Graph Theory and Spectral Methods for Pattern Recognition

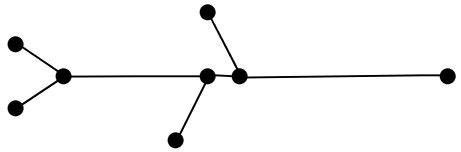
Richard C. Wilson
Dept. of Computer Science
University of York

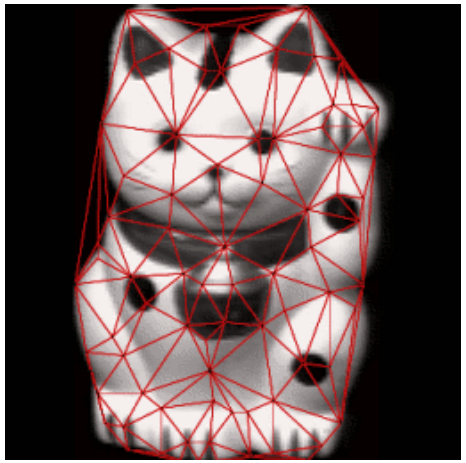
THE UNIVERSITY of York

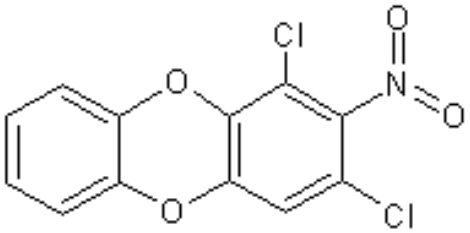
DEPARTMENT OF COMPUTER SCIENCE

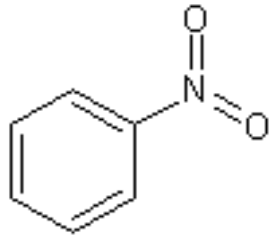
Graphs and Networks

Graphs and *networks* are all around us







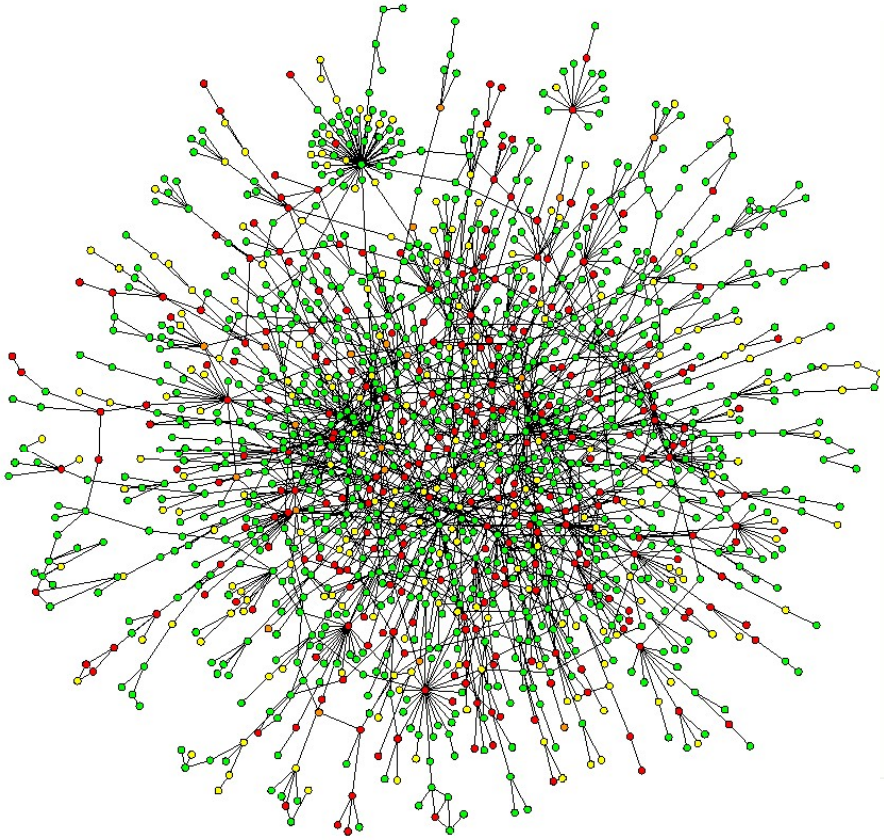


THE UNIVERSITY of York

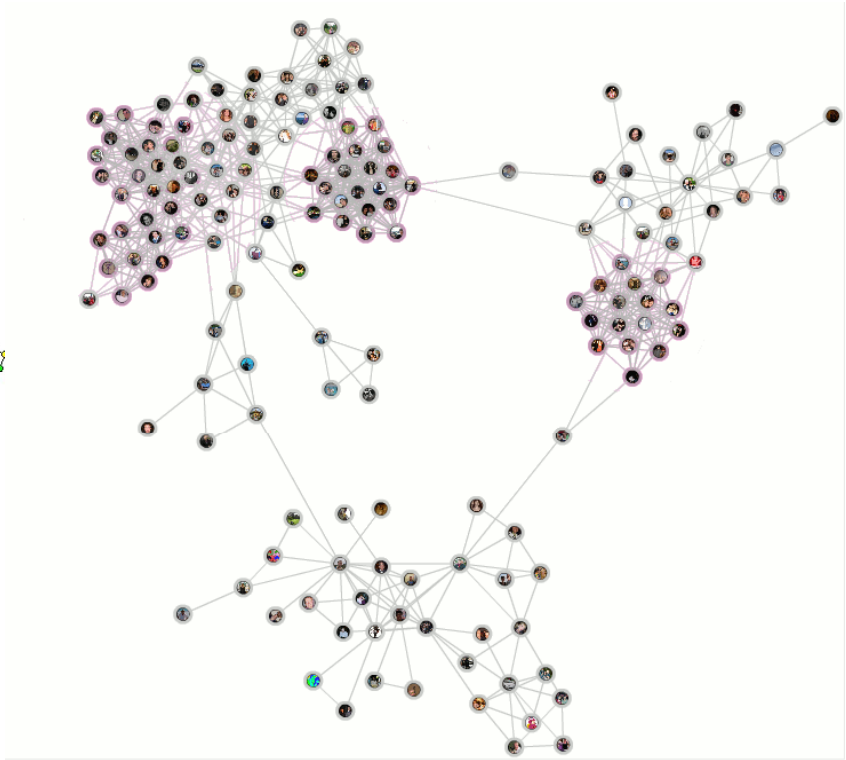
'Simple' networks
10s to 100s of vertices

DEPARTMENT OF COMPUTER SCIENCE

Graphs and networks



PIN



Social Network

‘Complex’ networks
1000s to millions of vertices

THE UNIVERSITY of York

DEPARTMENT OF COMPUTER SCIENCE

What is a network?

- A network consists of
  - a set of vertices (representing parts, elements, objects, features etc)
  - a set of edges (relationships between parts)
- The vertices of a network are often indistinguishable (or at least hard to tell apart)
  - If we can tell one vertex from another reliably, this is a different (easier) problem
- Information encoded in the relationships, not the parts themselves

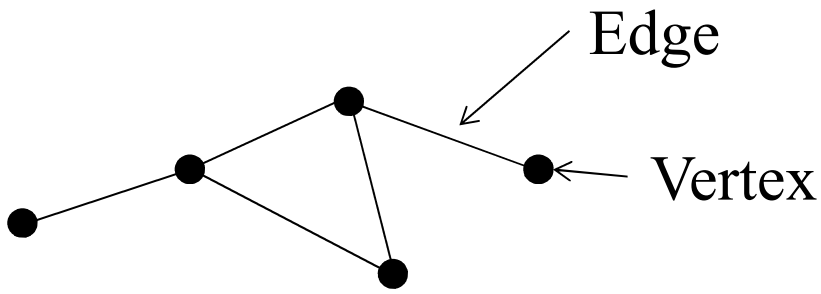
Graph and Networks

There are many interesting questions to ask about network:

What is the structure of a network?
Are there parts? (clustering)
How are they connected?
Do the parts look the same? (similarity, stationary)

Are two networks the same? (isomorphism)
How similar are they? (inexact matching)
Can we tell two types of network apart? (features)
How can we model a set of networks? (models)

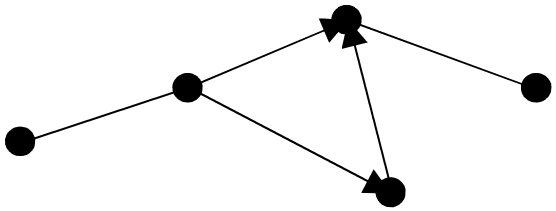
A Network



- Vertices denote objects and edges denote a relationship between a pair of vertices
- Vertices and edges may have discrete *labels* or continuous *measurements* associated with them
  - The graph is then called attributed or an attributed relational graph (ARG)
- A particular type of ARG has weights on the edges $\in [0,1]$ representing the strength of the connection
  - Called a *weighted graph*

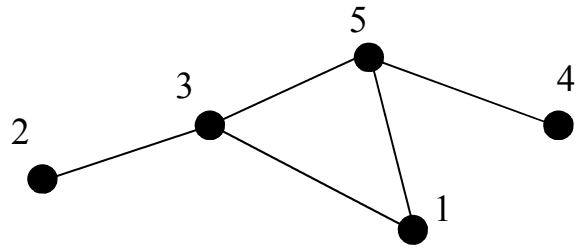
A Network

- Graphs can be undirected or directed.
  - Directed means the edges have a direction to them



- The degree of a vertex is the number of edges connected to that vertex
  - For directed graphs we have in-degree and out-degree

A Network



- Networks are structural – it is the arrangement of edges that matters
- In order to compare the edges, we need to know which is which
- We can do this by labelling the vertices
- In a ‘pure’ network, there is no intrinsic difference between the vertices
- We do not know which labelling is the best and there are $n!$ labellings

Notation

- Common notation

$$G = (V, E, A)$$

V is the set of vertices ($|V|$ is the order of the graph)

E is the set of edges ($|E|$ is the size of the graph)

$$e \in E \begin{cases} e = \{u, v\}, u \in V, v \in V & \text{undirected edge} \\ e = (u, v), u \in V, v \in V & \text{directed edge} \end{cases}$$

A is an attribute functions, maps vertices and edges onto their attributes

Key Graph Theory Problems

- Graph Isomorphism
  - Is there a mapping between the vertices which makes the edges sets of the graphs identical?
  - Unknown computational complexity
- Maximal Clique
  - A clique is a set of vertices which are all mutually connected
  - Finding the Maximal Clique is NP-complete
- Maximum Common Subgraph (MCS)
  - Find two subgraphs which are isomorphic between two graphs
  - Can be reduced to maximal clique
- Graph Edit Distance (GED)
  - An example of inexact similarity between two graphs
  - Under some conditions reducable to MCS
  - More on this later...

