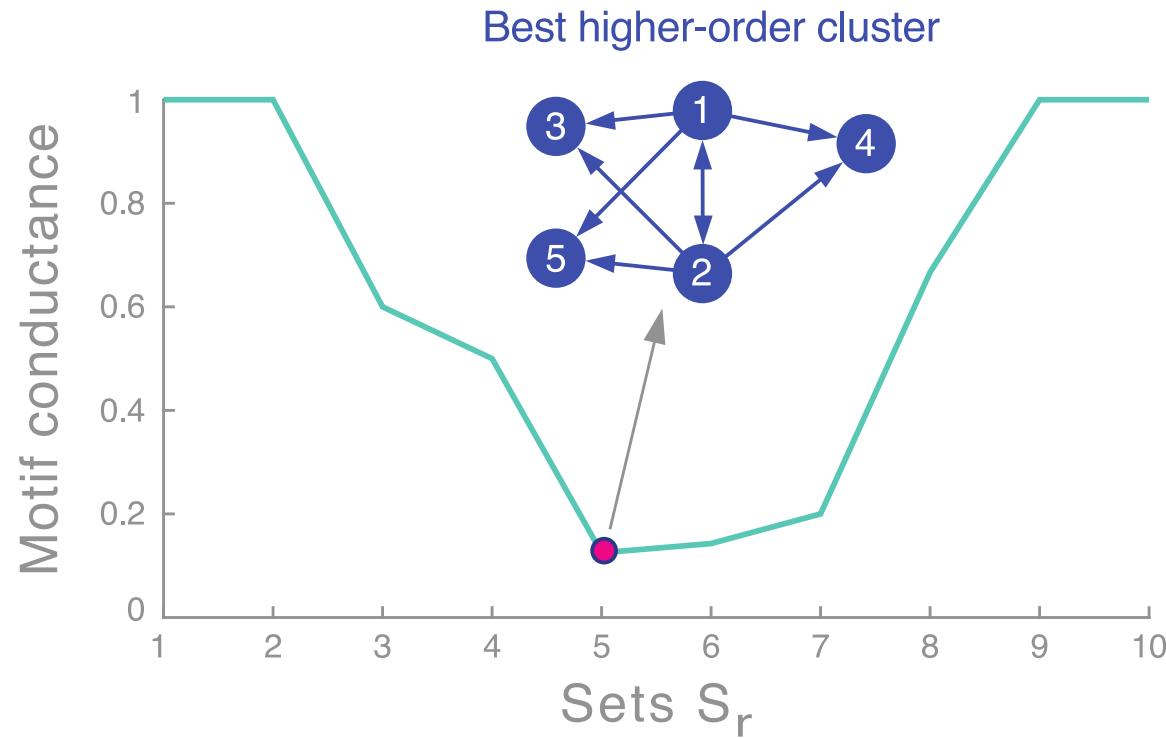
Set: $L^{(M)} = D - W^{(M)}$

Degree matrix
$$D_{ii} = \sum_j W_{ij}^{(M)}$$

Compute: $L^{(M)} f^{(M)} = \lambda_2 f^{(M)}$

Use $f^{(M)}$ to identify communities

Step 3: Grouping (Sweep Procedure)



