6-network-analysis

Thursday, February 19, 2026 9:38 AM

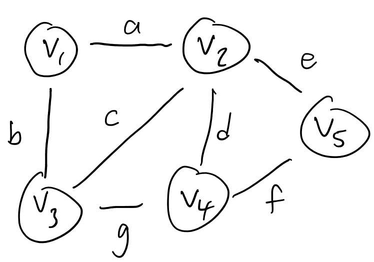
Protein Interactome

Proteins don't act in isolation. Instead, they work together in conjunction with each other. We can experimentally detect pairwise interactions using techniques such as Yeast-2-Hybrid, co-immuno-precipitation, proximity ligation, etc.

Can we go from pairwise interactions to understand complex systems of proteins?

Def. An (undirected) graph is a pair $G=(V,E)$ where $V=\{v_1,\dots,v_m\}$ is a set of vertices/nodes and E is a set of 2-element subsets of V called edges/links. Let $|E|=m'$.

Ex.



Def. The degree $d(v)=\left|\{u\in V \mid \{u,v\}\in E\}\right|$.

The diagonal degree matrix $D=D(G)=\begin{bmatrix} d(v_1) & & & 0 \\ & d(v_2) & & \\ & & \ddots & \\ 0 & & & d(v_m)\end{bmatrix}$.

Def. Given a graph $G=(V,E)$, $\forall$ two nodes $u,v\in V$, a path from $u\to v$ is a sequence of nodes $(v_0,\dots,v_k)$ s.t. $u=v_0$, $v=v_k$ and $\{v_i,v_{i+1}\}\in E$ for $0\le i\le k-1$. k is the length.

A path is closed if $u=v$. G is connected if $\forall u\neq v$, $\exists$ a path $u\to v$.

Connected components are equivalence classes of related nodes, where nodes are related if $\exists$ a path b/t them.

Def. The incidence matrix $B=B(G)\in\mathbb{R}^{m\times m'}$ is

$$b_{ij}=\begin{cases}1 & \text{if } e_j=\{v_i,v_k\} \text{ for some } k\\ 0 & \text{otherwise.}\end{cases}$$

Ex

$$\begin{array}{c} \\ 1\\2\\3\\4\\5\end{array}\begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f\end{array}\\ \begin{bmatrix}1&1&0&0&0&0\\1&0&1&1&1&0\\0&1&1&0&0&0\\0&0&0&1&0&1\\0&0&0&0&1&1\end{bmatrix}\end{array}$$

Def. The oriented incidence matrix B^σ is given by assigning an arbitrary orientation to every edge

each col has one positive & one negative

Def. The oriented incidence matrix B is given by assigning an arbitrary orientation to every edge

$$e = \{u,v\} = \{v,u\} \longrightarrow e = \underbrace{(u,v)}_{} \text{ or } e = \underbrace{(v,u)}_{}$$

we pretend the edges are directed $\Rightarrow$ ordered pair

Then
$$b_{ij} = \begin{cases} -1 & \text{if } e_j = (v_k, v_i) \text{ for some } k \\ 1 & \text{if } e_j = (v_i, v_k) \text{ for some } k \\ 0 & \text{otherwise.} \end{cases}$$

each col has one positive & one negative

Ex
$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix} \end{array}$$

Note: If we were working with directed graphs, this orientation comes for free.

Def. If exists edge $\{u,v\}$, then v is adjacent to u, denoted $u \sim v$.

Def. The adjacency matrix $A = A(G)$ is the symmetric matrix with

$$a_{ij} = \begin{cases} 1 & \text{if } u \sim v \\ 0 & \text{otherwise} \end{cases}$$

Ex.
$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

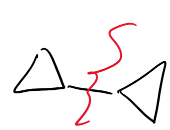
Aside: This is a diffusion operator as the value of Ax at v_i is the sum of the values of x at nodes v_j adjacent to v_i.

This gives a geometric interpretation of the eigenvalues of A, because

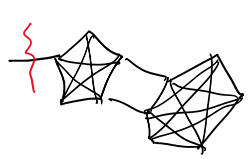
if $Ax = \lambda x$, then $\lambda x_i = \sum_{j \sim i} x_j$, so the sum of the values of x at nodes adjacent to i must be equal to λ times value at v_i.

Back to clustering

Idea 1: Minimize cut.



Problem:



Might just cut out dangling edge.

Idea 2: Cut edges with high betweenness

└ between every pair of nodes, find shortest path, ... based on number of

Idea 2: Cut edges with high betweenness

└ between every pair of nodes, find shortest path, + each edge gets a score based on number of shortest paths going through it

[Girvan-Newman early 2000s]

We can keep iterating until we have disconnected the graph into however many chunks we want.