Labelling

- Key point: **A graph or network does not change when we label it in a different way**
- So if we want to measure something useful about a graph (graph *feature*), then either

We need to make sure the labelling is the same every time (matching)

or

We need to make features which do not depend on the labelling (invariance)

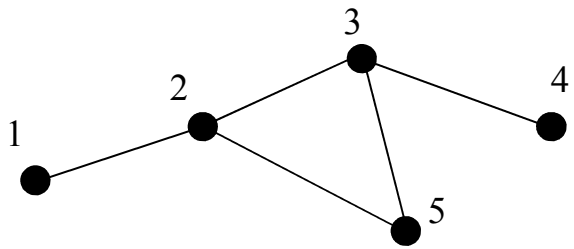
Graph Spectrum

THE UNIVERSITY of York

DEPARTMENT OF COMPUTER SCIENCE

Matrix Representation

- Spectral Graph Theory and related methods depend on the matrix representation of a graph
- A Matrix Representation **X** of a network is matrix with entries representing the vertices and edges
  - First we label the vertices
  - Then an element of the matrix X_{uv} represents the edge between vertices u and v
  - X_{uu} represents the vertex u
  - The most basic example is the adjacency matrix **A**



$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Matrix Representation

- For an undirected graph, the matrix is symmetric
- The adjacency contains no vertex information; The degree matrix **D** contains the degrees of the vertices
  - degree=number of edges containing that vertex

$$\mathbf{D} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

- The Laplacian (**L**) is $\mathbf{L} = \mathbf{D} - \mathbf{A}$

$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 \\ 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & -1 & -1 & 0 & 2 \end{pmatrix}$$

- Signless Laplacian $\mathbf{L}_s = \mathbf{D} + \mathbf{A}$

Matrix Representation

- Normalized Laplacian

$$\hat{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$$

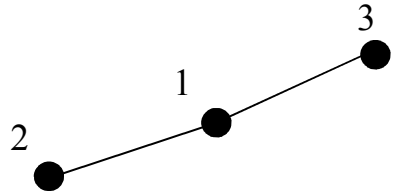
$$= \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$$

- Entries are

$$L_{uv} = \begin{cases} 1 & u = v \\ -\dfrac{1}{\sqrt{d_u d_v}} & (u,v) \in E \\ 0 & \text{otherwise} \end{cases}$$

Incidence matrix

- The incidence matrix of a graph is a matrix describing the relationship between vertices and edges



$$\mathbf{M} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} 1,2 & 1,3 \end{array} \\ \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

- Relationship to signless Laplacian

$$\mathbf{L}_s = \mathbf{M}\mathbf{M}^T$$

- Adjacency

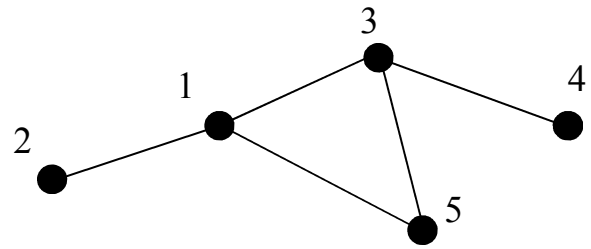
$$\mathbf{A} = \mathbf{M}\mathbf{M}^T - \mathbf{D}$$

- Laplacian

$$\mathbf{L} = 2\mathbf{D} - \mathbf{M}\mathbf{M}^T$$

Matrix Representation

- Consider the Laplacian (**L**) of this network



$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 \\ 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & -1 & -1 & 0 & 2 \end{pmatrix} \end{array}$$

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 3 & -1 & -1 & 0 & -1 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 3 & -1 & -1 \\ 0 & 0 & -1 & 1 & 0 \\ -1 & 0 & -1 & 0 & 2 \end{pmatrix} \end{array}$$

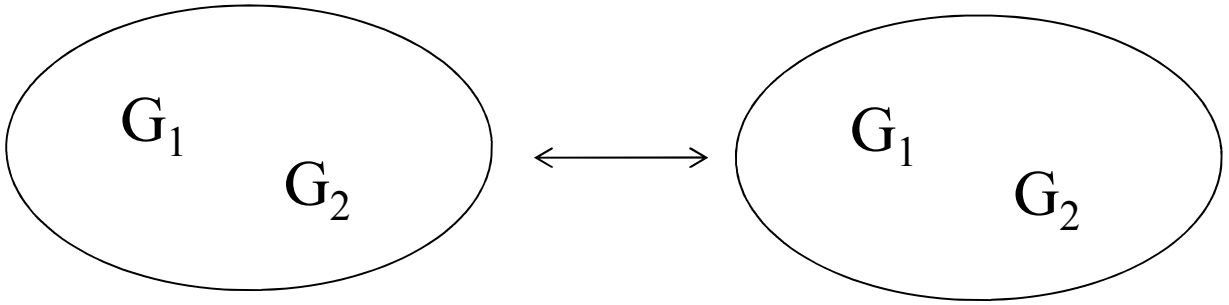
- Clearly if we label the network differently, we get a different matrix
- In fact

$$\mathbf{L}' = \mathbf{PLP}^T$$

represents the same graph for any permutation matrix **P** of the *n* labels

Characterisations

- Are two networks the same? (*Graph Isomorphism*)
  - Is there a bijection between the vertices such that all the edges are in correspondence?
- Interesting problem in computational theory
  - Complexity unknown
  - Hypothesised as separate class in NP-hierarchy, GI-hard
- Graph Automorphism: Isomorphism between a graph and itself
- Equivalence between GI and counting number of GAs



Characterisations

- An equivalent statement: Two networks are isomorphic iff there exists a permutation matrix **P** such that

$$\mathbf{X}_2 = \mathbf{P}\mathbf{X}_1\mathbf{P}^T$$

- **X** should contain all information about the network
  - Applies to **L**, **A** etc not to **D**

X is a *full* matrix representation

- **P** is a relabelling; changes the order in which we label the vertices
- Our measurements from a matrix representation should be invariant under this transformation (*similarity transform*)

Eigendecomposition

- At the heart of spectral graph theory are matrix eigenvalues and eigenvectors

$$\mathbf{X}\mathbf{u} = \lambda\mathbf{u}$$

  - **X** is the square matrix we are interested in
  - λ is an eigenvalue of the matrix
  - **u** is an (right) eigenvector of the matrix
- Left eigenvector

$$\mathbf{u}^T\mathbf{X} = \lambda\mathbf{u}^T$$

- For a symmetric matrix
  - Always *n* orthogonal eigenvectors
  - Eigenvalues real
  - Left & right eigenvectors the same

Spectral Graph Theory

- Any square matrix has an eigendecomposition (into eigenvectors and eigenvalues)
- When dealing with undirected graphs – these have a square and symmetric matrix representation
- The eigendecomposition is then

$$\mathbf{X} = \mathbf{U}\Lambda\mathbf{U}^T$$

$$\begin{pmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \dots & \mathbf{u}_{n-1} \end{pmatrix} \qquad \begin{pmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & & \\ \vdots & & \ddots & \\ 0 & & & \lambda_{n-1} \end{pmatrix}$$

All real numbers

- Later on, I will talk about transition matrices and directed graphs
- These have non-symmetric matrix representations
  - Left and right eigenvalues are the same, but left and right eigenvectors are different

$$\mathbf{X} = \mathbf{U}_R \Lambda \mathbf{U}_L^T$$

$$= \mathbf{U} \Lambda \mathbf{U}^{-1}$$

  - Real or complex-conjugate pairs for eigenvalues

Perron-Frobenius Theorem:

If **X** is an irreducible square matrix with non-negative entries, then there exists an eigenpair (λ,**u**) such that

$$\lambda \in \mathbb{R}^+$$

$$|\lambda| \geq |\lambda_i| \quad \forall i$$

$$u_j \geq 0$$

Applies to both left and right eigenvector

- Key theorem: if our matrix is non-negative, we can find a principal(largest) eigenvalue which is positive and has a non-negative eigenvector
- Irreducible implies associated digraph is strongly connected

Spectral Graph Theory

- The graph has a ordered set of eigenvalues ($\lambda_0, \lambda_1, \ldots \lambda_{n-1}$)
- Ordered in terms of size (I will use smallest first)
- The (ordered) set of eigenvalues is called the *spectrum* of the graph
- I will discuss how the spectrum and the eigenvectors provide useful information about the graph

A note on computation

- Many efficient computational routines available for eigendecomposition
  - Most notably Lapack + machine specific optimisations
  - N^3 complexity
- Suitable for networks with thousands of vertices
- Problematic for networks of 10000+ vertices
- Often such networks are sparse
  - Very low edge density
- In nearly all cases, you only need some of the largest eigenvalues
- For sparse network, small set of eigenvalues, use the Lanczos method