Step 3: Sort nodes by values in $f^{(M)}$: f_1, f_2, \dots, f_n

Let $S_r = \{f_1, \dots, f_r\}$ and compute the motif conductance of each S_r

Motif Cheeger Inequality

Theorem: The algorithm finds a set of nodes S for which

$$\phi_M(S) \leq 4\sqrt{\phi_M^*}$$

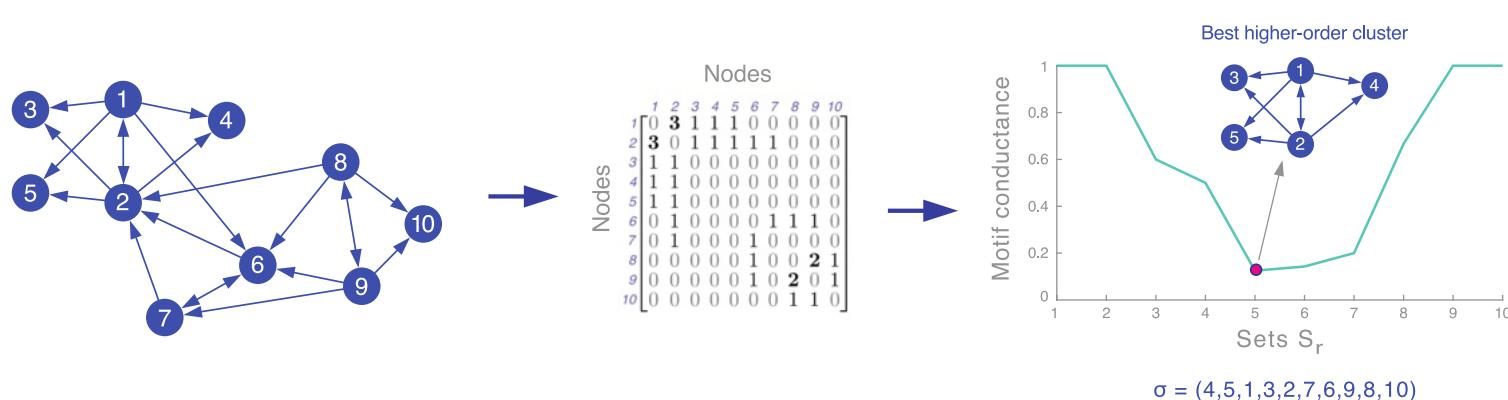
$\phi_M(S)$... motif conductance of S found by our algorithm

ϕ_M^* ... motif conductance of optimal set S^*

In other words: Clusters S found by the method are provably **near optimal**

Summary

- Generalization of community detection to higher-order structures
- Motif-conductance objective admits a motif Cheeger inequality
- Simple, fast, and scalable:



Two Examples

1) We don't know a motif of interest

- Food webs and new applications

2) We know the motif of interest

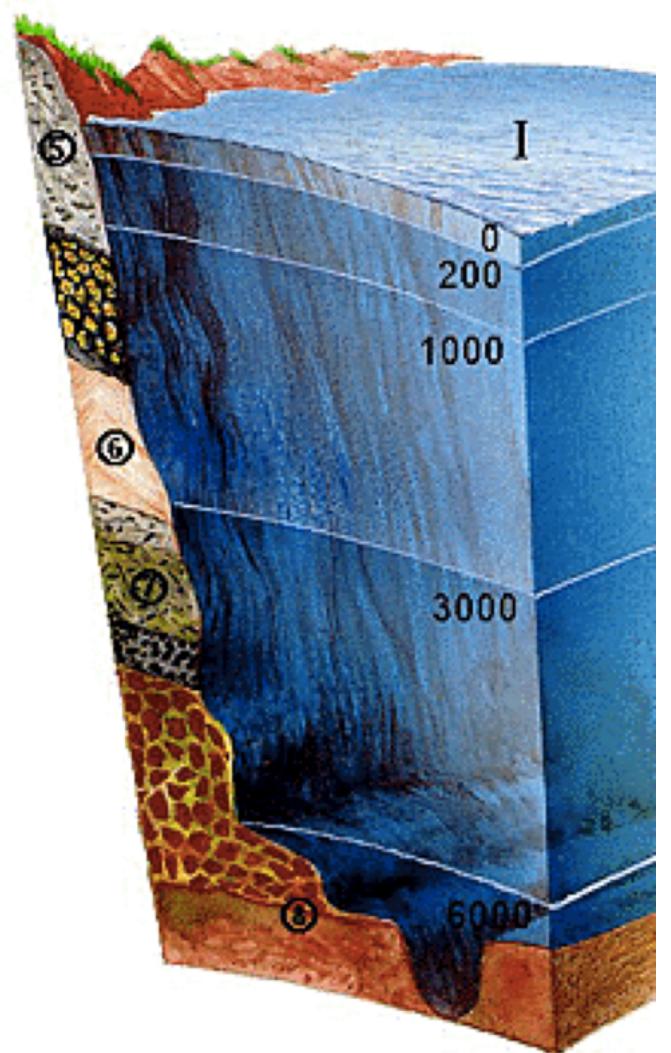
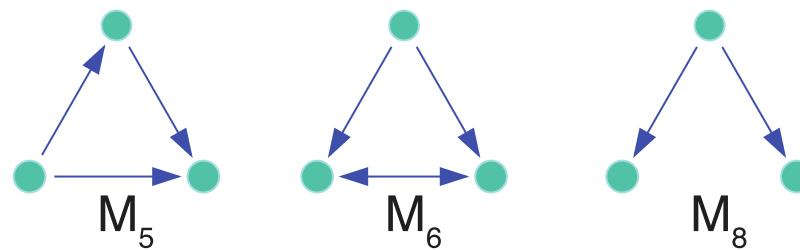
- Regulatory transcription networks, connectome, social networks

