Block-Stochastic model

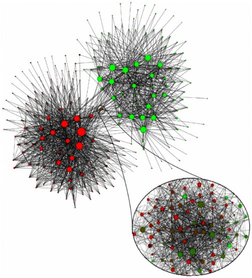
M. Mitrović Dankulov and A. Alorić

January 17, 2023

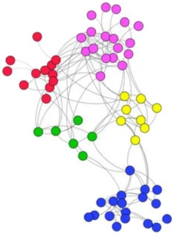
Outline

Real networks: mesoscopic heterogeneities

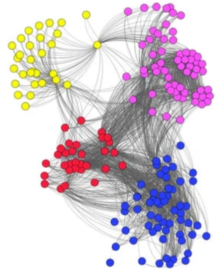
Social



Animal



Food web



Community detection algorithms: some

- Ravasz algorithm - similarity - bottom-up approach
- Girvan-Newman centrality - betweenness centrality - up-to-bottom approach
- Maximization of modularity:
  - Greedy algorithm
  - Louvain algorithm
- Infomap, OSLOM, etc.

Algorithm benchmarking

- Grivan-Newman (GN) benchmark - predefined size of communities, Erods-Renyi graphs, p_{in} and p_{out}
- Lancichinetti-Fortunato-Radicchi (LFR) Benchmark - community size $P(N_c) \sim N_c^{-\xi}$, degree distribution $P(k_i) \sim k_i^{-\gamma}$, $\mu = \frac{k^{ext}}{k^{ext}+k^{int}}$
- Can we find a model that is random but has community strucutre?

Stochastic-Block model

- Stochastic-Block model (SBM) is a simple graph, defined by a set of parameters $\theta = (c, \vec{z}, \mathcal{M})$:
  - c number of communities
  - $\vec{z}$ is a vector of size N (number of nodes), where element z_i shows community membership of nodes i
  - $\mathcal{M}_{rs}$ - an element shows a probability that node from community r is connected to community s
  - $\forall i, j$, $A_{ij} = 1$ if $\mathbb{M}_{rs}$, $A_{ij} = 0$ otherwise

- For $c == 1$, $M_{rs} == p$ - ER graph
- Networks within the community are ER graphs - $\mathcal{M}_{rs}$ for $r = s$
- Different connectivity between communities unlike *GN* benchmark model

SBM: properties

- It is a undirected binary but can be easly generalized to be directed
- Network is modular, and it depends on c and $\mathcal{M}$
- Degree distribution is a combination of Poisson distrbutions
- Small world - dimeter and shortest average path grow logarithmically with N
- Unclustered - clustering coefficient decreases with growth of network
- Giant component is comparable to network size for not too sparse networks

SBM: properties

- In SBM edges are mutually independent
- Nodes in the same community are e stochastically equivalent, they have the same connectivity patterns with other nodes
- A community in the SBM is a group of nodes with the same rules for connecting to nodes in other groups

SBM: generation

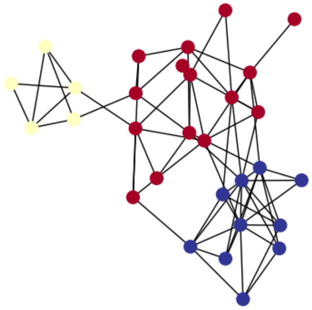
- 1. We set $\theta = (c, \vec{z}, \mathcal{M})$
- 2. We start with network G of N disconnected nodes
- 3. For a pair of nodes i and j we draw random number ξ_{ij}
- 4. If $\xi_{ij} \geq \mathcal{M}_{z_i z_j}$ we generated an edge between i and j
- Repeat steps 3. and 4. until we cover all nodes

SBM:exmaple

$c = 3;$

$z =$
$[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,$
$1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2];$

$$\mathcal{M} = \begin{pmatrix} 0.3 & 0.02 & 0.05 \\ 0.02 & 0.7 & 0.03 \\ 0.05 & 0.03 & 0.5 \end{pmatrix}$$