Spectrum

Theorem: The spectrum is unchanged by the relabelling transform

$$\mathbf{X}_2 = \mathbf{P}\mathbf{X}_1\mathbf{P}^T$$

$$\Lambda_2 = \Lambda_1$$

- The spectrum is an acceptable graph feature

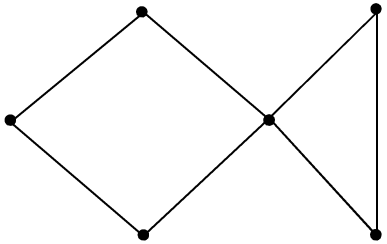
Corollary: If two graphs are isomorphic, they have the same spectrum

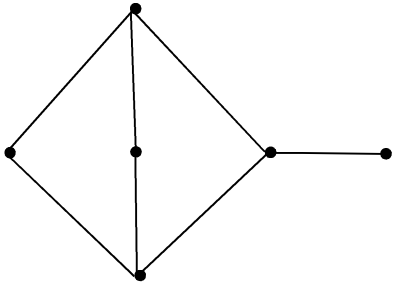
- This does not solve the isomorphism problem, as two different graphs may have the same spectrum

Spectrum

- These two graphs have the same spectrum using the Laplacian representation

[5.24]
$[3]^2$
2
0.76



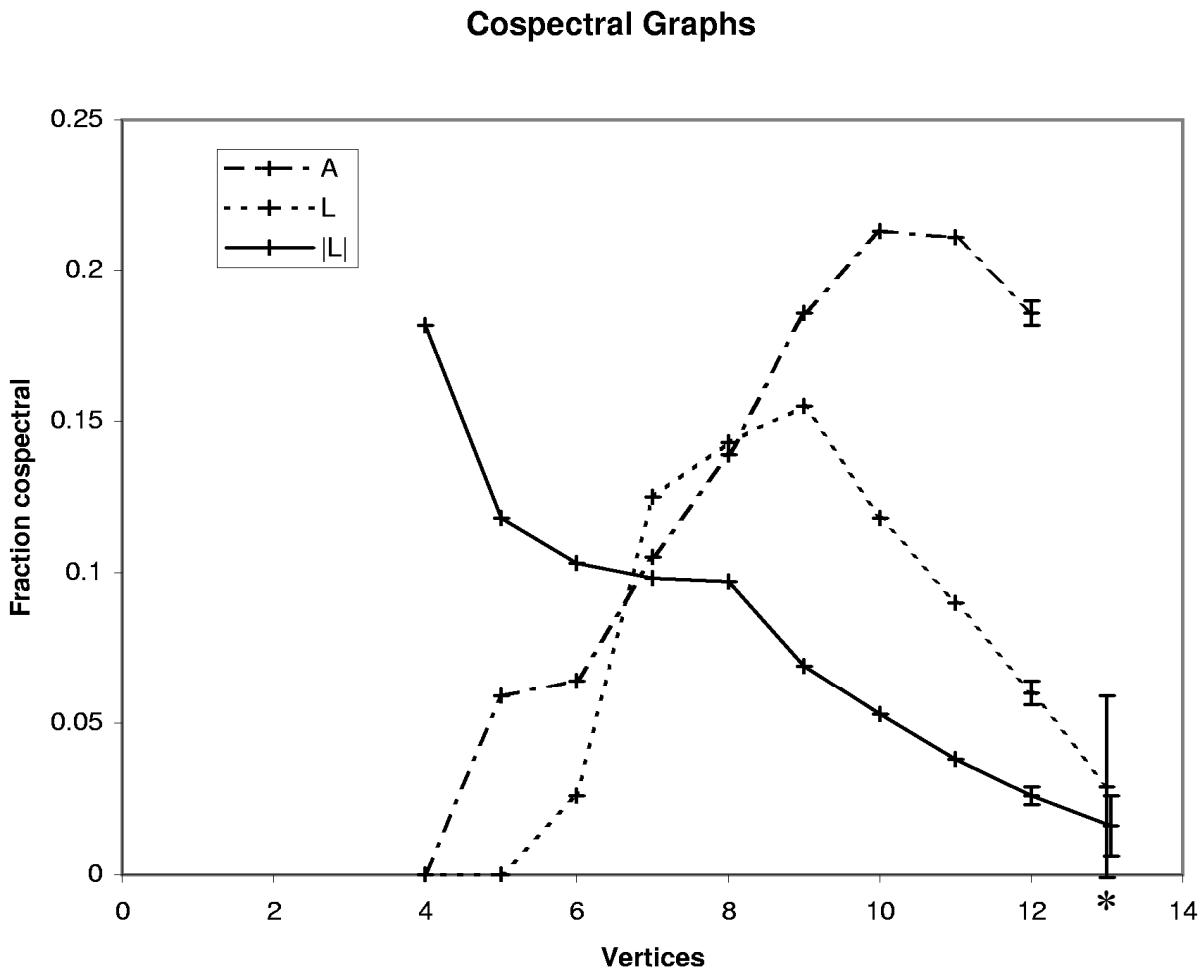


[5.24]
$[3]^2$
2
0.76

- This is a *cospectral* pair
- Necessary but not sufficient...
- The matrix representation we use has a big effect on how many of these cospectral graphs there are

Cospectral graphs

- How many such graphs are there and how does it depend on representation? (Zhu & Wilson 2008)



*50 trillion graphs of size 13

TH

DEPARTMENT OF COMPUTER SCIENCE

Cospectrality

- Open problem: Is there a representation in which nearly all graphs are determined by the spectrum (non-cospectral)?
- Answer for trees: No, nearly all trees are cospectral
- In practice, cospectrality not a problem
  - Two randomly selected graphs have tiny chance of being cospectral
- If we pick graphs from a specialised family, may be a problem
  - Regular, strongly regular graphs

Spectrum of A

Spectrum of **A**:

- Positive and negative eigenvalues

$$\sum \lambda_i = 0$$

$$-d_{\max} \leq \lambda_0 \leq \lambda_1 \ldots \leq 0 \leq \ldots \lambda_{n-1} \leq d_{\max}$$

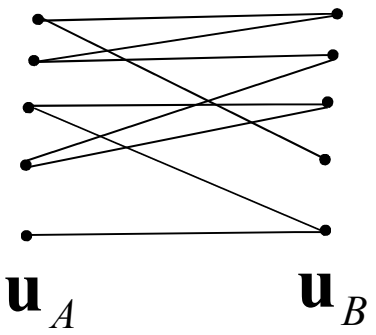
$$\lambda_{n-1} \geq -\lambda_0$$

- Bipartite graph
  - If λ is an eigenvalue, then so is $-\lambda$
  - Sp(**A**) symmetric around 0

Eigenvectors:

- Perron-Frobenius Theorem (**A** non-negative matrix)
  - λ_{n-1} is largest magnitude eigenvalue
  - Corresponding eigenvector $\mathbf{x}_{n-1}$ is non-negative

- Bipartite graph



- Adjacency has form

$$\mathbf{A} = \begin{pmatrix} 0 & \mathbf{A}' \\ \mathbf{A}'^T & 0 \end{pmatrix}$$

- If $(\mathbf{u}_A\ \mathbf{u}_B)^T$ is an eigenvector with eigenvalue λ then $(\mathbf{u}_A\ -\mathbf{u}_B)^T$ is an eigenvector with eigenvalue $-\lambda$
- The adjacency spectrum is symmetric around zero

Spectrum of L

Spectrum of **L**

- **L** positive semi-definite

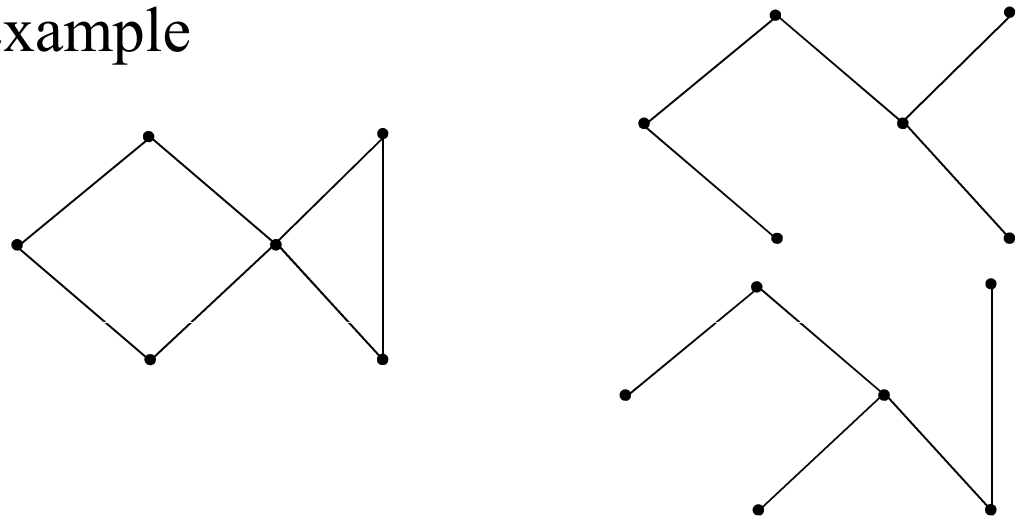
$$\sum \lambda_i = 2|E|$$

$$0 = \lambda_0 \leq \lambda_1 \ldots \leq \lambda_{n-1} < n$$

- There always exists an eigenvector **1** with eigenvalue 0
  - Because of zero row-sums
- The number zeros in the spectrum is the number of disconnected components of the graph.

