Modularity and Code Length in Community Detection

Social Networks (SOC6110, Spring 2021)

Barum Park (b.park@cornell.edu)

Modularity

Modularity: Two Definitions

You'll find two equivalent definitions of the modularity in the literature:

1. One that looks like this (Newman and Grivan 2004):

$$Q = \sum_{k=1}^{K} (e_{kk} - e_k^2)$$

   where

   ▷ K is the number of *modules*,
   ▷ e_{kk} is the *proportion* of edges connecting nodes within module k,
   ▷ e_k is the *marginal* proportion of edges that have at least one end node in module k.

Modularity: Two Definitions

You'll find two equivalent definitions of the modularity in the literature:

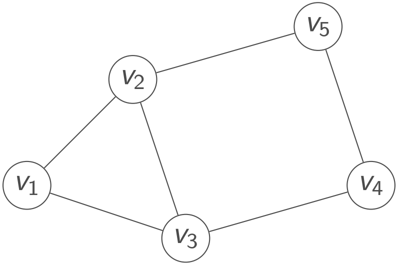
1. The other definition looks like this (Blondel et al. 2008):

$$Q = \frac{1}{2m}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(a_{ij} - \frac{d_i d_j}{2m}\right)\delta(c_i, c_j)$$

   where

   - m is the number of edges,
   - a_{ij} is the (i,j)th element of the adjacency matrix,
   - d_i is the degree of node i,
   - c_i is an indicator of the module to which node i belongs
   - δ is the Kronecker delta function.

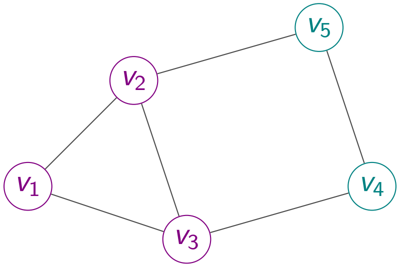
Consider the following graph



with adjacency matrix

$$\mathbf{A} = \begin{array}{c|ccccc} & v_1 & v_2 & v_3 & v_4 & v_5 \\ \hline v_1 & 0 & 1 & 1 & 0 & 0 \\ v_2 & 1 & 0 & 1 & 0 & 1 \\ v_3 & 1 & 1 & 0 & 1 & 0 \\ v_4 & 0 & 0 & 1 & 0 & 1 \\ v_5 & 0 & 1 & 0 & 1 & 0 \end{array}$$

Suppose we cluster this graph in to the following partition:



with adjacency matrix

$$\mathbf{A} = \begin{array}{c|ccccc} & v_1 & v_2 & v_3 & v_4 & v_5 \\ \hline v_1 & 0 & 1 & 1 & 0 & 0 \\ v_2 & 1 & 0 & 1 & 0 & 1 \\ v_3 & 1 & 1 & 0 & 1 & 0 \\ v_4 & 0 & 0 & 1 & 0 & 1 \\ v_5 & 0 & 1 & 0 & 1 & 0 \end{array}$$

From

$$\mathbf{A} = \begin{array}{c|ccccc} & v_1 & v_2 & v_3 & v_4 & v_5 \\ \hline v_1 & 0 & 1 & 1 & 0 & 0 \\ v_2 & 1 & 0 & 1 & 0 & 1 \\ v_3 & 1 & 1 & 0 & 1 & 0 \\ v_4 & 0 & 0 & 1 & 0 & 1 \\ v_5 & 0 & 1 & 0 & 1 & 0 \end{array}$$

We can create a new matrix that contains the *within- and between-module* sum of ties

$$\mathbf{B} = \begin{array}{c|cc} & C_1 & C_2 \\ \hline C_1 & 6 & 2 \\ C_2 & 2 & 2 \end{array}$$

Large numbers in the diagonals of this matrix mean that most ties connect nodes of the same module (strong clustering)

$$\mathbf{B} = \begin{array}{c} \\ C_1 \\ C_2 \end{array} \begin{array}{c} \begin{array}{cc} C_1 & C_2 \end{array} \\ \begin{bmatrix} 6 & 2 \\ 2 & 2 \end{bmatrix} \end{array}$$

Problem I: the numbers in the diagonal depend on the density of the network.

So, we divide each element of **B** by $2m$, where $m = 6$ is the number of edges in the network (why $2m$ and not m?).

$$\mathbf{B} = \begin{array}{c|cc} & C_1 & C_2 \\ \hline C_1 & 6 & 2 \\ C_2 & 2 & 2 \end{array}$$

Problem I: the numbers in the diagonal depend on the density of the network.