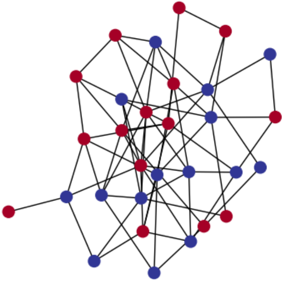
SBM: varitation

- SBM has large number of parameters - very flexible and can create large number of connectivity patterns
- Assortative pattern: $\mathcal{M}rs = q$ if $r! = s$, and $\mathcal{M}rr = p_r$ where $\vec{p} = (p_1, \ldots, p_c) == p$
- One parameter model $p_1 == p_2 == \ldots == p_c$, $N_i = \frac{N}{c}$, $c = 2$ and we fix $< k >$
- In one parameter model p and p are not independents; if we increase p we need to decrease q in order to keep fixed $< k >$; $p = \frac{<k>+\frac{\epsilon}{2}}{N}$ and $q = \frac{<k>-\frac{\epsilon}{2}}{N}$; by varying ϵ we change network connectivity

SBM one parameter: example

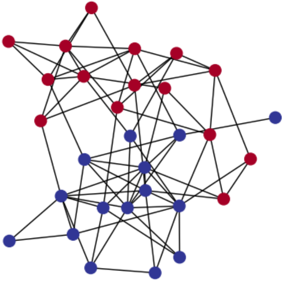
$$N = 30, < k >= 5, \ c = 2$$

$\epsilon = 2$



$p = 0.2; q = 1.333$

$\epsilon = 6$



$p = 2.667; q = 0.067$

Degree corrected SBM

- SB model creates ER graph with mesoscopic heterogeneities

- Degree corrected SBM (DC-SBM) creates graph with specified degree structure and mesoscopic heterogeneities

- DC-SBM parameters: $\theta(c, \vec{z}, \vec{k}, \mathcal{M})$
- $\vec{k}$ is expected degree sequence

- $\mathcal{M}$ is not a probability matrix but a counting matrix: $\mathcal{M}_{rs} = \sum_{i,j} A_{ij}\delta_{z_i,r}\delta_{z_j,s}$ - number of edges between r and s; if $r == s$ then it is doble number of edges

- DC-SBM is a *multigraph* - A_{ij} equals the number of links between nodes

DC-SBM: generation

- Groups can behave as super-nodes; degree of a group $k_r = \sum_i k_i \delta_{z_i,r}$
- Node propensity $\gamma_i = \frac{k_i}{k_{z_i}}$
- $\forall i > j$ $A_{ij} = A_{ji} = Poisson(\gamma_i \gamma_j \mathcal{M}_{z_i z_j})$ - expected number of edges between nodes i and j

DC-SBM: generation

- Generate DC-SBM network:
  - 1. Compute group level degree k_r
  - 2. Compute node propensity γ_r
  - 3. Start with network of N disconnected nodes
  - 4. For $i > j$ generate random nuber ξ from $Poisson(\gamma_i \gamma_j \mathcal{M}_{z_i z_j})$
  - 5. If $\xi > 0$ create undirected edge between i and j
  - 6. Repeat 4. and 5. until you cover all pairs of nodes

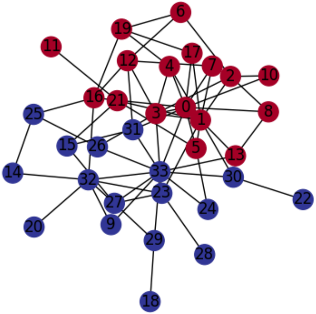
DC-SBM: example

Zachary's Karate Club network
$c = 2;\ N = 34$

$\vec{z} = [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$;

$\vec{k} = [16, 9, 10, 6, 3, 4, 4, 4, 5, 2, 3, 1, 2, 5, 2, 2, 2, 2, 2, 3, 2, 2, 2, 5, 3, 3, 2, 4, 3, 4, 4, 6, 12, 17]$

$$\mathcal{M} = \begin{pmatrix} 70 & 11 \\ 11 & 64 \end{pmatrix}$$



Statistical inference

- Probabilistic generative model $Pr(G|\theta)$
- When we know or set θ we are able by tosing a coin for each pair of nodes i and j to generate network G
- ER networks - $\theta = p$
- SBM - $\theta = (c, \vec{z}, \mathcal{M})$ or DC-SBM $\theta = (c, \vec{z}, \vec{k}, \mathcal{M})$
- We set θ and generate network using $Pr(\theta)$
- Inference is to find θ based on network data