Application 1: Food webs

Florida Bay food web:

- **Nodes:** species in the ecosystem
- **Edges:** carbon exchange (who eats whom)

Different motifs capture different energy flow patterns:



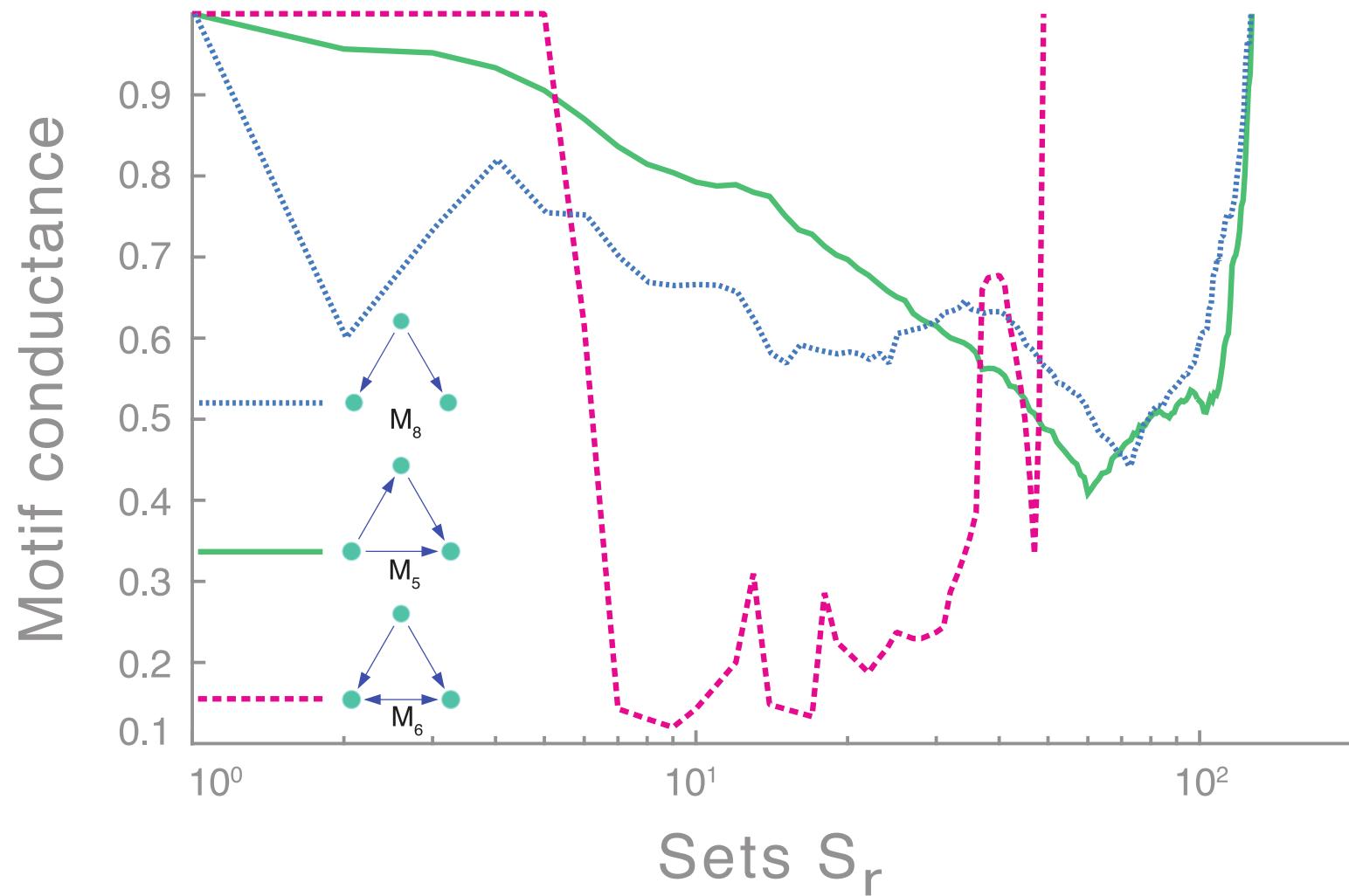
Florida Bay Food Web

Which motif organizes the food web?

