The Stochastic Block Model

Caterina De Bacco and Isabel Valera

1 The stochastic block model: the problem

After studying several approximations for performing inference in probabilistic modeling, we now focus on one popular application: the stochastic block model (SBM). This is a model aimed at clustering network data, in other words, it is a type of community detection on networks.

Imagine you have a dataset of pairwise interactions between individuals. For instance, consider a social network where people interact through friendship or other social relationship. This dataset can be represented using a so called *adjacency* matrix $\mathbf{A}$ such that the entries $A_{ij} = 1$ if person i is a friend of person j and 0 otherwise. This can be represented as a network where people are nodes and there is an edge between two people if they interact (i.e. if they are friends in this example). The network is represented as $G(V, E)$, where V is the set of nodes and E the set of edges. Typically, the number of nodes is denoted as $N = |V|$; an edge is denoted as (i, j). See figure 1 for an example of such network.

Objective: cluster people based on their pattern of interaction.
Obs: the notion of similarity is arbitrary based on the application. Here we assume that two people are similar if they interact in a similar way.

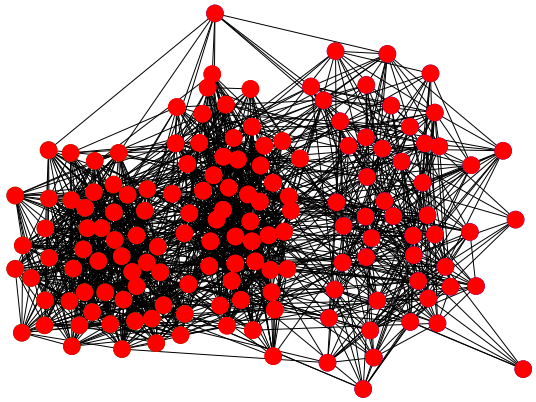

Figure 1: Example of a SBM-generated graph.

1.1 Stochastic equivalence

An approach to address this problem is to assume that people interact based on their membership to a particular community. For instance, people that play a particular sport are more likely to interact

with people that also play that sport. This pattern is called *assortativity*. The opposite scenario, where two people belonging to *different* community are more likely to interact, is called *disassortativity*. See figure 2 for an example of these cases. Other types of topologies are also possible.