So, we divide each element of **B** by $2m$, where $m = 6$ is the number of edges in the network (why $2m$ and not m?).

$$\mathbf{E} = \begin{array}{c|cc} & C_1 & C_2 \\ \hline C_1 & \frac{6}{12} & \frac{2}{12} \\ C_2 & \frac{2}{12} & \frac{2}{12} \end{array}$$

The closer the sum over the diagonals to 1, the stronger would be the clustering.

$$\mathbf{E} = \begin{array}{c|cc} & C_1 & C_2 \\ \hline C_1 & \frac{6}{12} & \frac{2}{12} \\ C_2 & \frac{2}{12} & \frac{2}{12} \end{array}$$

Problem II: we could always throw *all* nodes into one module and get a diagonal sum of 1.

The matrix **E** would have only one cell which *is* the diagonal (and equal to 1).

$$\mathbf{E} = \begin{array}{c} \\ C_1 \\ C_2 \end{array} \begin{array}{cc} C_1 & C_2 \\ \left[\begin{array}{cc} \frac{6}{12} & \frac{2}{12} \\ \frac{2}{12} & \frac{2}{12} \end{array}\right] \end{array}$$

Problem II: we could always throw *all* nodes into one module and get a diagonal sum of 1.

The matrix **E** would have only one cell which *is* the diagonal (and equal to 1).

So, we need a *null model* to which we can to compare the clustering strength (i.e., switch to relative strength).

$$\mathbf{E} = \begin{array}{c} \\ C_1 \\ C_2 \end{array} \begin{array}{c} \begin{array}{cc} C_1 & C_2 \end{array} \\ \begin{bmatrix} \frac{6}{12} & \frac{2}{12} \\ \frac{2}{12} & \frac{2}{12} \end{bmatrix} \end{array} \begin{array}{c} \\ \frac{2}{3} \\ \frac{1}{3} \end{array}$$

Notice that the sum of the kth row of $\mathbf{E}$ gives us the proportion of edges that have at least one end-point in cluster k.

Denote this proportion as e_k.

If we would place the edges in the network at random, under the constraint that the resulting module sizes remain fixed, the *expected proportion* of ties in the (k, l)th cell of $\mathbf{E}$ would be $e_k \times e_l$.

So the *observed* and *expected* between-module tie-matrix looks like:

$$\mathbf{E} = \begin{array}{c|cc|c} & C_1 & C_2 & \\ C_1 & \frac{6}{12} & \frac{2}{12} & \frac{2}{3} \\ C_2 & \frac{2}{12} & \frac{2}{12} & \frac{1}{3} \end{array} \qquad \mathbb{E}[\mathbf{E}] = \begin{array}{c|cc|c} & C_1 & C_2 & \\ C_1 & \frac{4}{9} & \frac{2}{9} & \frac{2}{3} \\ C_2 & \frac{2}{9} & \frac{1}{9} & \frac{1}{3} \end{array}$$

So the *observed* and *expected* between-module tie-matrix looks like:

$$\mathbf{E} = \begin{array}{c|cc|c} & C_1 & C_2 & \\ \hline C_1 & \frac{6}{12} & \frac{2}{12} & \frac{2}{3} \\ C_2 & \frac{2}{12} & \frac{2}{12} & \frac{1}{3} \end{array} \qquad \mathbb{E}[\mathbf{E}] = \begin{array}{c|cc|c} & C_1 & C_2 & \\ \hline C_1 & \frac{4}{9} & \frac{2}{9} & \frac{2}{3} \\ C_2 & \frac{2}{9} & \frac{1}{9} & \frac{1}{3} \end{array}$$

The modularity is simply the sum of the diagonal elements of $\mathbf{E} - \mathbb{E}[\mathbf{E}]$, i.e.,

$$Q = \text{Trace}(\mathbf{E} - \mathbb{E}[\mathbf{E}]) = \sum_{k=1}^{K} (e_{kk} - e_k^2),$$

where e_{kk} is the kth diagonal element and e_k is the kth row-sum of the matrix $\mathbf{E}$.