Statistical inference

- Based on network G we can infer a model $Pr(G|\hat{\theta})$ to:
  - To generate synthetic data
  - If parameters have a meaning we can discover principles that drive network evolution
  - Can give us insights about network organization (community structure, mixing patterns)
  - Compare competing network models
  - Node and edge attribute prediction; edge prediction

Likelihood

- $Pr(G|\theta)$ defines a probability distribution for network occurrence for given θ
- $\mathcal{L}(G|\theta)$ - likelihood function that represents probability of random network realizations conditional on particular values of θ
- We use maximum value of $\mathcal{L}(G|\theta)$ to estimate the value of $\hat{\theta}$
- $\hat{\theta}$ is our best estimate of model parameters

Likelihood for SBM

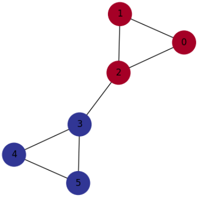
$$
\begin{aligned}
\mathcal{L}(G|\vec{z},\mathcal{M}) &= \prod_{i,j} Pr(i \to j|\vec{z},\mathcal{M}) \\
&= \Big(\prod_{(i,j)\in E} Pr(i \to j|\vec{z},\mathcal{M})\Big)\Big(\prod_{(i,j)\notin E} (1 - Pr(i \to j|\vec{z},\mathcal{M}))\Big) \\
&= \Big(\prod_{(i,j)\in E} \mathcal{M}_{z_i z_j}\Big)\Big(\prod_{(i,j)\notin E} (1 - \mathcal{M}_{z_i z_j})\Big)
\end{aligned}
\tag{1}
$$

Maximum likelihood estimate for SBM

- $e_{rs} = \sum_{i<j} A_{ij}\delta_{z_i r}\delta_{z_j s}$ and $n_{rs} = \sum_{i<j} \delta_{z_i r}\delta_{z_j s}$
- $\mathcal{L}(G|\vec{z}, \mathcal{M}) = \prod_{r,s}(\mathcal{M}_{rs})^{e_{rs}}(1 - \mathcal{M}_{rs})^{n_{rs}-e_{rs}}$
- $\hat{\mathcal{M}}_{rs} = \frac{e_{rs}}{n_{rs}}$ this is the best estimate
- Log-likelihood $\ln(\mathcal{L}(G|\vec{z}, \mathcal{M})) = \sum_{r,s} e_{rs} \ln \frac{e_{rs}}{n_{rs}} + (n_{rs} - e_{rs}) \ln \frac{n_{rs}-e_{rs}}{n_{rs}}$

Maximum likelihood estimate for SBM: example

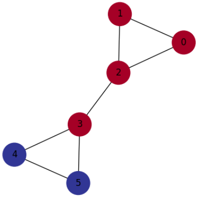
Good



e_{rs}/n_{rs}	red	blue
red	3/3	1/9
blue	1/9	3/3

$$\mathcal{L} = 0.043304\ldots$$
$$\ln \mathcal{L} = -3.1395\ldots$$

Bad



e_{rs}/n_{rs}	red	blue
red	4/6	2/8
blue	2/8	1/1

$$\mathcal{L} = 0.000244\ldots$$
$$\ln \mathcal{L} = -8.3178\ldots$$

Likelihood DC-SBM

$$
\begin{aligned}
\mathcal{L}(G|\vec{z},\gamma,\mathcal{M}) &= \prod_{i,j} Poisson(\gamma_i\gamma_j\mathcal{M}_{z_i z_j}) \\
&= \prod_{i<j} \frac{(\gamma_i\gamma_j\mathcal{M}_{z_i z_j})^{A_{ij}}}{A_{ij}!} \exp(-\gamma_i\gamma_j\mathcal{M}_{z_i z_j}) \cdot \\
&\cdot \prod_{i} \frac{(\frac{1}{2}\gamma_i^2\mathcal{M}_{z_i z_i})^{A_{ii}/2}}{(A_{ii}/2)!} \exp(-\frac{1}{2}\gamma_i^2\mathcal{M}_{z_i z_i})
\end{aligned} \tag{2}
$$