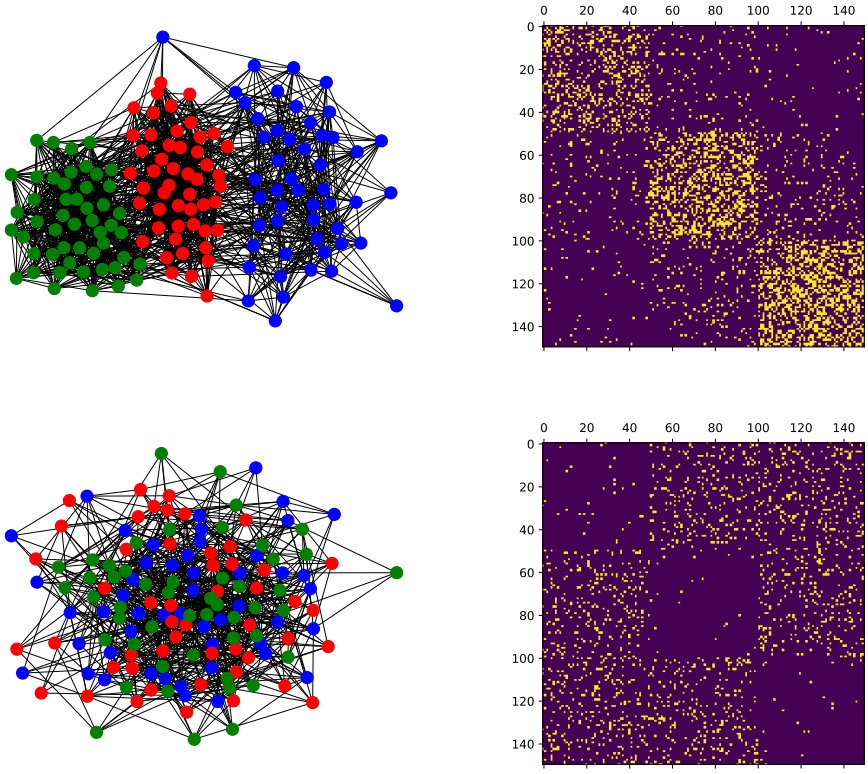

Figure 2: Example of a SBM-generated graph. Top) Assortative structure, this is also the one corresponding to Figure 1. Bottom) Disassortative structure. On the right we have the adjacency matrix.

Regardless the structure type, this problem can be formalized by introducing a set of hidden variables, the memberships:

$$q_i \in \{1, \dots, K\} \quad \text{for} \quad i = 1, \dots, N \quad , \tag{1}$$

where K is the number of groups/blocks/communities. Now, given two people i and j, we can make the assumption that the probability of them interacting, depends only upon their membership q_i and q_j:

$$Pr\left((i,j) \in E\right) = f\left(q_i, q_j\right) \quad . \tag{2}$$

This is the concept of *stochastic equivalence*. Under this notion, if i and k belong to the same community q, they have the same probability of interacting with a third person j. In other words, the pattern of interactions is completely captured by the hidden variables, no other microscopic details about the single people matter.
All we need to know to model the network based on the observed data, is the set of memberships and the interaction pattern, i.e. how interactions happen between members of the same or different groups. This is regulated formally by a $K \times K$ matrix $\mathbf{C}$ called *affinity* matrix where the entries C_{kq} are proportional to the probability that a member of group k interacts with a member of the group q.
The SBM is a probabilistic model that considers two set of parameters, the membership $\mathbf{q} = (q_1, \dots, q_N)$ and the affinity matrix $\mathbf{C}$, and makes the stochastic equivalence assumption.

Objective: estimating the parameters $\theta := (\mathbf{q}, \mathbf{C})$, the membership and affinity matrix, given the observed interactions, the adjacency matrix $\mathbf{A}$, i.e. performing inference.
Once we have the membership, we can cluster people based on where they belong to, which was our original goal.
Obs1: thanks to the group assignments and the affinity matrix, the model is flexible as it can represent different structures.
Obs2: is analytically tractable and computationally scalable. Several inference methods can be applied.

1.2 Choosing the number of groups K

In all this lecture we assume that K is given or known.
In real situation this is not the case and we should write explicitly K as an additional parameter of the model, i.e. $\theta := (\mathbf{q}, \mathbf{C}, K)$. However, its inference is usually different than that of the other hidden variables. One in fact treats the problem of choosing the best K as a *model selection* problem in and of itself.
Popular ways to perform model selection are calculating metrics like AIC, BIC or minimum description length. These calculate the cost of the amount of information needed compare to the likelihood score.
Another popular approach is to perform cross-validation: split the dataset into train and test sets. Run inference on the train and use the parameters obtained there to calculate performance metric in the test set. Repeat for different values of K and choose the one with best performance on the test.
A different approach is to consider a Bayesian formalism with hyper-parameters that depend on K explicitly, and thus calculate the posterior $P(K|\mathbf{A})$ by marginalizing out all the remaining parameters from the full posterior, see for instance Newman and Reinert (2016) or Peixoto (2017) for more details and Figure 3 for an example result on real networks.

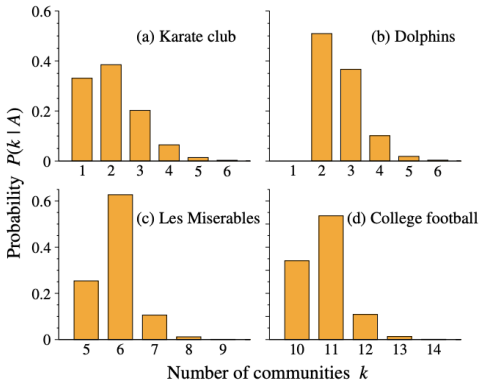

Figure 3: Example of finding the best K. This approach calculates the posterior $P(K|\mathbf{A})$ using a Bayesian approach. Figure taken from Newman and Reinert (2016).

