Detecting and Understanding the Large-Scale Structure of Networks

Mark Newman
University of Michigan

Michelle Girvan (Maryland)
Elizabeth Leicht (Michigan)
Modules, groups, or communities
Modular structure

• Modules are of interest in many cases:
  - World Wide Web
  - Citation networks
  - Social networks
  - Metabolic networks

• Properties of modules may be quite different from average properties of a network
Graph partitioning

- Find the division into groups of given sizes that minimizes the *cut size*, i.e., the number of edges running between groups.
Detecting modules

- Maximizing the number of edges within groups (or minimizing the number between groups) is not enough
- A good division into modules not just one with a large number of edges within groups, but one with a larger than expected number
- This leads us to the idea of modularity
Modularity


Define modularity to be

\[ Q = (\text{number of edges within groups}) - (\text{expected number within groups}). \]

• Modularity is measured relative to a null model

  - Defined by \( P_{ij} = \) probability of an edge between vertices \( i \) and \( j \)

  - Examples:

    \[ P_{ij} = p \text{ (Erdös-Rényi random graph)} \]
    \[ P_{ij} = k_i k_j / 2m \text{ (“configuration model”)} \]
Matrix formulation

Actual number of edges between $i$ and $j$ is

$$A_{ij} = \begin{cases} 
1 & \text{if there is an edge } (i, j), \\
0 & \text{otherwise.}
\end{cases}$$

Expected number of edges is $P_{ij}$.

Modularity is sum of $A_{ij} - P_{ij}$ over all pairs of vertices $(i,j)$ falling in the same group.

Define:

$$s_i = \begin{cases} 
+1 & \text{if vertex } i \text{ belongs to group 1,} \\
-1 & \text{if vertex } i \text{ belongs to group 2.}
\end{cases}$$
\[
Q = \frac{1}{2m} \sum_{ij} [A_{ij} - P_{ij}] \delta(g_i, g_j)
\]

\[
= \frac{1}{4m} \sum_{ij} [A_{ij} - P_{ij}] (s_i s_j + 1)
\]

\[
= \frac{1}{4m} \sum_{ij} [A_{ij} - P_{ij}] s_i s_j
\]

\[
= \frac{1}{4m} s^T B s
\]

where \( B_{ij} = A_{ij} - P_{ij} \)

**We call \( B \) the modularity matrix**
• Now we write \( s \) as a linear combination of the eigenvectors \( u_i \) of the modularity matrix:

\[
s = \sum_{i=1}^{n} a_i u_i, \quad \text{with} \quad a_i = u_i^T s
\]

\[
Q = \frac{1}{4m} s^T Bs = \frac{1}{4m} \sum_i a_i^2 \beta_i
\]

• Maximize by choosing \( s \) parallel to the leading eigenvector, or failing that, as near parallel as we can

\[
s_i = \begin{cases} 
+1 & \text{if } u_i^{(1)} \geq 0, \\
-1 & \text{if } u_i^{(1)} < 0.
\end{cases}
\]
Example: animal network
Books about politics
Spectral properties of modularity matrix

- Vector (1, 1, 1, ...) is always an eigenvector of $B$ with eigenvalue zero, corresponding to all vertices in the same group.
- Eigenvalues can be either positive or negative.
  - So long as there is any positive eigenvalue we will never put all vertices in the same group.
- But there may be no positive eigenvalues.
  - All vertices in same group gives highest modularity.
  - We call such networks indivisible.
Dividing into more than two groups

- Simplest approach is repeated division into two groups
  - Divide in two, then divide those parts in two, etc.
- Stop when there is no division that will increase the modularity
  - But this is precisely when the subgraph is indivisible
  - Stop when there are no positive eigenvalues of the modularity matrix
Negative eigenvalues

- Unlike the Laplacian, the modularity matrix has negative eigenvalues
- These tell us about minimization of the modularity
- A division with negative modularity has fewer edges than expected within communities (or more than expected between communities)
• This corresponds to a network with bipartite structure
• Or \( k \)-partite in the general case
Network of word adjacencies

- Green circles: Adjective
- Red squares: Noun
Network of word adjacencies

- Adjective
- Noun
Vertex classification
(Newman and Leicht 2007)

- We specify a very broad set of possible structures that we are interested in:
Definition of the model