Spanning trees

- A spanning tree of a graph is a tree containing only edges in the graph and all the vertices
- Example



- Kirchhoff's theorem

The number of spanning trees of a graph is

$$\frac{1}{n}\prod_{i=1}^{n-1}\lambda_i$$

Spectrum of normalised L

Spectrum of $\hat{\mathbf{L}}$

- $\hat{\mathbf{L}}$ positive semi-definite

$$\sum \lambda_i = |V|$$

$$0 = \lambda_0 \leq \lambda_1 \ldots \leq \lambda_{n-1} \leq 2$$

- As with Laplacian, the number zeros in the spectrum is the number of disconnected components of the graph.
- Eigenvector exists with eigenvalue 0 and entries

$$\left(\sqrt{d_1} \quad \sqrt{d_2} \quad \ldots \quad \sqrt{d_n}\right)^T$$

- 'scale invariance'

Information from Spectrum

- We can get useful information direct from the spectrum:
- The Laplacians are positive semidefinite with smallest eigenvalue 0
  - Normalized Laplacian has max eigenvalue 2
- Sp(**L**) for a graph of disconnected components is the union of the spectra of all the components
  - Hence the number of zero eigenvalues counts the number of components
- Spectra of Graphs [Brouwer & Haemers, Springer]

Information from Spectrum

- For regular graphs, the spectrum of **A** and **L** directly related

$$\mathbf{D} = k\mathbf{I}$$

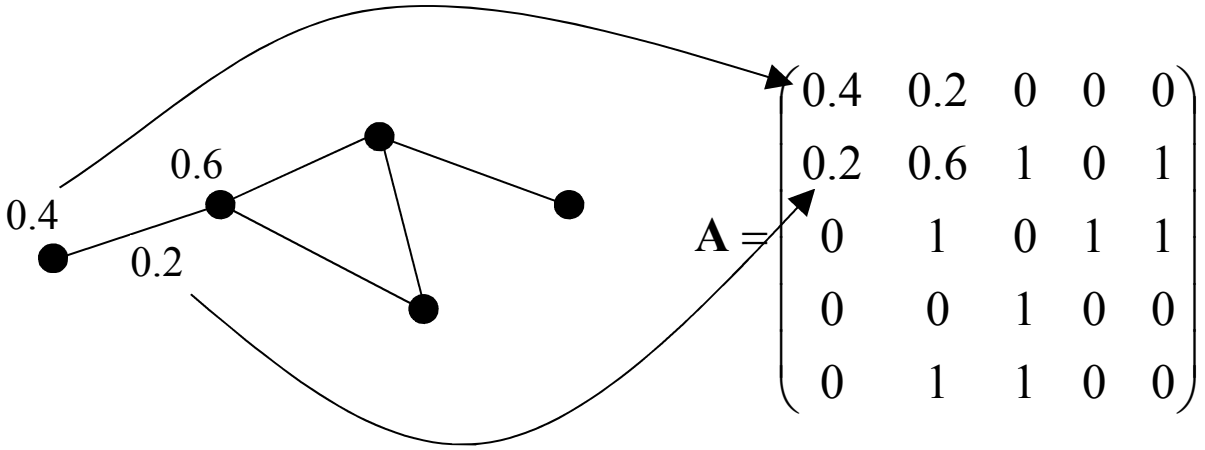
$$\mathbf{L} = k\mathbf{I} - \mathbf{A}$$

$$\lambda_L = k - \lambda_A$$

- Smallest eigenpair of **A** becomes largest of **L**
- For non-regular graphs, eigenpairs are not simply connected
  - But small eigenvalues of **A** correspond in some sense to large eigenvalues of **L**

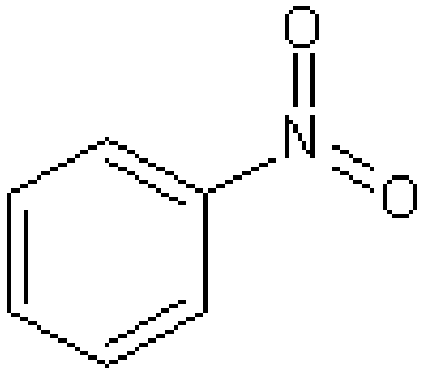
Coding Attributes

- So far, we have considered edges only as present or absent {0,1}
- If we have more edge information, can encode in a variety of ways
- Edges can be weighted to encode attribute
- Include diagonal entries to encode vertices



Coding Attributes

- Note: When using Laplacian, add diagonal elements after forming **L**
- Label attributes: Code labels into [0,1]
- Example: chemical structures



Edges	
—	0.5
=	1.0
Aromatic	0.75

Vertices	
C	0.7
N	0.8
O	0.9

Coding Attributes

- Spectral theory works equally well for complex matrices
- Matrix entry is $x+iy$
- Can encode two independent attributes per entry, x and y
- Symmetric matrix becomes Hermitian matrix
  - Unchanged by conjugate transpose †, transpose+complex conjugate

$$\mathbf{A}^{\dagger} = \mathbf{A}$$

$$\mathbf{A} = \begin{pmatrix} 0 & 0.5-0.3i & 0 \\ 0.5+0.3i & 0 & 0.1+0.2i \\ 0 & 0.1-0.3i & 0 \end{pmatrix}$$

- Eigenvalues real, eigenvectors complex

Coding Attributes

- Example: Shape skeletons

- Shock graph has vertices where shocks meets and edges with lengths l and angles θ
- Encode as complex weight

$$A_{ij} = l_{ij} e^{i\theta_{ij}} = l_{ij}\cos\theta_{ij} + i l_{ij}\sin\theta_{ij}$$

- Naturally hermitian as

$$l_{ij} = l_{ji}$$

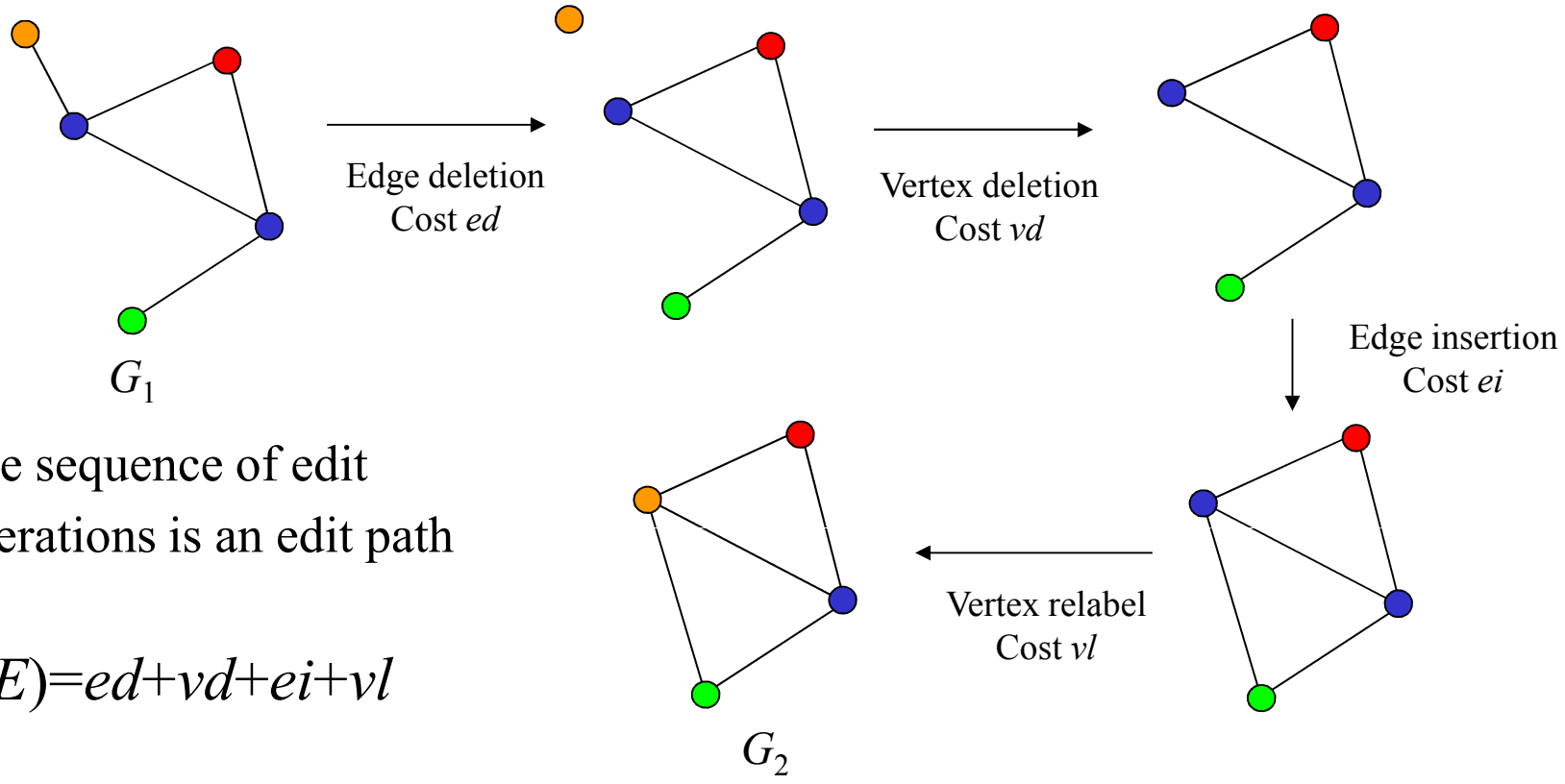
$$\theta_{ij} = -\theta_{ji}$$

Similarity

Similarity of Networks

- How can we measure the similarity of two networks?
- Key idea: Graph Edit Distance(GED)
- Edit operations
  - Vertex insertion, deletion
  - Edge insertion, deletion
  - Relabelling a vertex
- Associate a cost with each operation
- Find a sequence of edit operations which transforms one network into the other
- The minimum possible cost of a sequence is the graph edit distance
- NP-complete so we cannot actually compute it