1.3 The standard model (binary entries)

Given the group assignments $\mathbf{q}$, the edges are generated independently and the probability of their existence is:

$$Pr\left((i,j) \in E\right) = C_{q_i q_j} \quad . \tag{3}$$

For simplicity, in all the following, we do not allow self-edges, i.e. $A_{ii} = 0$, and consider undirected network, i.e. symmetric $\mathbf{A}$. Formally, we can define a likelihood function for the data considering a Bernoulli distribution as:

$$P(\mathbf{A}|\theta) = \prod_{i<j} \text{Bern}\left(A_{ij}; C_{q_i q_j}\right) \tag{4}$$

$$= \prod_{i<j} C_{q_i q_j}^{A_{ij}} (1 - C_{q_i q_j})^{(1-A_{ij})} \quad . \tag{5}$$

1.4 Inference: Maximum Likelihood approach

If we assume no prior knowledge about the group assignments, i.e. they are distributed uniformly at random, we can find the parameters by maximizing the likelihood (5). By defining n_r the number of nodes in community r, such that $N_{rs} = n_r n_s$ is the total number of possible edges between group r and s, we can rewrite the likelihood changing the product series to run over all pairs of groups, rather than over all pairs of vertices, thanks to the stochastic equivalence:

$$P(\mathbf{A}|\theta) = \prod_{r,s} C_{rs}^{m_{rs}} (1 - C_{rs})^{n_{rs} - m_{rs}} \quad , \tag{6}$$

where m_{rs} is the number of edges between group r and s. Taking the derivative w.r.t C_{rs} we get:

$$\frac{\partial P(\mathbf{A}|\theta)}{\partial C_{rs}} = C_{rs}^{m_{rs}} (1 - C_{rs})^{n_{rs} - m_{rs}} \left[\frac{m_{rs}}{C_{rs}} - \frac{N_{rs} - m_{rs}}{1 - C_{rs}} \right] \quad , \tag{7}$$

setting it to zero gives:

$$\hat{C}_{rs} = \frac{m_{rs}}{N_{rs}} \quad . \tag{8}$$

We can substitute it inside (6) to get:

$$P(\mathbf{A}|\theta) = \prod_{r,s} \left(\frac{m_{rs}}{N_{rs}} \right)^{m_{rs}} \left(1 - \frac{m_{rs}}{N_{rs}} \right)^{n_{rs} - m_{rs}} \quad . \tag{9}$$

Taking the logarithm yields:

$$\log P(\mathbf{A}|\theta) = \sum_{r,s} \left[m_{rs} \log \left(\frac{m_{rs}}{N_{rs}} \right) + (n_{rs} - m_{rs}) \log \left(1 - \frac{m_{rs}}{N_{rs}} \right) \right] \quad . \tag{10}$$

Applying the rules of logarithm and collecting terms yields:

$$\log P(\mathbf{A}|\theta) = \sum_{r,s} \left[m_{rs} \log m_{rs} + (n_{rs} - m_{rs}) \log (n_{rs} - m_{rs}) - n_{rs} \log n_{rs} \right] \quad , \tag{11}$$

which is a function only of the counts induced by $\mathbf{q}$. We further have the constraints that:

$$\sum_{q=1,\ldots,Q} n_q = N \tag{12}$$

$$\sum_{r<s=1,\ldots,Q} m_{rs} = |E| \quad , \tag{13}$$

again assuming undirected graphs, i.e symmetric $\mathbf{A}$.

Example. See for instance the network of Figure 4. Using Eq. (11), the partition on the left has $P(\mathbf{A}|\theta) = 0.0433$ whereas the one on the right has $P(\mathbf{A}|\theta) = 0.0002$, i.e. the one on the left has much higher likelihood than the other.