Approach:

- Run motif spectral clustering separately for each of the 13 motifs
- Examine the **Sweep profile (next slide)** to see which motif gives best clusters

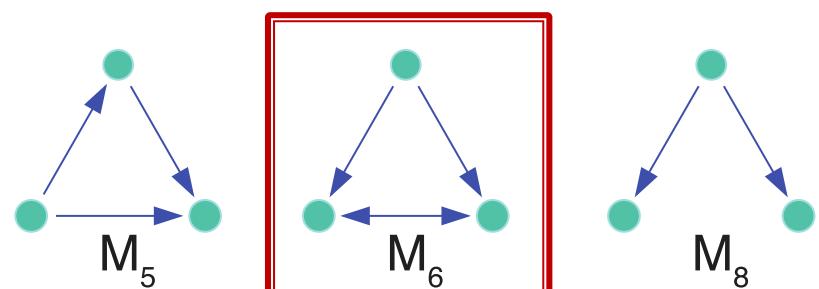
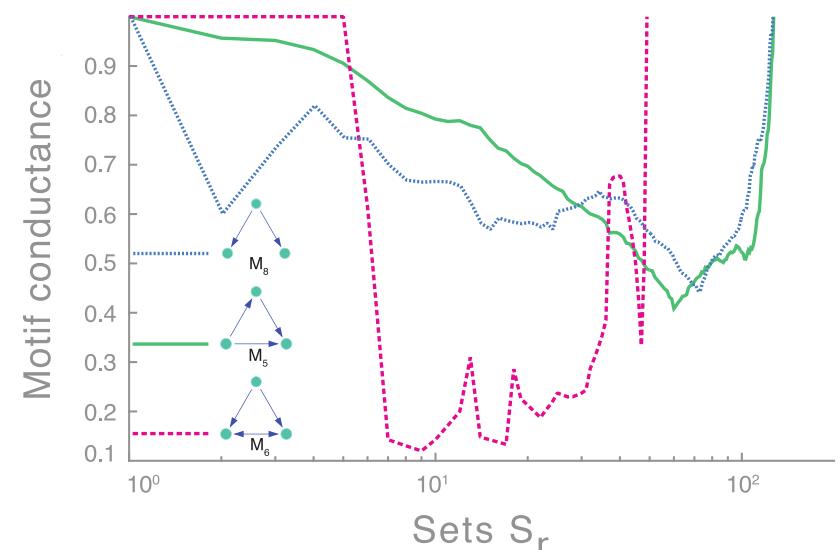
Florida Bay Food Web