GED - example



The sequence of edit operations is an edit path E

$c(E)=ed+vd+ei+vl$

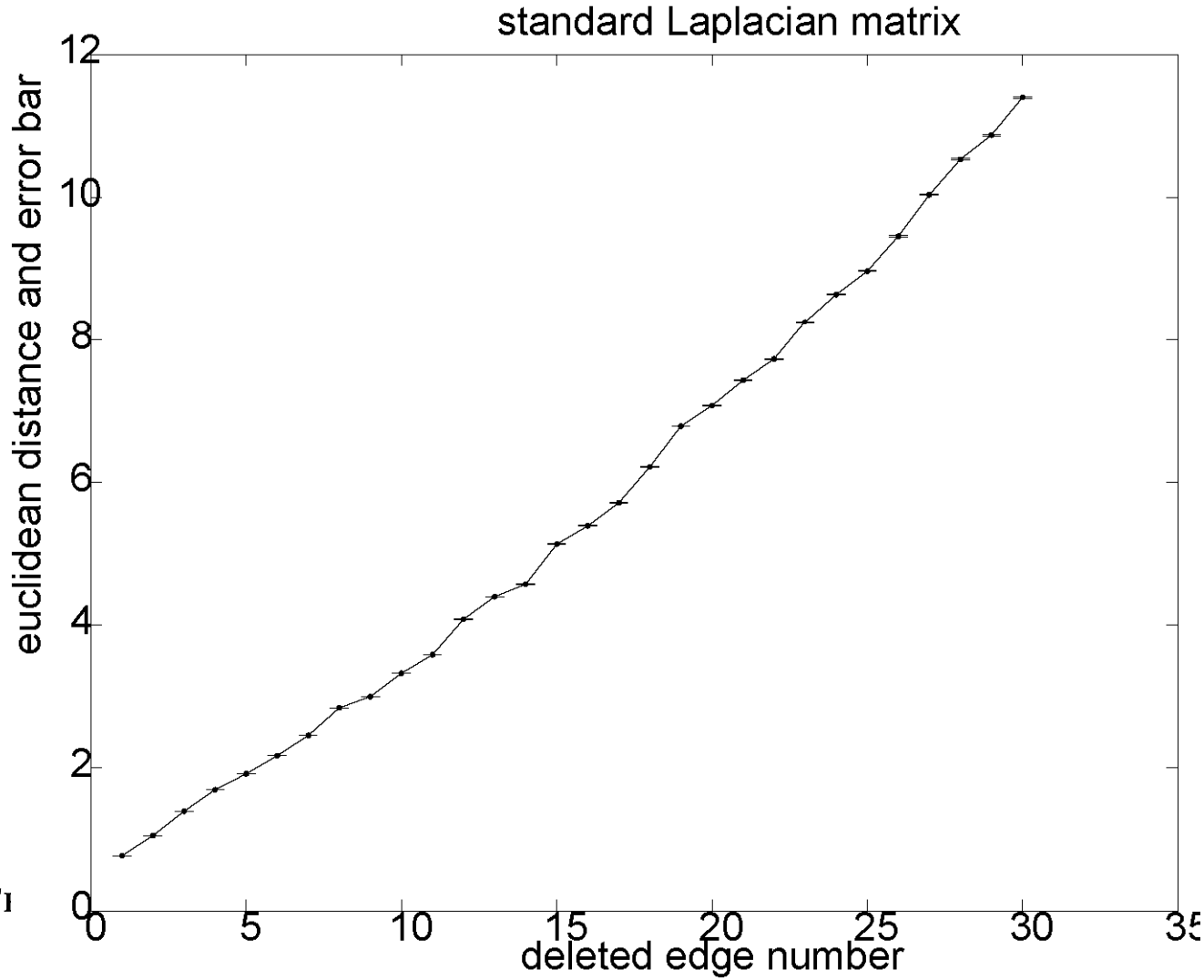
$$GED(G_1, G_2) = \min_{E} c(E)$$

Graph similarity

- The simplest form of GED is zero cost for vertex operations and relabelling
  - Then equivalent to Maximum Common Subgraph [Bunke, PAMI 1999]
- Since we cannot compute GED, we generally resort to approximate methods
  - Compute matches
  - Compare features
- If we can get good features, we can use them to compare graphs

Spectral Similarity

- How good is the spectrum for similarity comparisons? [Zhu, Wilson 2008]



Spectral Features

- The eigendecomposition of a matrix representation is

$$\mathbf{X} = \mathbf{U}\Lambda\mathbf{U}^T$$

- We used the eigenvalues in the spectrum, but there is valuable information in the eigenvectors.
  - Unfortunately the eigenvectors are not invariant $\mathbf{U} \rightarrow \mathbf{PU}$
  - The components are permuted
- Spectral approach partially solves labelling problem
  - Reduced from a similarity transform to permutation

Eigenvectors

Theorem: The eigenvector components are permuted by the relabelling transform

$$\mathbf{X}_2 = \mathbf{P}\mathbf{X}_1\mathbf{P}^T$$

$$\mathbf{U}_2\Lambda\mathbf{U}_2^T = \mathbf{P}\mathbf{U}_1\Lambda\mathbf{U}_1^T\mathbf{P}^T$$

$$\mathbf{U}_2 = \mathbf{P}\mathbf{U}_1$$

The columns of **U** are ordered by the eigenvalues, but the rows still depend on the labelling

Additional problem: If eigenvalues repeat, then **U** is not unique

$$(\lambda, \mathbf{u}_1), (\lambda, \mathbf{u}_2)$$

$$(\lambda, a\mathbf{u}_1 + b\mathbf{u}_2)$$

Spectral Features

- Can we use the eigenvectors to provide features for a network?
- Observation:

$$S_2(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2x_3$$

is a polynomial which does not change when the variables are permuted

- Part of a family of elementary symmetric polynomials invariant to permutation [Wilson & Hancock 2003]

$$S_r(\mathbf{x}) = \sum_{i_1<i_2<\ldots<i_r} x_{i_1,\omega}x_{i_2,\omega}\ldots x_{i_r,\omega}$$

- Hence if $\mathbf{u}$ is an eigenvector, $S_r(\mathbf{u})$ is a network feature

- Shape graphs distributed by polynomial features

Spectral Features

Theorem:

All graphs which have simple spectra can be distinguished from each other in polynomial time

- Simple spectrum means than there are no repeated eigenvalues in the spectrum
- Hence the eigendecomposition is unique
- Then we can order the components of the eigenvectors in polynomial time
  - For example by sorting
- Comparison then determines if they are isomorphic
- Open Problem: Repeated eigenvalues, difficult graphs for isomorphism and labelling ambiguity are all connected in a way not yet understood

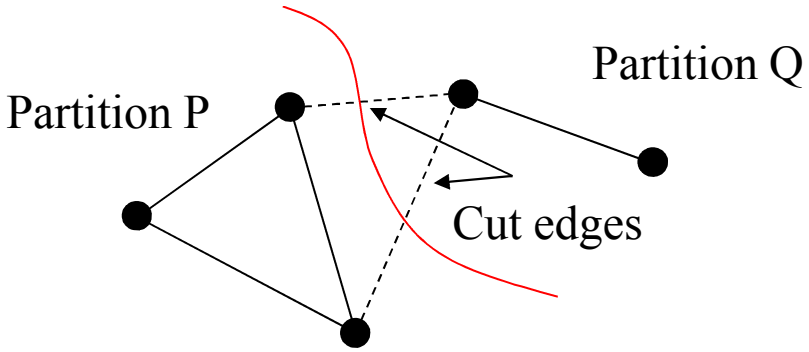
Partitioning

Spectral Partitioning

- The *clustering* problem is a central one for networks
- Also called *community detection*
- Partition the network into parts
  - Highly connected within parts
  - Weakly connected between parts
- Spectral Graph theory can address this problem

Graph Partitioning

A *graph cut* is a partition of a graph into two disjoint sets



The size of the cut is the number of edges cut, or the sum of the weights for weighted graphs

The minimum cut is the cut with smallest size

$$\text{cut}(P,Q) = \sum_{u \in P, v \in Q} A_{uv}$$

Graph Partitioning

- Assume edges indicate similarity
- The goal of clustering is to maintain high intracluster similarity and low intercluster similarity
- Cut measures cost of partition in terms of similarity
  - But must be compared to overall similarity of partitions
- Can measure overall similarity with association

$$\text{assoc}(P,V) = \sum_{u \in P, v \in V} A_{uv}$$

Normalized cut

$$\text{Ncut}(P,Q) = \frac{\text{cut}(P,Q)}{\text{assoc}(P,V)} + \frac{\text{cut}(P,Q)}{\text{assoc}(Q,V)}$$

Shi & Malik 2000

Normalized cut

Define partition vector **x** such that

$$x_u = \begin{cases} +1 \text{ if } u \in P \\ -1 \text{ if } u \in Q \end{cases}$$

Then $$\text{Ncut}(\mathbf{x}) = \frac{\sum_{x_u>0, x_v<0} -A_{uv} x_u x_v}{\sum_{x_u>0, v \in V} A_{uv}} + \frac{\sum_{x_u<0, x_v>0} -A_{uv} x_u x_v}{\sum_{x_u<0, v \in V} A_{uv}}$$

With a bit of transformation we can turn this into a matrix form

$$\text{Ncut}(\mathbf{y}) = \frac{\mathbf{y}^T(\mathbf{D}-\mathbf{A})\mathbf{y}}{\mathbf{y}^T\mathbf{D}\mathbf{y}}$$

$$y_u \in \{1, -b\}$$

$$\mathbf{y}^T\mathbf{D}\mathbf{1} = 0$$

And we should try to minimise Ncut to find the best partition

Normalized cut

- As it is, the problem is hard because **y** is discrete
- Take the relaxation of the problem, i.e. **y** allowed to take real values
- Solution is easily given by solving the eigenvalue problem

$$(\mathbf{D}-\mathbf{A})\mathbf{y}=\lambda\mathbf{D}\mathbf{y}$$

$$\mathbf{D}^{-\frac{1}{2}}(\mathbf{D}-\mathbf{A})\mathbf{D}^{-\frac{1}{2}}\mathbf{z}=\lambda\mathbf{z} \qquad (\mathbf{z}=\mathbf{D}^{\frac{1}{2}}\mathbf{y})$$

$$\left(\mathbf{I}-\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\right)\mathbf{z}=\lambda\mathbf{z}$$

$$\hat{\mathbf{L}}\mathbf{z}=\lambda\mathbf{z}$$

- Hence the solution is an eigenvector of the normalized Laplacian

Normalized Cut

- If we want the smallest Ncut, then we should choose the eigenvector with smallest eigenvalue
- 0 is an eigenvalue of $\hat{\mathbf{L}}$, with corresponding eigenvector

$$\mathbf{u}_0 = \left(\sqrt{d_1} \quad \sqrt{d_2} \quad \ldots \quad \sqrt{d_n}\right)^T$$