Inference can be performed by a greedy algorithm based on performing Monte Carlo moves described in Karrer and Newman (2011) as follows:

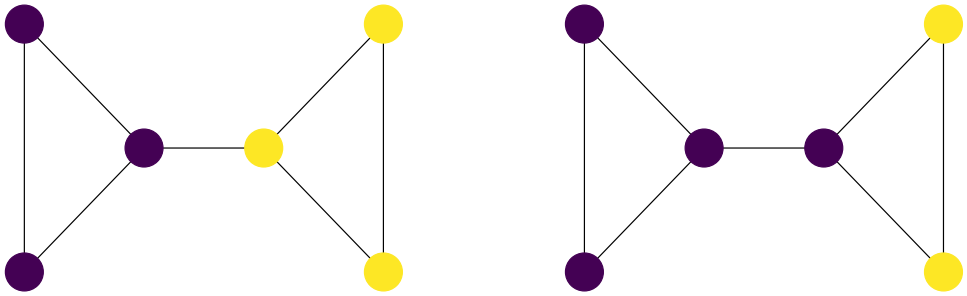

Figure 4: Example of network with two triangles.

1. Initialize the system with groups assignments $\mathbf{q}$ extracted uniformly at random.
2. Select a candidate node i and assign a new group q_i^{new} based on the group q that maximizes the change in log likelihood $\Delta L = \log P(\mathbf{A}|\theta^{new}) - P(\mathbf{A}|\theta^{old})$. Notice that it can also be negative, i.e. there can be moves that decrease the likelihood.
3. Repeat for all nodes.
4. Take the state that had highest log likelihood and set this as the starting point for a new iteration.
5. Repeat until the log likelihood does not increase anymore (convergence).

Another approach is to use MCMC Metropolis-Hasting methods as we saw in the previous tutorials. The simplest implementation of this protocol for the inference of SBMs is to start from a random partition $\mathbf{q}_0$, and use move proposals where a node i is randomly selected, and then its new group membership q_i' is chosen randomly between all $K+1$ choices (where the remaining choice means we populate a new group) according with this probability:

$$P(q_i'|\mathbf{q}) = \frac{1}{K+1} \quad . \tag{14}$$

One then has to decide whether to make the move or not, based on the probability ratio $P(\mathbf{q}'|\mathbf{A})/P(\mathbf{q}|\mathbf{A})$, which determines how much "it costs" to make the move. Calculating $P(\mathbf{q}'|\mathbf{A})/P(\mathbf{q}|\mathbf{A})$ is usually efficient (we do not need to calculate normalization constants, they cancel out).
Another approach is to use Variational methods like Variational Expectation Maximization (VEM). We will cover this in the next Lectures.

1.5 Weighted network

If the network is weighted, so that $\mathbf{A}$ is not binary anymore but can have other positive and discrete values, it is better to consider the Poisson distribution for the likelihood:

$$P(\mathbf{A}|\theta) = \prod_{i<j} \text{Pois}\left(A_{ij}; C_{q_i q_j}\right) \tag{15}$$

$$= \prod_{i<j} \frac{e^{-C_{q_i q_j}} C_{q_i q_j}^{A_{ij}}}{A_{ij}!} \quad . \tag{16}$$

Following similar reasonings to before, i.e. using the stochastic equivalence, we get:

$$P(\mathbf{A}|\theta) = \prod_{r<s} \frac{e^{-N_{rs} C_{rs}} C_{rs}^{m_{rs}}}{A_{ij}!} \quad . \tag{17}$$

Proceeding again in an MLE fashion, we get by simple differentiation:

$$\hat{C}_{rs} = \frac{m_{rs}}{N_{rs}} \quad , \tag{18}$$

which is the same as for the Bernoulli (binary) case. Substituting back into the likelihood, taking the log and ignoring constants we get the (unnormalized) loglikelihood:

$$\log P(\mathbf{A}|\mathbf{q}, \hat{C}) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{N_{rs}} \quad . \tag{19}$$