Modularity Maximization

Newman and Grivan (2004) use a divisive algorithm to find a partition that maximizes Q

Blondel et al. (2008) use a multi-level agglomerative algorithm to find partitions that maximize Q

Implications

1. The modularity shows how "good" a clustering is, where "good" means better than a scenario where
   - ▷ ties are placed at random between the nodes of the network
   - ▷ given the constraint that the module sizes remain fixed

2. As the modularity is a sum of the difference of proportions, $-1 \leq Q \leq 1$.

Considerations

1. Suppose we have a network with two clusters of equal size, where all ties are within each module and no tie connects nodes from different modules. What would be the modularity?

Considerations

1. Suppose we have a network with two clusters of equal size, where all ties are within each module and no tie connects nodes from different modules. What would be the modularity?

   It would be $Q = 0.5$.

   - ▷ Half of the edges will be within-modules in the null model (i.e., by chance)
   - ▷ The upper bound of $Q = 1$ is reached only in networks with infinite clusters.

Considerations

1. Suppose we have a network with two clusters of equal size, where all ties are within each module and no tie connects nodes from different modules. What would be the modularity?

2. It is designed for *undirected graphs*

Considerations

1. Suppose we have a network with two clusters of equal size, where all ties are within each module and no tie connects nodes from different modules. What would be the modularity?

2. It is designed for *undirected graphs*

   There were suggestions to generalize it for directed graphs, some of them are more successful than others.

Considerations

1. Suppose we have a network with two clusters of equal size, where all ties are within each module and no tie connects nodes from different modules. What would be the modularity?

2. It is designed for *undirected graphs*

3. *Resolution limit*

Equivalence to Second Formula

The second expression of the modularity is given as

$$Q = \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(a_{ij} - \frac{d_i d_j}{2m} \right) \delta(c_i, c_j)$$

It's quite simple to show that this is the same as the expression from the previous slides.

Recall that we had

$$Q = \sum_{k=1}^{K} (e_{kk} - e_k^2) = \sum_k e_{kk} - \sum_k e_k^2.$$

We will focus of each of the term separately.

First term: $\sum_k e_{kk}$

Let $I_{ik} = 1$ if $c_i = k$ and zero otherwise, where $c_i = k$ means that node i belongs to module k.

Recall that e_{kk} is simply the proportion of ties that are in module k. So, we can write

$$e_{kk} = \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} I_{ik} I_{jk}$$

and summing over all k, we get

$$\sum_{k=1}^{K} e_{kk} = \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \delta(c_i, c_j).$$

Second term: $\sum_k e_k^2$

We note that

$$e_k^2 = \left(\frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} I_{ik} \right)^2 = \left(\frac{1}{2m} \sum_{i=1}^{n} d_i I_{ik} \right)^2$$
$$= \frac{1}{4m^2} \sum_{i=1}^{n} \sum_{j=1}^{n} d_i d_j I_{ik} I_{jk}$$

Summing this over all k gives, therefore,

$$\sum_k e_k^2 = \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{d_i d_j}{2m} \right) \delta(c_i, c_j).$$

Putting the terms together

Putting these two expressions together, we obtain

$$\begin{aligned} Q &= \sum_{k=1}^{K} e_{kk} - \sum_{k} e_k^2 \\ &= \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \delta(c_i, c_j) - \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{d_i d_j}{2m} \right) \delta(c_i, c_j) \\ &= \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(a_{ij} - \frac{d_i d_j}{2m} \right) \delta(c_i, c_j) \end{aligned}$$

as desired.

Code Length and the Map Equation

Minimum Description Length

This will be a *very superficial* overview of the minimum description length (MDL) principle and the map equation

Informally stated, the MDL principle states

> *Any regularity in the data can be compressed with the help of a model. The shorter our description of the data (with the help of the model) the better our model.*

Minimum Description Length

This will be a *very superficial* overview of the minimum description length (MDL) principle and the map equation

Informally stated, the MDL principle states

> *Any regularity in the data can be compressed with the help of a model. The shorter our description of the data (with the help of the model) the better our model.*

So, if

1. there are patterns in the data
2. and we use a “good” model

we'll be able to describe our data with a shorter code

(we won't deal with any concrete code here).

Preliminaries

Consider a discrete random variable X that takes on values $\{A, B\}$ with probability, respectively, p_A and p_B.

Assume we want to communicate a sequence of n independent realizations of this random variable to someone else without loss of information.

We have two codewords available to us, c_1 and c_2, which differ in length with $L(c_2) > L(c_1)$.

Preliminaries

Consider a discrete random variable X that takes on values $\{A, B\}$ with probability, respectively, p_A and p_B.

Assume we want to communicate a sequence of n independent realizations of this random variable to someone else without loss of information.