Food Web: Observations

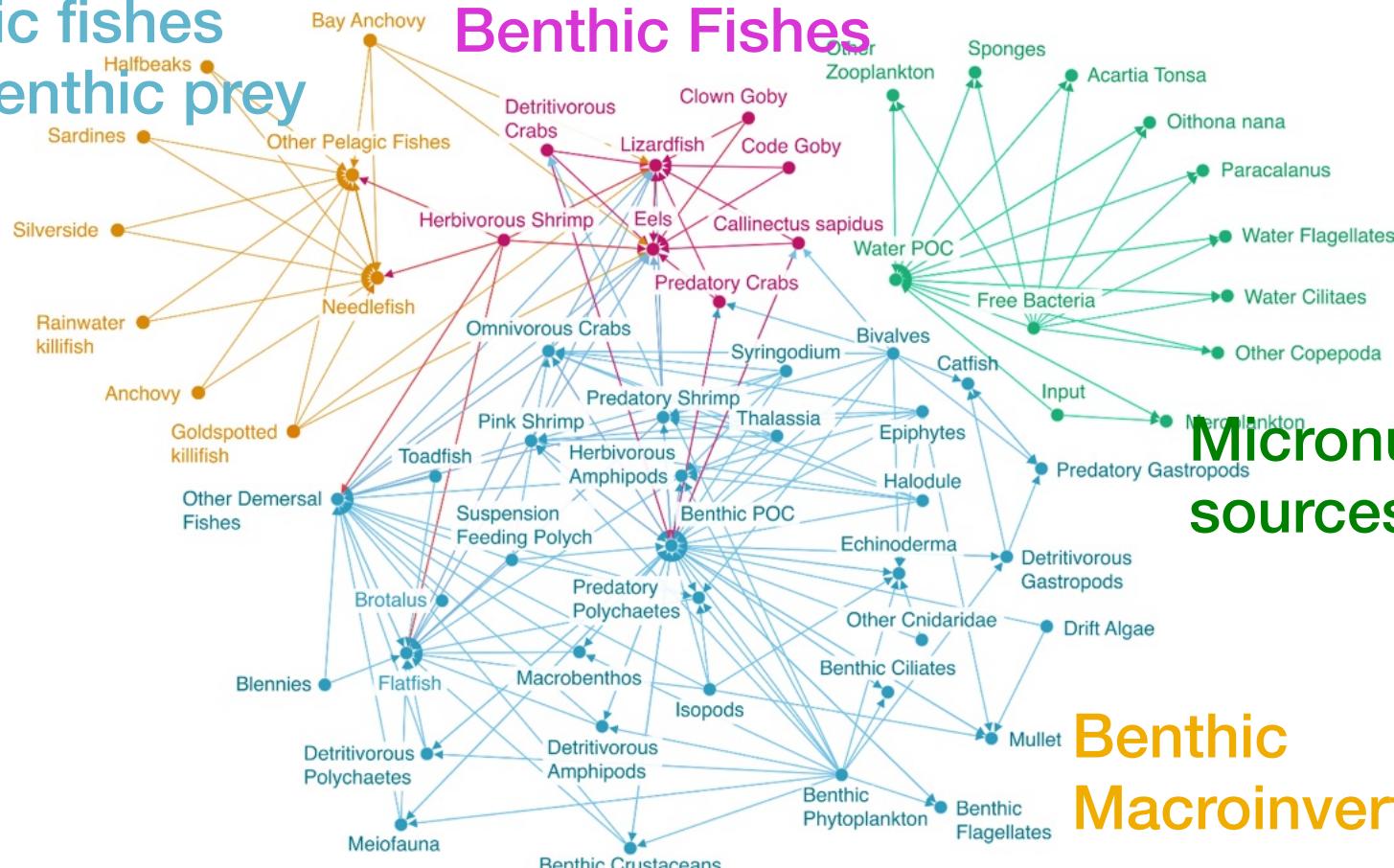
Observation:
Network organizes
based on motif M_6 (but
not M_5 or M_8)