1.5.1 MLE interpretation

Let's now rewrite that expression in a different way by applying some simple algebra:

$$\log P(\mathbf{A}|\mathbf{q}, \hat{C}) = \sum_{rs} \frac{m_{rs}}{2|E|} \log \frac{m_{rs}/2|E|}{N_{rs}/N^2} \quad , \tag{20}$$

where we have neglected irrelevant constants. If we now pick one edge uniformly at random, the probability that its end nodes have group assignment r and s is $p_K(r,s) = \frac{m_{rs}}{2|E|}$. If the network had the same group assignment but edges where assigned completely at random, i.e. with no dependence on the group membership, this probability would be equal to N_{rs}/N^2 instead. This is the denominator of the logarithm inside Eq. (20). We call this second distribution $p_0(r,s)$. We can then rewrite:

$$\log P(\mathbf{A}|\mathbf{q}, \hat{C}) = \sum_{rs} p_K(r,s) \log \frac{p_K(r,s)}{p_0(r,s)} \quad , \tag{21}$$

which is the KL divergence between the two distributions. We can conclude that the most likely group assignments under the ordinary stochastic blockmodel are then those assignments that require the most information to describe starting from a model that does not have group structure.

1.6 Belief Propagation approach

Another approach is to apply Belief Propagation as done in Decelle *et al.* (2011). We can introduce a further parameter γ_q, the probability that a node takes group q, i.e. $\gamma_q = P(q_i = q)$. In the limit of large N, we have $\gamma_q = \frac{N_q}{N}$. Since we are interested in sparse networks where $C_{kq} = O(1/N)$, it is better to parametrize the affinity matrix with entries $C'_{kq} = N C_{kq}$.

We then have:

$$P(\mathbf{A}, \mathbf{q}|\mathbf{C}, \boldsymbol{\gamma}) = P(\mathbf{A}|\mathbf{q}, \mathbf{C}) P(\mathbf{q}|\mathbf{C}, \boldsymbol{\gamma}) \tag{22}$$

$$= \prod_{i<j} \left(\frac{C'_{q_i q_j}}{N} \right)^{A_{ij}} \left(1 - \frac{C'_{q_i q_j}}{N} \right)^{(1-A_{ij})} \prod_i \gamma_{q_i} \quad . \tag{23}$$

Obs1: if we now want to *only* infer the group assignment $\mathbf{q}$ *given* observed network and remaining parameters, we need to calculate $P(\mathbf{q}|\mathbf{A}, \mathbf{C}, \boldsymbol{\gamma})$. This is denoted as *BP-Inference*.
Obs2: a complementary problem would be that of learning the other parameters $(K, \mathbf{C}, \boldsymbol{\gamma})$, a problem denoted as *BP-Learning*.

As we already saw in the previous lectures, $P(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma}) = \frac{P(\mathbf{A},\mathbf{q}|\mathbf{C},\boldsymbol{\gamma})}{\sum_{\mathbf{q}} P(\mathbf{A},\mathbf{q}|\mathbf{C},\boldsymbol{\gamma})}$ can be written as a Boltzmann probability with Hamiltonian:

$$H(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma}) = -\sum_i \log \gamma_{q_i} - \sum_{i<j} \left[A_{ij} \log C'_{q_i q_j} + (1 - A_{ij}) \log \left(1 - \frac{C'_{q_i q_j}}{N} \right) \right] \quad . \tag{24}$$

In the sparse case $C'_{kq} = O(1)$, there are strong $O(1)$ interactions between connected nodes, i.e. $A_{ij} = 1$, and weak $O(1/N)$ between non-connected nodes.