We have two codewords available to us, c_1 and c_2, which differ in length with $L(c_2) > L(c_1)$.

What code should we use for which outcome in order to make our code efficient as possible (minimize our code-length)?

The most efficient way would be to

1. assign c_1 to A if $p_A > p_B$.
2. assign c_1 to B if $p_B > p_A$.

i.e., we assign the *shorter codeword to the more likely outcomes*.

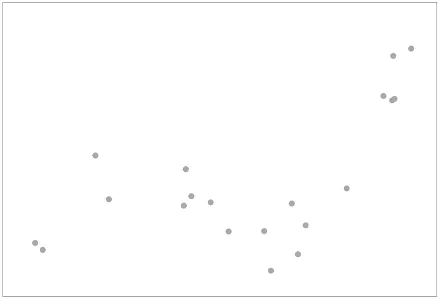
This is a way to exploit the “regularity” (difference in prob. of occurrence) in our random variable to shorten the code-length.

It can be shown that our the expected description length of n realizations of X will be the longest if $p_A = p_B$, i.e., if the "uncertainty" in the outcome is the largest.

In general, *more uncertain* the outcome, the *longer the code* that we need to describe n realizations of X.

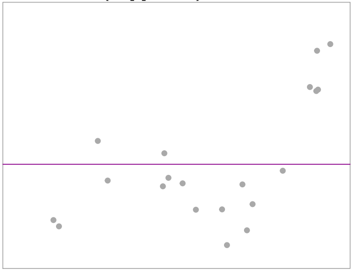
Example: Regression Problem

Suppose we have the following dataset and we want to describe how the *y*-values depend on *x*-values.

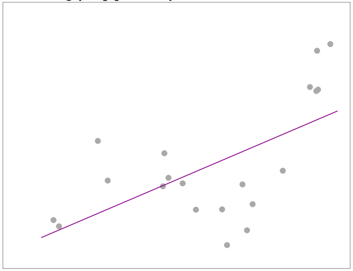


We'll try to capture the pattern in the data using a polynomial regression model.

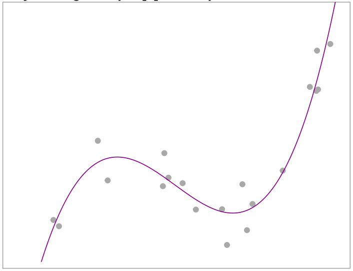
Mean Outcome (Var[e]: 0.0389)



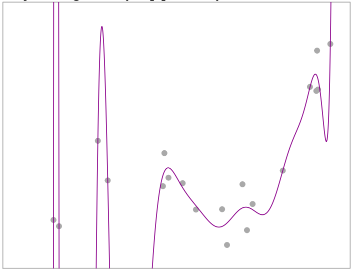
Linear Reg (Var[e]: 0.0241)



Poly. of Degree 3 (Var[e]: 0.0033)



Poly. of Degree 15 (Var[e]: 0.0019)



The uncertainty in the data decreases monotonically as our model becomes more complex (i.e, residual variance decreases).

In fact, it cannot increase with more included variables.

However, the *the description length* of the data (encoded with the help of our model) will first increase but, thereafter, decrease again.

Hence, minimizing the description length offers a way to find the "sweet spot" between underfitting and overfitting.

The uncertainty in the data decreases monotonically as our model becomes more complex (i.e, residual variance decreases).

In fact, it cannot increase with more included variables.

However, the *the description length* of the data (encoded with the help of our model) will first increase but, thereafter, decrease again.

Hence, minimizing the description length offers a way to find the “sweet spot” between underfitting and overfitting.

Let’s see why.

Let D be the data and M our model, then the total description length of the data (encoded with the “help” of our model) is

$$L(D) = L(D \mid M) + L(M).$$

While $L(D \mid M)$ decreases monotonically with the compexity in M, $L(M)$ increases monotonically.

By choosing a model that minimizes the description length, we choose a model which

1. shortens the description length of the data
2. but only if the gain in efficiency is larger than the description length of the model

The Map Equation

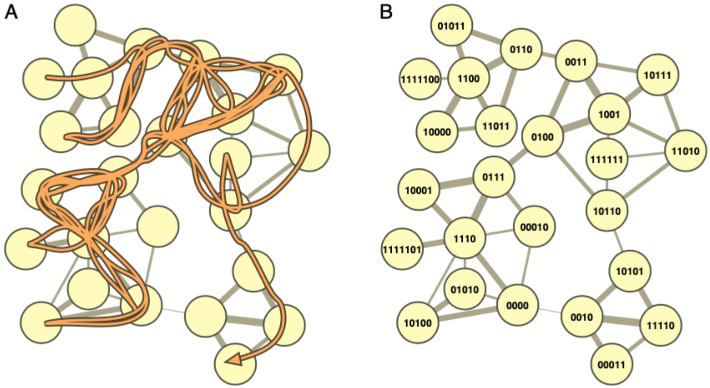
Consider describing an infinite-length random walk on a graph.