  - But $\mathbf{z}=\mathbf{u}_0$ does not satisfy condition $\mathbf{y}^T\mathbf{D1} = 0$

$$\mathbf{y}^T\mathbf{D1} = \mathbf{z}^T\mathbf{D}^{-\frac{1}{2}}\mathbf{D1} = \mathbf{1}^T\mathbf{D}\ \mathbf{1} \neq 0$$

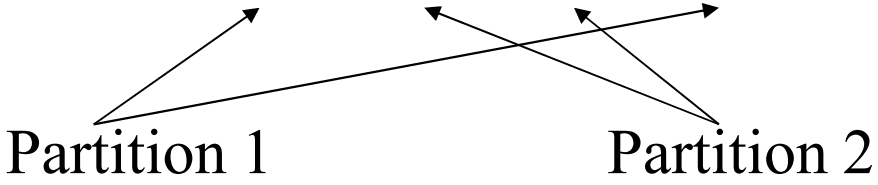
Normalized Cut

- However the eigenvector with second smallest eigenvalue does satisfy this condition
- This is called the *Fiedler vector* and gives an approximate solution to min normalized cut $\mathbf{y}^T\mathbf{D1} = 0$

$$\mathbf{z} = \mathbf{x}_1$$

$$\mathbf{y} = \mathbf{D}^{-1/2}\mathbf{z}$$

- The sign of the components of the Fiedler vector gives the partition
  - Eg $\mathbf{x}_1 = (-0.2 \quad 0.1 \quad 0.3 \quad -0.6)$



Node centrality

- Another issue of interest for complex networks is node centrality
  - How important or significant is a node in a network
- Simple measures of node centrality
- *Degree centrality*
  - Just the degree of the vertex or sum of weights
  - Simple but completely local
- *Betweeness centrality*
  - Measures how many shortest paths between other vertices pass through this vertex
- *Closeness centrality*
  - Finds the 'median' vertex, in the sense of the one which is closest to the rest

Centrality from spectral graph theory

- There is a solution to the centrality problem from spectral graph theory
- Idea: The centrality of u is proportional to the centrality of its neighbours

$$x_u = \alpha \sum_{(u,v)\in E} x_v$$

$$= \frac{1}{\lambda} \sum_{v\in V} A_{uv} x_v$$

- A simple rearrangement of this gives

$$\mathbf{Ax} = \lambda \mathbf{x}$$

- An eigenvector of $\mathbf{A}$ will give a centrality measure

Eigenvector Centrality

- We also require non-negative centrality

$$x_u \geq 0 \forall u$$

- Perron-Frobenius Theorem guarantees for non-negative **A** the principal eigenvector is non-negative
- Eigenvector centrality given by principal eigenvector of **A**

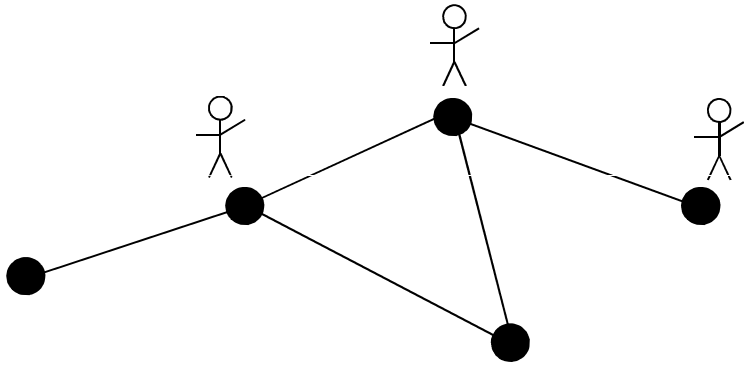
Random Walks

THE UNIVERSITY of York

DEPARTMENT OF COMPUTER SCIENCE

Random Walks

- Spectral features are not tightly coupled to structure
- Can we explore the structure of the network?
- A random walker travels between vertices by choosing an edge at random



- At each time step, a step is taken down an edge

Discrete Time Random Walk

- Imagine that we are standing at vertex u_i
- At each time, we chose one of the available edges with equal probability
- Then the probability of arriving at vertex u_j is

$$P(u_j \mid u_i) = \begin{cases} \frac{1}{d_i} & (u_i, u_j) \in E \\ 0 & (u_i, u_j) \notin E \end{cases} = \frac{A_{ij}}{d_i}$$

- Therefore, at the next time step, the distribution is

$$P_{t+1}(u_j) = \sum_i P(u_j \mid u_i) P_t(u_i)$$

$$= \sum_i \frac{A_{ij}}{d_i} P_t(u_i)$$

Discrete Time Random Walk

- We can write this in matrix form

$$P_{t+1}(u_j) = \sum_i \frac{A_{ij}}{d_i} P_t(u_i)$$

$$\boldsymbol{\pi}_{t+1} = \boldsymbol{\pi}_t \mathbf{D}^{-1}\mathbf{A}$$

$$\boldsymbol{\pi}_{t+1} = \boldsymbol{\pi}_t \mathbf{T} \qquad (\mathbf{T} = \mathbf{D}^{-1}\mathbf{A})$$

- **T** is the transition matrix of the walk
  - Stochastic (rows sum to 1)
  - Largest magnitude eigenvalue 1
- If we start in state $\boldsymbol{\pi}_0$ then at time t

$$\boldsymbol{\pi}_t = \boldsymbol{\pi}_0 \mathbf{T}^t$$

$$\mathbf{T} = \mathbf{U}_R \Lambda \mathbf{U}_L^T$$

- What happens after a very long time?

$$\boldsymbol{\pi}_s = \lim_{t\to\infty} \boldsymbol{\pi}_0 \mathbf{T}^t$$

$$T^t = \mathbf{U}\Lambda^t\mathbf{U}^T$$

$$= \mathbf{U}\begin{pmatrix} \lambda_0^t & \cdots & 0 \\ \vdots & \lambda_1^t & \\ & & \ddots \\ 0 & & \lambda_{n-1}^t \end{pmatrix}\mathbf{U}^T$$

$$|\lambda| < 1 \qquad \lim_{t\to\infty} \lambda^t = 0$$

$$\lambda = 1 \qquad \lim_{t\to\infty} \lambda^t = 1$$

$$\boldsymbol{\pi}_s = \lambda_{n-1}\boldsymbol{\pi}_0 \mathbf{u}_{R,n-1}\mathbf{u}_{L,n-1}^T \qquad \lambda_{n-1} = 1, \mathbf{u}_{R,n-1} = \mathbf{1}$$

$$= 1 \cdot 1 \cdot \mathbf{u}_{L,n-1}^T$$

Discrete Time Random Walks

- After a very long time, the walk becomes stationary
  - Only the largest (left) eigenvector of **T** survives

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{T}$$

  - This is the principal eigenvector of **T** (with λ=1) and is easy to solve; it is

$$\pi_i = \frac{d_i}{|E|}$$

  - After a long time, we are at each node with a probability proportional to its degree
- It is natural to think of the probability as a measure of centrality
- In this situation, eigenvector centrality (of **T**) coincides with degree centrality

PageRank

- One important application for centrality is for the web
  - More central pages are more important
- Idea:
  - Surfer clicks links to new pages at random
  - May also quit and start fresh at a random page ('teleporting')
  - Importance of page is prob of ending up there
- Links are directed, but makes no difference to the formulation

$$\mathbf{T} = (1-\alpha)\mathbf{D}^{-1}\mathbf{A} + \alpha\mathbf{J}$$

  - **J** is matrix of all-ones (teleportation transitions)
  - α is the probability of starting over
- Eigenvector centrality for **T** is the PageRank (Google) of each page

Walk spectra

- **T** is another matrix representation, although it is not symmetric
  - As before can use spectrum as a graph feature
  - Same in character as spectra of other representations
- Symmetric representation can be provided by *support graph*
  - Edge in support graph if there is an n-step path between start and end vertices
  - Equivalent to non-zero entry in $\mathbf{T}^n$
- $\mathbf{S}(\mathbf{T}^n)$ is the support of $\mathbf{T}^n$, set non-zero entries to 1
  - Adjacency matrix of support graph
- Look at the spectrum of $\mathbf{S}(\mathbf{T}^n)$
- For regular graphs, directly related to spectrum of A,T

Differential Equations

THE UNIVERSITY of York

DEPARTMENT OF COMPUTER SCIENCE

Differential Equations on Graphs

- A whole host of important physical processes can be described by differential equations
- Diffusion, or heat flow

$$\frac{\partial p}{\partial t} = \nabla^2 p$$

- Wave propagation

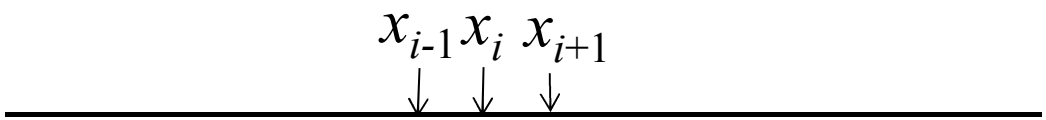
$$\frac{\partial^2 p}{\partial t^2} = -\nabla^2 p$$

- Schrödinger Equation

$$i\hbar \frac{\partial p}{\partial t} = -\frac{\hbar^2}{2m}\nabla^2 p + Vp$$

Laplacian

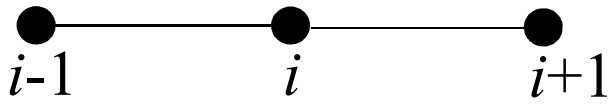
- ∇^2 is the Laplacian differential operator
- In Euclidean space $$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$$
- Different in non-flat spaces
- Take a 1D discrete version of this
  - i, i-1,i+1 denote neighbouring points



$$\nabla^2 p = \frac{\partial^2 p}{\partial x^2} \approx \frac{\partial p}{\partial x}(x_{i+1/2}) - \frac{\partial p}{\partial x}(x_{i-1/2})$$
$$\approx \left[p(x_{i+1}) - p(x_i)\right] - \left[p(x_i) - p(x_{i-1})\right]$$
$$\approx p(x_{i+1}) + p(x_{i-1}) - 2p(x_i)$$

Laplacian

- A graph which encodes the neighbourhood structure



- The Lapacian of this graph is $\mathbf{L} = \begin{pmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{pmatrix}$
- Apply **L** to a vector (a 'function' taking values on the vertices) $(\mathbf{Lp})_i = -p_{i+1} - p_{i-1} + 2p_i$

$$\approx -\nabla^2 p$$

- So the graph Laplacian is a discrete representation of the calculus Laplacian
  - Vertices are points in space
  - Edges represent neighbourhood structure of space
  - Note minus sign!