The Boltzmann distribution is:

$$\mu(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma}) = P(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma}) = \frac{e^{-H(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma})}}{Z(\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma})} \tag{25}$$

Obs1: if the parameters $K, \boldsymbol{\gamma}, \mathbf{C}'$ are known, then the Boltzmann distribution (25) is asymptotically uniform over all configurations with the right group sizes and the right number of edges between each pair of groups, $N_q/N = \gamma_q$ and $m_{kq}/N = C'_{kq}\gamma_k\gamma_q$. The original, correct group assignment is just one of these configurations. In a statistical physics sense, the original group assignment is an equilibrium configuration for the Boltzmann distribution, rather than its ground state. In particular, if we were presented with the original assignment $\mathbf{q}^{GT}$ and a typical assignment $\mathbf{q}^{sample}$ sampled according to the Boltzmann distribution, we would be unable to tell which one was correct.

The marginals of the Boltzmann distribution, i.e. the probabilities $\nu(q_i)$ that a node i belongs to a group q_i are:

$$\nu_i(q_i) = \sum_{q_j, j \neq i} \mu(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma}) \quad . \tag{26}$$

The estimate of the original group assignment is:

$$q_i^* = \arg\max_{q_i} \nu_i(q_i) \quad , \tag{27}$$

which corresponds to the most-likely group. This is also called *maximum posterior marginal*.
Obs2: the set of q_i^* for $i = 1, \ldots, N$ may not correspond to the ground state (minimum) of the Hamiltonian $H(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma})$, i.e. the configuration that maximizes $\mu(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma})$! In general, the latter has slightly smaller overlap with the original assignment.
Obs3: the Boltzmann distribution is symmetric with respect to permutations of the group labels.

Now we are ready to write the BP equations and denote with $c_{kq} := C'_{kq}$ for simplicity of notation:

$$\psi_{q_i}^{i \to j} \propto \gamma_{q_i} \prod_{k \in \partial i \setminus j} \left[\sum_{q_k} c_{q_i q_k}^{A_{ik}} \left(1 - \frac{c_{q_i q_k}}{N}\right)^{1 - A_{ik}} \psi_{q_k}^{k \to i} \right] \quad . \tag{28}$$

As usual, we iterate until convergence and then we get the marginals:

$$\psi_{q_i}^{i} \propto \gamma_{q_i} \prod_{k \in \partial i} \left[\sum_{q_k} c_{q_i q_k}^{A_{ik}} \left(1 - \frac{c_{q_i q_k}}{N}\right)^{1 - A_{ik}} \psi_{q_k}^{k \to i} \right] \quad . \tag{29}$$

Notice that the explicit presence of N now helps simplifying the equations. In fact, we can assume that the messages sent from i to all its non-neighbors are equal. Consider $(i, j) \notin E$:

$$\begin{aligned} \psi_{q_i}^{i \to j} &\propto \gamma_{q_i} \prod_{k \in \partial i} \left[\sum_{q_k} c_{q_i q_k} \psi_{q_k}^{k \to i} \right] \prod_{k \notin \partial i \setminus j} \left(1 - \frac{1}{N} \sum_{q_k} c_{q_i q_k} \psi_{q_k}^{k \to i} \right) & (30) \\ &= \psi_{q_i}^{i} + O(1/N) \quad . & (31) \end{aligned}$$

Instead, if $(i, j) \in E$:

$$\begin{aligned} \psi_{q_i}^{i \to j} &\propto \gamma_{q_i} \prod_{k \in \partial i \setminus j} \left[\sum_{q_k} c_{q_i q_k} \psi_{q_k}^{k \to i} \right] \prod_{k \notin \partial i} \left(1 - \frac{1}{N} \sum_{q_k} c_{q_i q_k} \psi_{q_k}^{k \to i} \right) & (32) \\ &= \gamma_{q_i} e^{-h_i} \prod_{k \in \partial i \setminus j} \left[\sum_{q_k} c_{q_i q_k} \psi_{q_k}^{k \to i} \right] \quad , & (33) \end{aligned}$$

where we have neglected terms that contribute $O(1/N)$ and we have defined the auxiliary external field:

$$h_i := \frac{1}{N} \sum_k \sum_{q_k} c_{q_i q_k} \psi^k_{q_k} \quad . \tag{34}$$

In practice the algorithmic updates are done by:

1. Update all the K components of the messages $\psi^{i\to j}_{q_i}$ using Eq. (33);
2. Compute $\psi^i_{q_i}$ using Eq. (29);
3. Update the field h_i using Eq. (34).