Problem: how many components is the right number?

Idea 3: "Modularity" quality function.

Clusters should

(1) have few edges crossing clusters

(2) have many edges within clusters

} problem: best solution is not to cut!

compared to a randomized version of "same" network / partitioning

Let A be adj matrix

d_i the degree of node i

c_i the community of node i, partition P

$\delta(c_i, c_j) = 1$ iff $c_i = c_j$ (0 otherwise)

Proportion of edges inside community: $\frac{1}{2m} \sum_{i,j \in V} A_{ij} \, \delta(c_i, c_j)$

In "random" configuration model, where degrees are kept the same but connections randomized:

$$\frac{1}{2m} \sum_{i,j \in V} \frac{d_i d_j}{2m} \, \delta(c_i, c_j)$$

Def. modularity

$$Q(P) = \frac{1}{2m} \sum_{i,j \in V} \left[A_{ij} - \frac{d_i d_j}{2m} \right] \delta(c_i, c_j)$$

$$= \frac{1}{2m} \sum_{C \in P} \left[e_C - \frac{\left(\sum_{i \in C} d_i\right)^2}{2m} \right]$$

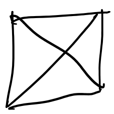
} also due to Girvan-Newman

With a quality function, we can now separate out partitioning algorithms from measuring how good the partitions are.

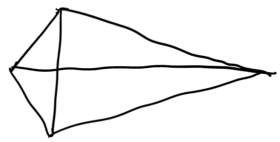
Directly optimizing for Q is NP-hard, but we can now use other heuristics to approximately optimize it, like Louvain or Leiden.

Directly optimizing for Q is NP-hard, but we can now use other heuristics to approximately optimize it, like Louvain or Leiden.

Graph Drawing (also sometimes called graph embedding or graph immersion)

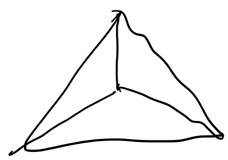


or



are the same graph.

or



How do we compute a "good" drawing?

Aside: maybe a good drawing will tell us something about clusters too.

Def. The matrix $L = B^\sigma (B^\sigma)^T$ is called the (unnormalized) graph Laplacian.

Exercise: Show that for any orientation σ, $L = D - A$.

Def. We can define a weighted graph by a weighted adjacency matrix W where $w_{ij} = w_{ji} > 0$ is the weight on edge $\{v_i, v_j\}$, and $w_{ij} = 0$ iff that edge does not exist.

Def. The degree $d(v_i) = \sum_{j \neq i} w_{ij}$ in a weighted graph.

We can thus define a diagonal matrix D.

Def. The Laplacian matrix is $L = D - W$.

Exercise: If we define the weighted oriented incidence matrix B^σ by

$$b_{ij} = \begin{cases} \sqrt{w_{ij}} & \text{if } v_i \text{ is the source of } e_j \\ -\sqrt{w_{ij}} & \text{if } v_i \text{ is the target of } e_j \\ 0 & \text{else} \end{cases}$$

where σ is an arbitrary orientation of the edges,

then $L = B^\sigma (B^\sigma)^T$.

Theorem L is positive semidefinite. (i.e. it has all nonnegative eigenvalues)

... equivalent statement is that the symmetric quadratic form

Theorem L is positive semidefinite.

proof. Recall that an equivalent statement is that the symmetric quadratic form given by L, $x^T L x$, is always nonnegative.

(Else, let x be an eigenvector with eigenvalue $\lambda < 0$. Then $x^T L x = \lambda x^T x = \lambda \|x\|^2 < 0$.)

For any $x \in \mathbb{R}^m$,

$$x^T L x = x^T D x - x^T W x$$

$$= \sum_{i=1}^{m} d_i x_i^2 - \sum_{i,j} w_{ij} x_i x_j$$

$\left(d_i = \sum_j w_{ij}\right)$

$$= \frac{1}{2}\left(\sum_{i=1}^{m} d_i x_i^2 - 2\sum_{i,j} w_{ij} x_i x_j + \sum_{j=1}^{m} d_j x_j^2 \right)$$

$$= \frac{1}{2}\left(\sum_{i=1}^{m}\sum_{j=1}^{m} \left[w_{ij} x_i^2 - 2 w_{ij} x_i x_j + w_{ij} x_j^2 \right] \right)$$

$$= \frac{1}{2} \sum_{i,j} w_{ij} (x_i - x_j)^2 \geq 0.$$

Exercise: Prove that the all 1's vector $\vec{1}$ is an eigenvector of L with eigenvalue 0.