Diffusion

- On a network, we identify the Laplacian operator ∇^2 with the Laplacian of the network **L**

$$\nabla^2 \longrightarrow \mathbf{-L}$$

- Discrete space, continuous time diffusion process

$$\frac{\partial p}{\partial t} = \nabla^2 p$$

$$\downarrow$$

$$\frac{\partial \mathbf{p}}{\partial t} = -\mathbf{L}\mathbf{p}$$

Heat Kernel

- Solution

$$\boxed{\begin{aligned} p(t) &= \mathbf{H}(t)\,p(0) \\ \mathbf{H}(t) &= \exp(-\mathbf{L}t) \end{aligned}}$$

$$\frac{\partial}{\partial t}\mathbf{H}(t)\mathbf{p}(0) = \left(\frac{\partial}{\partial t}\mathbf{H}(t)\right)\mathbf{p}(0)$$

$$= \left(\frac{\partial}{\partial t}\exp(-\mathbf{L}t)\right)\mathbf{p}(0)$$

$$= -\mathbf{L}\exp(-\mathbf{L}t)\mathbf{p}(0) = -\mathbf{L}\mathbf{H}(t)\mathbf{p}(0)$$

- Heat kernel $\mathbf{H}(t)$
- $H_{ij}(t)$ describes the amount of heat flow from vertex i to j at time t
- Essentially another matrix representation, but can vary time to get different representations

Diffusion as continuous time random walk

- Consider the following walk on a *k*-regular graph
  - At each time step:
    - stay at the same vertex with probability (1-s)
    - Move with prob. s to an adjacent vertex chosen uniformly at random
- This is called a *lazy random walk*
- Transition matrix

$$\mathbf{T} = (1-s)\mathbf{I} + s\frac{1}{k}\mathbf{A}$$

$$= \mathbf{I} - \frac{s}{k}(k\mathbf{I} - \mathbf{A})$$

$$= \mathbf{I} - \frac{s}{k}\mathbf{L}$$

Diffusion as continuous time random walk

- Let s be a time-step
- $n=t/s$ is the number of steps to reach time t

$$\mathbf{T}(t) = \mathbf{T}^n$$

$$= \left[\mathbf{I} - \frac{s}{k}\mathbf{L}\right]^n$$

$$= \left[\mathbf{I} - \frac{t}{nk}\mathbf{L}\right]^n$$

$$\lim_{s\to 0} \mathbf{T}(t) = \lim_{n\to\infty}\left[\mathbf{I} - \frac{t}{nk}\mathbf{L}\right]^n = \exp\left(-\frac{t}{k}\mathbf{L}\right)$$

Spectral representation

$$\mathbf{H}(t) = \exp(-\mathbf{L}t) = \mathbf{U}\exp(-\Lambda t)\mathbf{U}^T$$

$$\exp(-\Lambda t) = \begin{pmatrix} e^{-\lambda_1 t} & 0 & \cdots \\ 0 & e^{-\lambda_2 t} & \\ \vdots & & \ddots \end{pmatrix}$$

- Small times

$$\mathbf{H}(t) = \mathbf{I} - \mathbf{L}t + \frac{1}{2!}\mathbf{L}^2 t^2 + \ldots$$

$$\approx \mathbf{I} - \mathbf{L}t$$

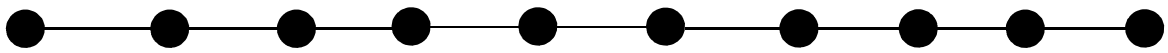
- Large times $e^{-\lambda t} \to 0$
  - Only smallest eigenvalues survive, λ_1=0 and λ_2

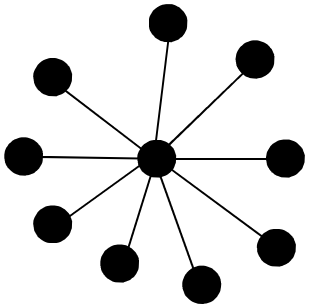
$$\mathbf{H}(t) \approx \frac{1}{n}J + e^{-\lambda_2 t}\phi_2\phi_2^T$$

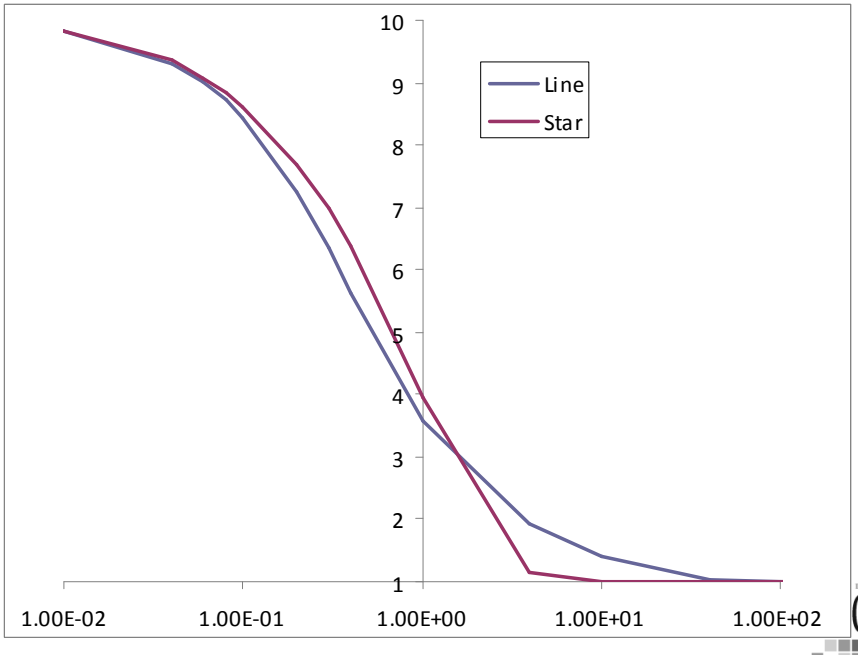
  - Behaves like Fiedler vector (Normalized Cut)

Heat Kernel

- Trace of **H** is a network feature [Xiao, Wilson, Hancock 09]
- Describes a graph based on the shape of heat as it flows across network
  - How much heat is retained at a vertex at time *t*







Heat Kernel Trace

$$\mathrm{Tr}[\mathbf{H}(t)] = \sum_i \exp[-\lambda_i t]$$

- Use moments to describe shape of this curve [Xiao, Wilson, Hancock 09]

$$\boxed{\mu(s) = \int_0^\infty t^{s-1}\,\mathrm{Tr}[\mathbf{H}(t)]dt}$$

Heat Kernel Signature

- Diagonal elements of the heat kernel have been used to characterise 3D object meshes [Sun et al 2009]

$$\text{HKS} = [H_{t_0}(x,x), H_{t_1}(x,x), H_{t_2}(x,x), \ldots]$$

- Describes a particular vertex (for matching) by heat content at various times
- Global version to characterise whole mesh

$$\text{GHKS} = \begin{pmatrix} \text{hist}\left(H_{t_0}(u_1,u_1), H_{t_0}(u_2,u_2), \ldots\right) \\ \text{hist}\left(H_{t_1}(u_1,u_1), H_{t_1}(u_2,u_2), \ldots\right) \\ \vdots \end{pmatrix}$$

Subgraph Centrality

- We can use the heat kernel to define another type of node centrality measure
- Consider the following adjacency matrix as a weighted graph (with weights $1/\sqrt{d_u d_v}$ on the edges)

$$\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$$

- The weighted sum of all paths of length k between two vertices u and v is given by

$$\left(\hat{\mathbf{A}}^{k}\right)_{uv}$$

Subgraph Centrality

- Total communication between vertices is sum over paths of all lengths

$$C_{uv} = \sum_{k=0}^{\infty} \alpha \left(\hat{\mathbf{A}}^k \right)_{uv}$$

- α allows us to control the weight of longer paths vs shorter
- What should α be?
  - Number of possible ways to go increases factorially with *k*
  - Longer paths should be weighted less

$$C_{uv} = \sum_{k=0}^{\infty} \frac{t^k}{k!} \left(\hat{\mathbf{A}}^k \right)_{uv}$$

Subgraph centrality

- Subgraph centrality (Estrada, Rodríguez-Velázquez 2005): centrality is the ability of vertex to communicate with others

$$s_u = \sum_v C_{uv} = \sum_v \sum_{k=0}^{\infty} \frac{t^k}{k!} \left(\hat{\mathbf{A}}^k\right)_{uv}$$