Obs1: at convergence of BP, one can calculate the Bethe free energy using the converged messages and other parameters. This can be used to calculate useful quantities, like the optimal number of groups K.
Obs2: after convergence, one can optionally update the other parameters $\boldsymbol{\gamma}, \mathbf{C}$ by maximizing the posterior $P(\mathbf{q}|\mathbf{A}, \mathbf{C}', \boldsymbol{\gamma})$ with respect to $\boldsymbol{\gamma}$ or $\mathbf{C}$. Then repeat with BP cycle and so on until convergence of $\boldsymbol{\gamma}, \mathbf{C}$. This is a typical EM routine for performing *BP-learning* where the expectation step is made using BP, but other scheme can be considered as well, e.g. MCMC.
Obs3: the BP approach is highly scalable, with a running time that is linear in the size of the network.

1.7 Extra: Bayesian approach

We can extend the ML approach by considering priors for the parameters; we will follow the model proposed in Nowicki and Snijders (2001). We then assume that the memberships are i.i.d. random variables that follow a multinomial distribution with parameter $\boldsymbol{\gamma}$, where $\gamma_q = P(q_i = q)$ is the prior probability that a node is in group q. If we denote $\mathbf{z_i} = (z_{i1}, \ldots, z_{iK})$ a vector with all 0 entries except one equal to 1, for the group assignment of node i, we can then write:

$$\mathbf{z}_i \sim \text{Mult}(\mathbf{z}_i; \boldsymbol{\gamma}) = \frac{K!}{\prod_k \gamma_k!} \prod_{k=1}^{K} \gamma_k^{z_{ik}} \quad . \tag{35}$$

For the affinity matrix's entries C_{kq}, we use a Beta prior with parameters (a_{kq}, b_{kq}):

$$C_{kq} \sim \text{Beta}(C_{kq}; a_{kq}, b_{kq}) = \frac{\Gamma(a_{kq} + b_{kq})}{\Gamma(s_{kq})\Gamma(b_{kq})} C_{kq}^{a_{kq}-1} (1 - C_{kq})^{b_{kq}-1} \tag{36}$$

The hyperparameters $a, b, \boldsymbol{\gamma}$ can be fixed a priori or also be given an hyperprior. For instance, we can assume:

$$\boldsymbol{\gamma} \sim \text{Dir}(\boldsymbol{\gamma};\, \alpha \mathbf{1}_K) = \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \gamma_k^{\alpha-1} \quad , \tag{37}$$

with constraint that $\mathbf{1}\left(\sum_k \gamma_k = 1\right)$ needed for the Multinomial distribution. The overall posterior is then:

$$\begin{aligned}
P(\mathbf{Z}, \mathbf{C}, \boldsymbol{\gamma}|\mathbf{A}, a, b, \alpha) &\propto P(\mathbf{Z}, \mathbf{C}, \boldsymbol{\gamma}, \mathbf{A}|a, b, \alpha) && (38)\\
&= P(\mathbf{A}|\mathbf{Z}, \mathbf{C}, \boldsymbol{\gamma})\, P(\mathbf{Z}, \mathbf{C}, \boldsymbol{\gamma}|a, b, \alpha) && (39)\\
&= P(\mathbf{A}|\mathbf{Z}, \mathbf{C}, \boldsymbol{\gamma})\, P(\mathbf{Z}|\mathbf{C}, \boldsymbol{\gamma}, a, b, \alpha) P(\mathbf{C}|\boldsymbol{\gamma}, a, b, \alpha)\, P(\boldsymbol{\gamma}|a, b, \alpha) && (40)\\
&= P(\mathbf{A}|\mathbf{Z}, \mathbf{C})\, P(\mathbf{Z}|\boldsymbol{\gamma})\, P(\mathbf{C}|a, b)\, P(\boldsymbol{\gamma}|\alpha) && (41)\\
&= \prod_{i<j} \text{Bern}(A_{ij}; \mathbf{z}_i^T \mathbf{C} \mathbf{z}_j^T) \prod_i \text{Mult}(\mathbf{z}_i; \boldsymbol{\gamma}) \\
&\quad \times \text{Dir}(\boldsymbol{\gamma};\, \alpha \mathbf{1}_K) \prod_{k \le q} \text{Beta}(C_{kq}; a_{kq}, b_{kq}) && (42)
\end{aligned}$$