- There exist good cuts for M_6 but not for M_5 or M_8



Food Web: Clusters

Pelagic fishes
and benthic prey

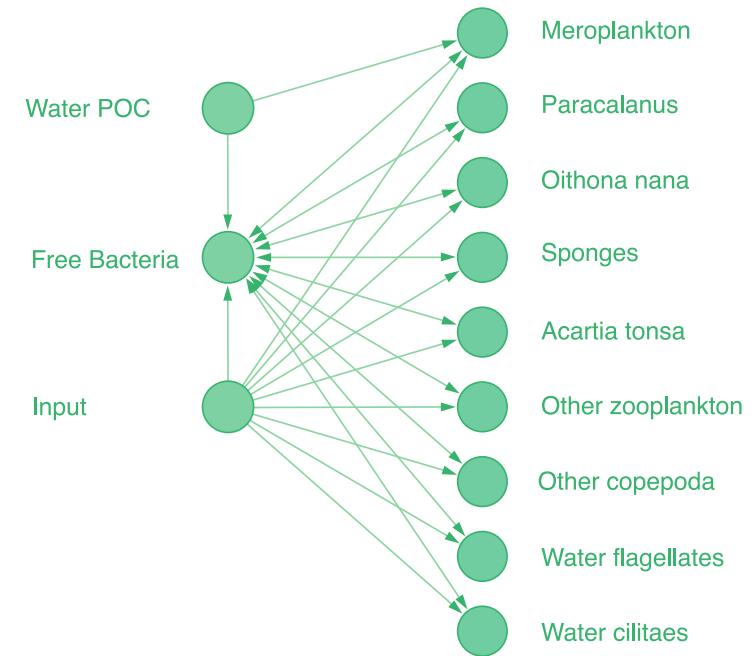
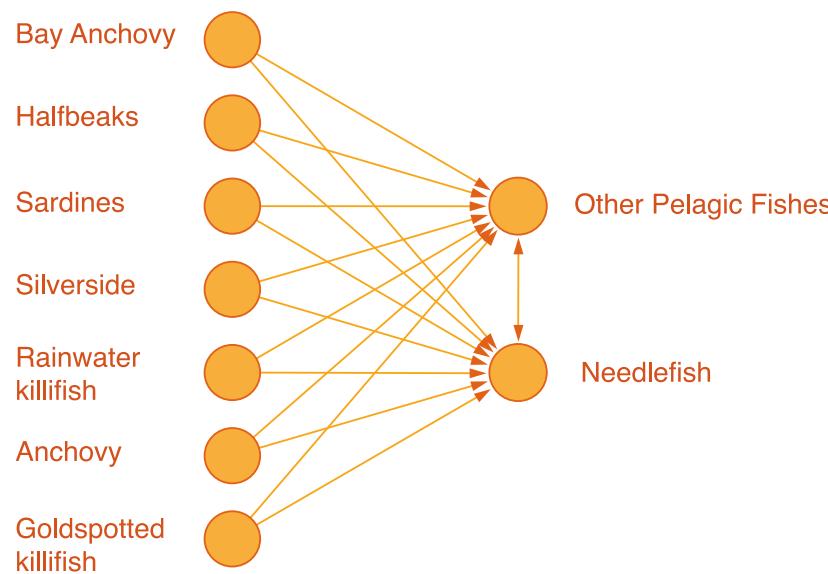


Micronutrient
sources

Benthic
Macroinvertebrates

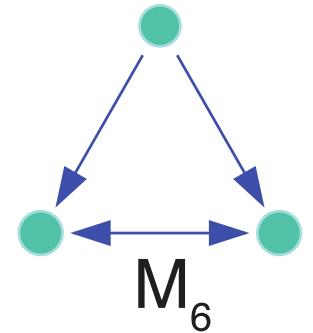
M_6 reveals known aquatic layers with
higher accuracy (84% vs. 65%)