• There are three kinds of quantities in this approach:
  − Observed data: the pattern of edges observed between the vertices. These are given to us by the experimenter.
  − Missing data: We assume that the vertices divide into $c$ groups. We denote the group to which vertex $i$ belongs by $g_i$. These are missing data.
  − Model parameters: these describe the patterns of connection between vertices in different groups.
Definition of the model

Directed case:

\[ \pi_r = \text{probability of being in group } r \]

and

\[ \theta_{ri} = \text{probability of a link to vertex } i \]

These satisfy

\[ \sum_{r=1}^{c} \pi_r = 1, \quad \sum_{i=1}^{n} \theta_{ri} = 1. \]
Likelihood and log-likelihood

• The likelihood is

\[ \Pr(A, g | \pi, \theta) = \Pr(A | g, \pi, \theta) \Pr(g | \pi, \theta) \]

• Here

\[ \Pr(A | g, \pi, \theta) = \prod_{ij} \theta_{g_i,j}^{A_{ij}}, \quad \Pr(g | \pi, \theta) = \prod_{i} \pi_{g_i} \]

• So

\[ \Pr(A, g | \pi, \theta) = \prod_{i} \left[ \pi_{g_i} \prod_{j} \theta_{g_i,j}^{A_{ij}} \right] \]

\[ \mathcal{L} = \ln \Pr(A, g | \pi, \theta) = \sum_{i} \left[ \ln \pi_{g_i} + \sum_{j} A_{ij} \ln \theta_{g_i,j} \right] \]
• Unfortunately, we don't know the values of the missing data, so we can't evaluate this expression.

• However, we can make a pretty good guess at the values of the missing data if we know $A$, $\pi$, and $\theta$. More specifically, we can calculate the probability that $g_i$ takes a particular value $r$ thus:

$$q_{ir} = \Pr(g_i = r | A, \pi, \theta) = \frac{\Pr(A, g_i = r | \pi, \theta)}{\Pr(A | \pi, \theta)}.$$ 

• The numerator we can calculate by summing $\Pr(A, g | \pi, \theta)$ over all the $g$s except $g_i$.

• The denominator is fixed by the normalization.
• The result is:

\[ q_{ir} = \frac{\pi_r \prod_j \theta_{rj}^{A_{ij}}}{\sum_s \pi_s \prod_j \theta_{sj}^{A_{ij}}} \]

• This looks odd: we're saying you can calculate \( q_{ir} \) given the model and the data, and then we're going to calculate the model from \( q_{ir} \) and the data?

• Yes, but we have to do it self-consistently. . .
Expected likelihood

- We can now make a guess about the value of the log-likelihood. Our best guess is just the expectation value:

\[
\mathcal{L} = \sum_{g_1=1}^{c} \cdots \sum_{g_n=1}^{c} \Pr(g|A, \pi, \theta) \sum_{i} \left[ \ln \pi_{g_i} + \sum_{j} A_{ij} \ln \theta_{g_i,j} \right]
\]

\[
= \sum_{ir} \Pr(g_i = r|A, \pi, \theta) \left[ \ln \pi_r + \sum_{j} A_{ij} \ln \theta_{rj} \right]
\]

\[
= \sum_{ir} q_{ir} \left[ \ln \pi_r + \sum_{j} A_{ij} \ln \theta_{rj} \right].
\]
Now it's a straightforward matter to maximize this with respect to $\pi$ and $\theta$ to find the best values. The result is:

\[
\pi_r = \frac{1}{n} \sum_i q_{ir}, \quad \theta_{rj} = \frac{\sum_i A_{ij} q_{ir}}{\sum_i k_i q_{ir}},
\]

So we have $\pi$ and $\theta$ in terms of $q$ and we have $q$ in terms of $\pi$ and $\theta$

To find a self-consistent solution to both sets of equations, we iterate from a suitable set of starting values.
Expectation-Maximization Algorithm

• Has a number of clear advantages:
  
  - Very simple: just a few lines of computer code to implement the method
  
  - Fast: typically only a few seconds to analyze even a large network
  
  - Simultaneously tells us how to group the vertices in the network and what the appropriate definition is for the groups

• Derivation is more complicated for undirected case, but the final equations are exactly the same
Example: Social network
Example: Lexical network
Ordinary community detection

EM algorithm
• References:
  - See: \url{http://www.umich.edu/~mejn/pubs.html}