The parameters $\mathbf{Z}, \mathbf{C}, \boldsymbol{\gamma}$ can be inferred for instance using Gibbs sampling as done in Nowicki and Snijders (2001). This is an iterative simulation scheme that operates by repeatedly drawing in turn each of a set of unknown random variables or vectors, each conditionally on the values of all of the other random variables. Going back to consider the variable $\mathbf{q}$, (recall that $P(q_i = k) = P(z_{ik} = 1) = \gamma_k$) this can be done in 2 steps:

1. Draw $\mathbf{C}, \boldsymbol{\gamma}$ from their posterior distribution *given* $\mathbf{A}, \mathbf{q}$.
2. Draw q_i from its posterior *given* $\mathbf{A}, \mathbf{C}, \boldsymbol{\gamma}, \{q_j\}_{j \neq i}$, $\forall i = 1, \dots, N$.

The posterior of q_i *given* $\mathbf{A}, \mathbf{C}, \boldsymbol{\gamma}$ can be found by considering the *complete conditional* of the full posterior (42). This is the distribution obtained by isolating all the terms containing only a given q_i (we omit here all the hyperpriors' dependence):

$$P(q_i = k | \mathbf{A}, \mathbf{C}, \boldsymbol{\gamma}, \{q_j\}_{j \neq i}) \propto \gamma_k \prod_{i<j} C_{kq_j}^{A_{ij}} (1 - C_{kq_j})^{1 - A_{ij}} \tag{43}$$

$$= \gamma_k \prod_{q} C_{kq}^{d_{iq}} (1 - C_{kq})^{\hat{d}_{iq}} \quad , \tag{44}$$

where d_{iq} is the number of neighbors of i that belong to group q, and $\hat{d}_{iq}$ is the number of non-neighbors of i that belong to group q.
Assuming Dirichlet priors for both $\mathbf{C}$ and $\boldsymbol{\gamma}$, we get that also the posterior of $\mathbf{C}, \boldsymbol{\gamma}$ is made of independent Dirchelet, given $\mathbf{A}, \mathbf{q}$.
Obs1: in practice, the equilibrium time of Markov chains is high for large networks. Thus, the running time of Gibbs sampling is one reason why the Bayesian approach is not that widely used for community detection.

1.8 SBM: summary

- SBM is a classical inference problem that allows flexibility to model various datasets and systems.
- It's a nice testbed to try different inference techniques as discussed here.
- It can be further generalized to include more insights specific to the application considered, like dynamicity, overlapping communities, multilayer networks etc...
- It raises several interesting and deep questions like what to do with the final partitions, should we sample or optimize the division in group? How do we select for number of groups?
- As for other similar problems, it is affected by the detectability problem, i.e. there is a rang of parameters within which there is no chance to recover the ground truth partition.

Two recent references for this lecture are Funke and Becker (2019); Peixoto (2017).

References

M. E. Newman and G. Reinert, Physical review letters **117**, 078301 (2016).

T. P. Peixoto, arXiv preprint arXiv:1705.10225 (2017).

B. Karrer and M. E. Newman, Physical review E **83**, 016107 (2011).

A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Physical Review Letters **107**, 065701 (2011).

K. Nowicki and T. A. B. Snijders, Journal of the American statistical association **96**, 1077 (2001).

T. Funke and T. Becker, PloS one **14**, e0215296 (2019).