Structure of Aquatic Layers



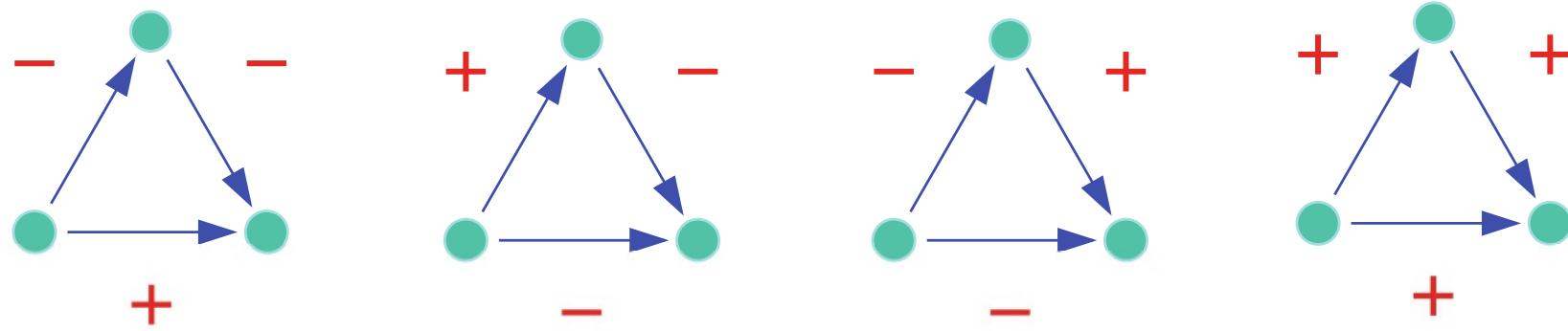
Aquatic layers organize based on M_6

- Many instances of M_6 inside
- Few instances of M_6 cross



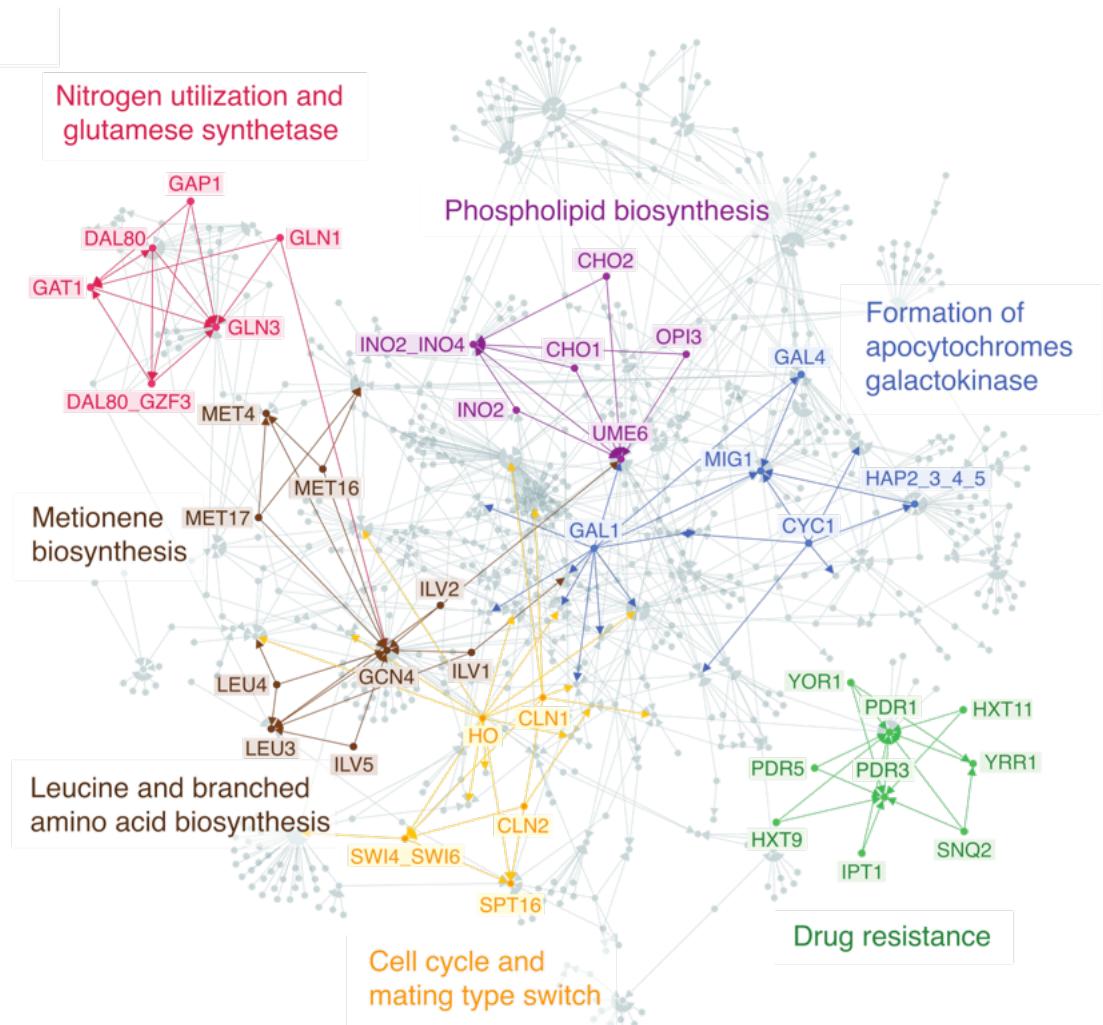
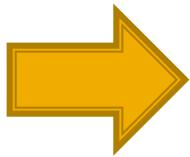
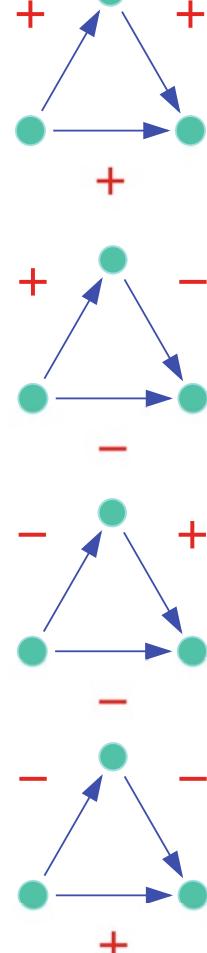
(2) Gene Regulatory Networks

- Nodes are groups of genes in mRNA
- Edges are directed transcriptional regulation relationships