Fact: For any weighted symmetric graph $G = (V, W)$, the eigenvalues (with multiplicity) $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_m$ of L are real and nonnegative, and there is an orthonormal basis of eigenvectors.

Exercise: Prove that if G is disconnected, $\lambda_2 = 0$.

Hint: Construct two independent eigenvectors for the eigenvalue 0.

Fact: $\dim(\operatorname{Ker} L) = \#$ connected components of G.

Exercise: Prove this fact using $L = B^\sigma (B^\sigma)^T$.

Corollary: If G is connected, $\lambda_2 > 0$.

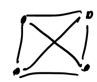
λ_2 known as the **Fiedler number** of the graph.

Def. A **graph drawing** is a function $\rho : V \to \mathbb{R}^n$. The matrix of a graph drawing ρ is an $m \times n$ matrix R where the row $\rho(v_i)$ corresponds to the point representing v_i in $\mathbb{R}^n$.

of a graph drawing ρ is an $m \times n$ matrix R whose ith row $\rho(v_i)$ corresponds to the point representing v_i in $\mathbb{R}^n$.

Ex.

$$R = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$



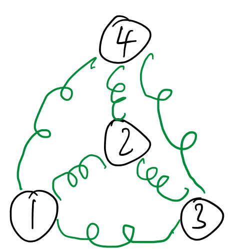
We let $\rho(v_i) = e_i R$ where $e_i = (0 \cdots 0 \, 1 \, 0 \cdots 0)$
↑ pos. i.

Def. A graph drawing is *balanced* iff the sum of the entries of every col of the matrix R is 0. i.e. $\mathbb{1}^T R = 0$.
(i.e. the graph drawing is centered at the origin)

Def. The *energy* of a drawing R is

$$\mathcal{E}(R) = \sum_{\{v_i, v_j\} \in E} \| \rho(v_i) - \rho(v_j) \|^2$$

Analogy:



Connect nodes by springs and minimize potential energy of the system.

"good" drawings are ones that minimize energy, but are not trivial.

Define: The energy of a drawing R of a weighted graph $G = (V, W)$ is

$$\mathcal{E}(R) = \sum_{\{v_i, v_j\} \in E} w_{ij} \| \rho(v_i) - \rho(v_j) \|^2.$$

(think of w_{ij} as spring stiffness)

Thm: $\mathcal{E}(R) = tr(R^T L R)$

proof.

$$\mathcal{E}(R) = \sum_{\{v_i, v_j\} \in E} w_{ij} \| \rho(v_i) - \rho(v_j) \|^2$$

$$= \sum_{k=1}^{n} \sum_{\{v_i, v_j\} \in E} w_{ij} (R_{ik} - R_{jk})^2$$

$$= \sum^{n} \ldots \sum^{m} \ldots (R \ldots - R \ldots)^2$$

$$k=1 \quad \{v_i,v_j\}\in E$$

$$= \sum_{k=1}^{n} \frac{1}{2} \sum_{i,j=1}^{m} w_{ij} \left(R_{ik} - R_{jk}\right)^2$$

$$= \sum_{k=1}^{n} (R^k)^T L R^k \qquad \text{(where } R^k \text{ is the } k\text{th col of } R\text{)}$$

$$= tr(R^T L R)$$

So the energy is the sum of the (nonnegative) eigenvalues of $R^T L R$.

Notice that for any invertible matrix M, $p(v_i)M$ is another graph drawing that conveys the same information as R, so we may as well choose R to have pairwise orthonormal unit length cols, $\underline{R^T R = I}$.

this gives us an orthonormal graph drawing, & rules out trivial drawings.

Notice: We can choose an orthonormal eigenbasis of L

$$u_1, u_2, \dots, u_m$$

$$0 = \lambda_1 < \lambda_2 \le \cdots \le \lambda_m$$

Let $R = \underbrace{\begin{bmatrix} u_2 & u_3 & \cdots & u_{n+1} \end{bmatrix}}_{\text{each } u_i \text{ is a column}}$. Then $R^T L R = \begin{bmatrix} \lambda_2 & & 0 \\ & \ddots & \\ 0 & & \lambda_{n+1} \end{bmatrix}$, so $Tr(R^T L R) = \sum_{k=2}^{n+1} \lambda_k$.

Fact: This choice minimizes the energy of a balanced orthonormal drawing.

Exercise: Explain why we don't want to use the 1st eigenvector associated with $\lambda_1 = 0$.

Notice: We can draw onto the 1D number line, and then divide the graph in half by positive & negative nodes. This corresponds to the Fiedler eigenvector, and is the idea behind spectral clustering.