We need to describe where the random walker is at time t and, thus, need a code for each node in the graph.

Under reasonable conditions, the visiting probabilities will converge to a unique limiting distribution $\boldsymbol{\pi}_\infty$

The most efficient way to describe the trajectory of the random walker is by

1. assigning short codes to nodes with high visiting probabilities (elements of $\boldsymbol{\pi}_\infty$ that are close to 1)
2. long codes to nodes with low visiting probabilities



```
1111100 1100 0110 11011 10000 11011 0110 0011 10111 1001 0011
1001 0100 0111 10001 1110 0111 10001 0111 1110 0000 1110 10001
0111 1110 0111 1110 1111101 1110 0000 10100 0000 1110 10001 0111
0100 10110 11010 10111 1001 0100 1001 10111 1001 0100 1001 0100
0011 0100 0011 0110 11011 0110 0011 0100 1001 10111 0011 0100
0111 10001 1110 10001 0111 0100 10110 111111 10110 10101 11110
00011
```

Source: Rosvall & Bergstrom 2008 PNAS

Here is the crucial observation for community detection:

Just as we can use a regression model to capture the pattern in a dataset and, thereby, reduce the description length of the data, *we can exploit the clustering pattern in the network to reduce the description length of the random walk*.

Here is the crucial observation for community detection:

Just as we can use a regression model to capture the pattern in a dataset and, thereby, reduce the description length of the data, *we can exploit the clustering pattern in the network to reduce the description length of the random walk*.

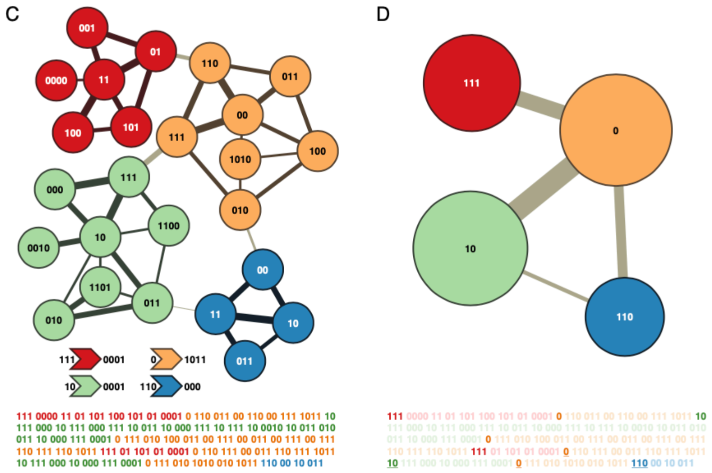
If the network is sufficiently clustered. We can

1. assign unique codes to *modules*
2. and repeatedly use the *same short codes* within each module

to reduce the length of the description.

As long as we are able to communicate when the random walker moves from one module to another, there is no ambiguity in the communication.

$$\boxed{L(D) = L(D \mid M) + L(M).}$$



Source: Rosvall & Bergstrom 2008 PNAS

The Map Equation

It's the same principle as we used in the polynomial regression example.

We don't need to assign "real" codes to the node. Instead, we rely on Shannon's (1948) Source Coding Theorems to obtain the *map equation*:

$$\begin{aligned} L(D) &= L(D \mid M) + L(M) \\ &= \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i) + q_{\curvearrowright} H(\mathcal{Q}). \end{aligned}$$

Considerations

1. MDL is a "general principle" to find good models. Accordingly, the Map Equation can be easily generalized to find more complex structures
   - ▷ Second-order Markov dyamics (Rosvall et al., 2014, Nature Communications)
   - ▷ Finding multi-level community structures (Rosvall and Bergstrom, 2011, Plos One)
   - ▷ Finding overlapping community structures (Esquivel and Rosvall, 2011, Physical Review X)
2. The Map Equation naturally incorporates the *direction of ties* in networks
3. If you are curious about MDL, read Grünwald (2007)