- The “feedforward loop” motif represents biological function [Alon ‘07]

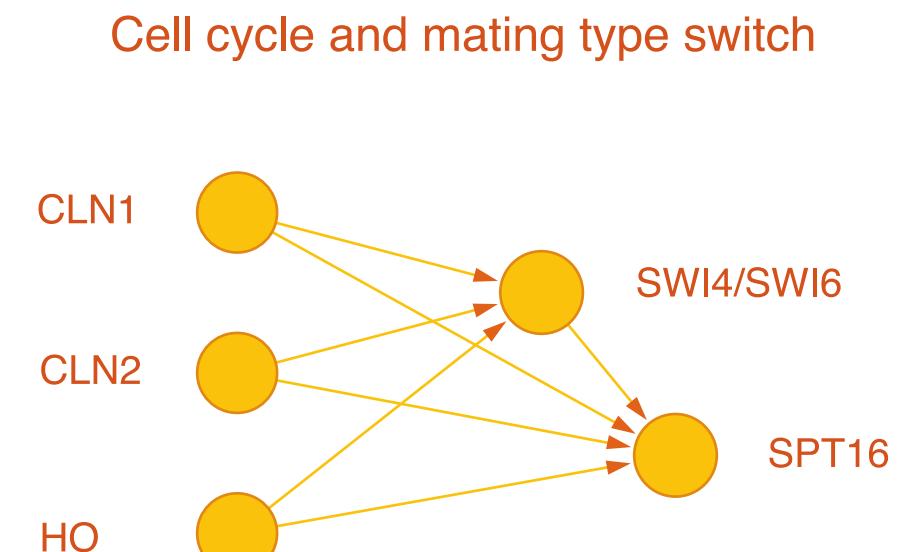
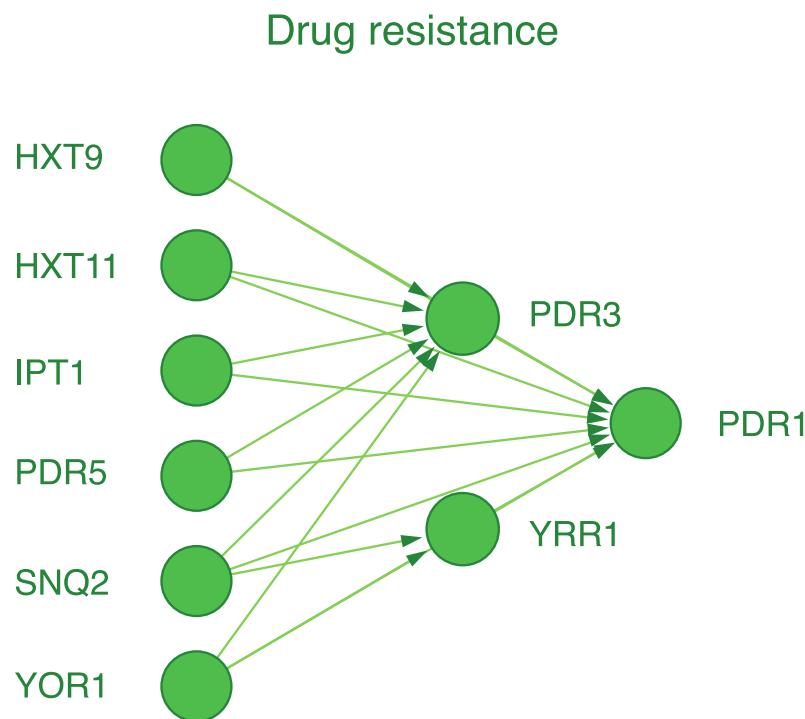
Yeast Regulatory Network



97% detection accuracy (vs. 68-82%)

Structure of Modules

- Feed forward loops:



Many other partitioning methods

■ METIS:

- Heuristic but works really well in practice
- <http://glaros.dtc.umn.edu/gkhome/views/metis>

■ Graclus:

- Based on kernel k-means
- <http://www.cs.utexas.edu/users/dml/Software/graclus.html>

■ Louvain:

- Based on Modularity optimization
- <http://perso.uclouvain.be/vincent.blondel/research/louvain.html>

■ Clique percolation method:

- For finding overlapping clusters
- <http://angel.elte.hu/cfinder/>