Maximum likelihood estimate for DC-SBM

- $\hat{\gamma}_i = \frac{k_i}{k_{z_i}}$
- $\hat{\mathcal{M}}_{rs} = \omega_{rs}$
- $\omega_{rs} = \sum_{i,j=1}^{N} A_{ij}\delta_{z_i r}\delta_{z_j s}$ and $k_r = \sum_s \omega_{rs} = \sum_i k_i \delta_{z_i r}$
- $\ln \mathcal{L} = \sum_{r,s} \omega_{rs} \ln \frac{\omega_{rs}}{k_r k_s}$

Finding a good partition

- By finding the estimates of SBM and DC-SBM model parameters on $\vec{z}$ we have only solved part of the problem
- We need to find the "good" partition or we need to find $\vec{z}$
- Potential algorithms: Markov chain Monte Carlo, expectation-maximization, belief propagation, etc.
- Locally greedy heuristic (LGH), a generalized Kernighan-Lin algorithm for solving the minimum-cut graph partitioning problem

LGH: initialization

- Create $\vec{z}_0$ by assigning each node to a one of the c partitions uniform at random
- Calculate and record its log-likelihood $L_0 = \mathcal{L}_0$
- Set $t = 1$ and set all nodes to be *unfrozen*

LGH: run a phase

- Phase lasts until $t < N$ - all nodes become *frozen*
- Copy last partition from step $t-1$ $z_t = z_{t-1}$
- For each of the $n-t$ unfrozen nodes in step t:
  - s is the curent group of node i
  - We move node i to group $r \in \{1, 2, \ldots, c\} - s$ and calculate log-likelihood for that new partition and remember it as $(i, L^i_{s\to r}, r)$
  - of $(i, L^i_{s\to r}, r)$ keep the one with the most positive log-likelihood and store it in a set
  - reset z_t (move i back to s)
- **Greedy choice**- of the $(n-t)$ combinations (i, L, r), one for each unfrozen node, select the one with the most positive L
- **Freez that node**: set group of node i to be r, set L_t to be a new log-likelihood, and move to step $t+1$

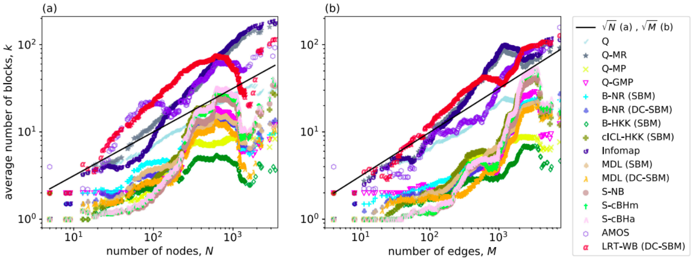
LGH: evaluate the phase

- Among the $(n+1)$ log-likelihoods $L_0, \ldots, L_n$ identify the largest L^* and corresponding partition $\vec{z^*}$
- If $L^* \leq L_0$ then keep $\vec{z_0}$ and STOP
- If $L^* > L_0$ set $\vec{z_0} = \vec{z^*}$, $L_0 = L^*$ and start a new phase
- LGH is only guaranteed to converge on a local optimum
- To be sure, you need to run this algorithm several times; the more the better

Number of communities - c

- So far c was a free parameter
- Model regularizations: as we increase c the better the model should fit the data; if model for c is not much better than one for $c-1$ it is not worth it; there are many different ways to regularize a model

Community detection dilema



How we define what communities and the choice of an algorith have an enormous influence on what communities we find

Source: https://aaronclauset.github.io/courses/5352/csci5352_F22_L8.pdf

Further reading

Aaron Clauset Lecture notes:

- Lecture 7: https://aaronclauset.github.io/courses/5352/csci5352_F22_L7.pdf
- Lecture 8: https://aaronclauset.github.io/courses/5352/csci5352_F22_L8.pdf