- Relationship to heat kernel

$$\mathbf{H}(t) = e^{-\hat{\mathbf{L}}t} = e^{-\mathbf{I}t+\hat{\mathbf{A}}t}$$

$$= e^{-t} e^{\hat{\mathbf{A}}t}$$

$$= e^{-t} \sum_{k=0}^{\infty} \frac{t^k}{k!} \hat{\mathbf{A}}^k = e^{-t}\mathbf{C}$$

$$\mathbf{s} = \frac{e^t}{|V|}\mathbf{H1}$$

- Actually subgraph centrality uses **A**, but results coincide exactly for regular graphs

Graph Complexity

THE UNIVERSITY of York

DEPARTMENT OF COMPUTER SCIENCE

Complexity

- What is complexity?
- Entropy

$$S = \ln W$$

  - Number of ways of arranging system with same macroscopic properties
- Ensemble – collection of systems with identical macroscopic properties
  - Compute probability of particular state

$$S = -\sum_i p_i \ln p_i$$

Graph Complexity

- The *complexity* of a graph is not a clearly defined term
- An empty graph has no complexity, but what about the complete graph?
- Different definitions serve different purposes
- Coloring complexity
  - Number of ways to color a graph
  - NP-hard to compute
- Randomness complexity
  - Distribution on set of graphs
  - Shannon entropy of distribution
- Statistical complexity
  - Based on edge/vertex graph statistics

Graph Complexity

- Heterogeneity Index [Estrada 2010]
- Complexity is non-uniformity of vertex degrees
  - Irregular graphs are complex

$$S = \alpha \sum_{(u,v)\in E} \left(\frac{1}{\sqrt{d_u}} - \frac{1}{\sqrt{d_v}} \right)^2 = \alpha \mathbf{1}^T \hat{\mathbf{L}} \mathbf{1} \propto |V| - \sum \frac{1}{\sqrt{d_u d_v}}$$

- α is a normalizing constant
- 0 for regular graphs
- 1 (maximal) for star graphs

Von-Neumann entropy

- Mixtures of quantum systems are characterised by a density matrix ρ
- This matrix completely characterizes an ensemble of quantum systems
- The ensemble is a probabilistic mixture of quantum systems in a superposition of states
- There is a natural measure of the entropy of the system, the von Neumann entropy
  - An extension of classical entropy

$$S = -\mathrm{Tr}(\rho \ln \rho)$$

- ρ is an hermitian matrix with trace 1

Von Neumann Entropy

- $\hat{\mathbf{L}}$ is a symmetric (so hermitian) matrix with trace |V|
- So we can use $\frac{1}{|V|}\hat{\mathbf{L}}$ as the density matrix of a quantum system, with the von Neumann entropy as its complexity
- Von Neumann graph complexity

$$S = -\mathrm{Tr}\left(\frac{\hat{\mathbf{L}}}{|V|}\ln\frac{\hat{\mathbf{L}}}{|V|}\right) = -\sum_i \frac{\lambda_i}{|V|}\ln\frac{\lambda_i}{|V|}$$

- Depends on spectrum of normalized Laplacian

Approximate von-Neumann Entropy

- Von Neumann entropy can measure complexity of graphs from spectrum, connection to structure not clear
- Approximation:

$$x \ln x \approx x(1-x) \qquad 0 \le x \le 1$$

$$S \approx -\sum_i \frac{\lambda_i}{|V|}\left(1-\frac{\lambda_i}{|V|}\right)$$

$$= \sum_i \frac{\lambda_i^2}{|V|^2} - \sum_i \frac{\lambda_i}{|V|}$$

$$= \frac{1}{|V|^2}\mathrm{Tr}\left(\hat{\mathbf{L}}^2\right) - \frac{1}{|V|}\mathrm{Tr}\left(\hat{\mathbf{L}}\right)$$

Approximate von Neumann Entropy

- Approximate vNE directly connected to structure

$$\mathrm{Tr}\left(\hat{\mathbf{L}}\right)=|V|$$

$$\mathrm{Tr}\left(\hat{\mathbf{L}}^2\right)=|V|+2\sum_{(i,j)\in E}\frac{1}{d_i d_j}$$

$$S=\frac{1}{|V|}+\frac{2}{|V|^2}\sum_{(i,j)\in E}\frac{1}{d_i d_j}-1$$

- Compared with heterogenity, depends on $1/d_i d_j$ rather than $1/\sqrt{d_i d_j}$

Von Neumann Entropy

- Von Neumann entropy can also be used to control modelling complexity [Han et al 2010]
- Minimum description length criterion

$$MDL = LL(M \mid X) + S(M)$$

- Log-likelihood of model given observed data + cost of describing model
- Model cost is entropy

Another complexity

- Partition function for graphs

$$Z = \sum_{H} e^{-E(H)/kT}$$

- E(H) is the energy of graph H
- Can define energy level of graph as [Gutmann 1978]

$$E(G) = \sum \left| \lambda_i \right|$$

- This derives from statistical mechanics
  - Graphs are particles with ‘heat’ and random motion

Another complexity

- Boltzmann distribution

$$P(G) = \frac{e^{-E(G)/kT}}{Z}$$

- $P(G)$ is the probability of a thermalized particle appearing in state G at temperature T
- Then another entropy is given by

$$S = \ln P(G) = -\frac{1}{kT} E(G) - \ln Z$$

$$= -\frac{1}{kT} \sum |\lambda_i| - \ln Z$$

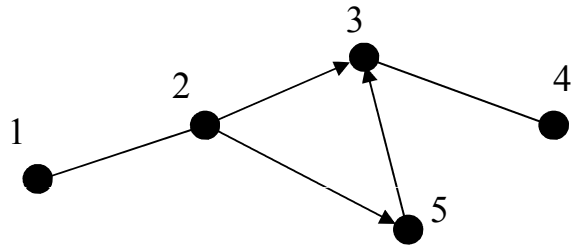
Directed Graphs

THE UNIVERSITY of York

DEPARTMENT OF COMPUTER SCIENCE

Directed graph

- Directed graphs pose some interesting questions for spectral methods
- **A** will be non-symmetric



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

- Spectrum will be complex
  - Real or complex conjugate pairs
- Now have in-degree d_{in} and out-degree d_{out}

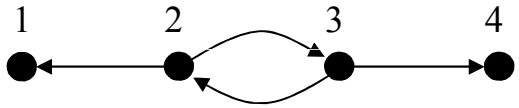
Walks on directed graphs

- The random walk transition matrix can be defined as
  - We select an out-edge at random

$$\mathbf{T} = \mathbf{D}_{\text{out}}^{-1}\mathbf{A}$$

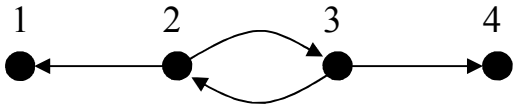
Note **D** is still formed from row-sums (out-degree)

- Walk does not (necessarily) have nice properties



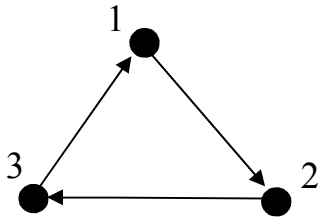
- 1 and 4 are *sink* nodes – once we arrive we can never leave
- Inverse of $\mathbf{D}_{\text{out}}$ not defined when such nodes exist (d_{out}=0)
  - Modify $\mathbf{D}_{\text{out}}^{-1}$ so that the entry is 0 in this case

Walks on directed graphs



- Consider starting a walk at 2, with time t=0
  - At time t=1 there is prob 0.5 of being at 1
  - There is some additional probability of the sequence 2→3 →2 →1
  - Therefore $p_\infty(1) > 0.5$
- Now consider starting at 3
  - By symmetry $p_\infty(4) > 0.5 \quad \Rightarrow p_\infty(1) < 0.5$
- Conclusion: limiting distribution of random walk on directed graph depends on initial conditions
  - Unlike the case of undirected graph

Walks on directed graphs



- Initialise at 1
- Walk follows sequence 1→ 2→3 →1 → 2…
- Walk is *periodic*
  - No limiting distribution

Walks on directed graphs

Strongly connected directed graph:

There exists a path between every pair of vertices

Therefore there are no sinks

- Strongly connected implies that **T** is an irreducible matrix and we can apply the Perron-Frobenius theorem to show (as in the undirected case) that there is a unique non-negative left eigenvector:

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{T}$$

  - Which has eigenvalue 1
- There may be other eigenvectors with absolute eigenvalue 1
  - If there are, then the walk is periodic

Walks on directed graphs

- In spectral theory for directed graphs, we normally confine ourselves to graphs which are
  - Strongly connected (T is irreducible)
  - Aperiodic (T has a single eigenvector with eigenvalue magnitude 1)
- Then the walk converges to a limiting distribution of $\boldsymbol{\pi}$
- The solution for $\boldsymbol{\pi}$ is non-trivial unlike the undirected walk

Laplacian of directed graph

- We can use the walk to define the Laplacian on a directed graph
- The details are a little technical, see [2]
- Let Φ be a diagonal matrix with the elements of $\boldsymbol{\pi}$ on the diagonal
- Laplacian: $\mathbf{L} = \boldsymbol{\Phi} - \frac{1}{2}\left(\boldsymbol{\Phi}\mathbf{T} + \mathbf{T}^T\boldsymbol{\Phi}\right)$
- Normalized Laplacian: $\hat{\mathbf{L}} = \mathbf{I} - \frac{1}{2}\left(\boldsymbol{\Phi}^{\frac{1}{2}}\mathbf{T}\boldsymbol{\Phi}^{-\frac{1}{2}} + \boldsymbol{\Phi}^{-\frac{1}{2}}\mathbf{T}^T\boldsymbol{\Phi}^{\frac{1}{2}}\right)$
- Symmetric
- Coincide with undirected definitions

[2] Laplacians and the Cheeger inequality for directed graphs, *Annals of Combinatorics*, Fan Chung 2005