L2: Stochastic block model and mixed-membership

Caterina De Bacco

May 7, 2025

1. Introduction

In the previous lecture we learned about the standard stochastic block model and its main assumptions. Here we investigate various topics related to this problem:

a. **Mixed-membership** version, where nodes are allowed to belong to more than one group.
b. **Model selection**, to evaluation what model is the best.
c. Adding **extra information**, as node attributes. This is important in presence of additional data, beyond the network adjacency matrix.

2. Poisson matrix factorization: the problem

We have seen the stochastic block model (SBM) as a foundational method to cluster nodes in a network into communities based on their pairwise interactions.

Beyond enforcing stochastic equivalence and conditional independence between edges, the SBM assumes that nodes can belong to only one group. In many scenarios this assumption is too restrictive. For instance, in social networks, it is fair to assume that people can belong to more than one group, and with different intensities. In other words, we want a model for mixed-membership, or overlapping communities.

Objective: cluster people in multiple groups based on their pattern of interaction.

Imposing mixed-membership can be obtained by defining the membership $u_i \in \mathbb{R}^K_{>0}$ of node i as K-dimensional vector of positive entries where there can be more than one non-zero entry (as opposed to the SBM where only one entry is equal to 1 and all the others 0, as in one-hot encoding).
Furthermore, we may assume that the more two nodes belong to the same multiple groups, the higher their probability of interaction. Mathematically, this can be obtained by considering the dot products of the membership vectors, so that now stochastic equivalence is generalized as:

$$Pr(A_{ij} = a) = f(u_i \cdot u_j) \quad , \tag{1}$$

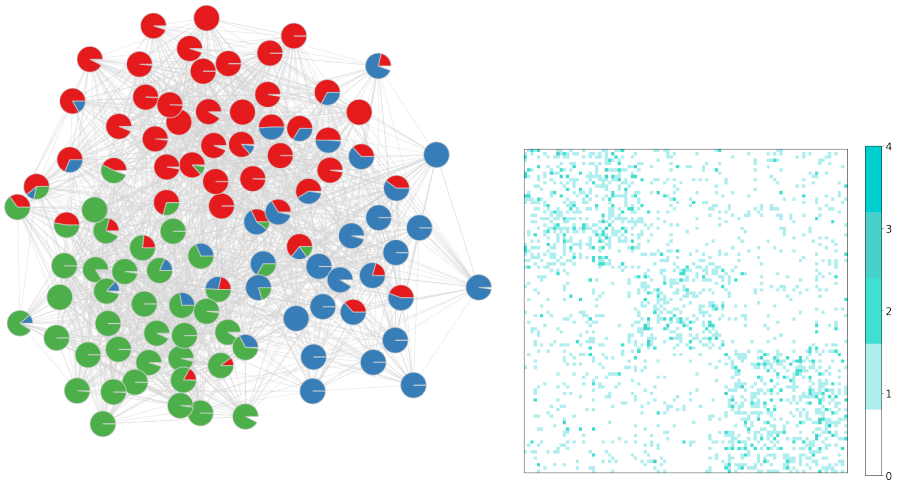

FIGURE 1. Example of a mixed-membership graph and its adjacency matrix.

where a can be any scalar value. Here we focus on it being positive and discrete, to represent count data.
In order to allow directed networks, i.e. the probability of an edge (i, j) may be different than that of the opposite edge (j, i), we should consider the existence of two types of membership, out-going u_i and in-coming v_j. We can still keep a $K \times K$ affinity matrix C, that allows to tune different topology structures, e.g. assortative, disassortative etc... Putting all together, we assume the likelihood:

$$P(A; u, v, C) = \prod_{ij} \frac{\left(\sum_{k,q} u_{ik} v_{jq} c_{kq}\right)^{A_{ij}}}{A_{ij}!} e^{-\sum_{k,q} u_{ik} v_{jq} c_{kq}} \quad . \tag{2}$$

This is an example of a matrix factorization, in particular a Poisson matrix factorization (PMF). In fact, denoting with U and V the $N \times K$ membership matrices of rows u_i and v_i respectively, we obtain that the expected value of the adjacency matrix is:

$$\mathbb{E}[A] = UCV^T \quad , \tag{3}$$

which can be related to a linear algebra problem of finding a low-rank approximation of a matrix $A \approx UV^T$.

2.1. Maximum likelihood and EM

We start by showing the approach using MLE estimation. We fit the model to an observed network by maximizing this probability with respect to the parameters, or equivalently (and more conveniently) maximizing its logarithm. Defining $\lambda_{ij} = \sum_{k,q} u_{ik} v_{jq} c_{kq}$, yields:

$$\mathcal{L}(u, v, C) = \log P(A; u, v, C) = \sum_{ij} A_{ij} \log \lambda_{ij} - \sum_{ij} \lambda_{ij} \quad , \tag{4}$$

where we omitted constants not depending on the parameters.

REMARK 1. *Notice that this time we can take derivatives w.r.t. the parameters, as these are now real-valued quantities.*

REMARK 2. *In addition, it is not evident how to set up a Monte Carlo scheme for mixed-membership cases, as for hard-communities one was switching the colors of nodes at each step.*

If we directly derive this expression by one of the parameters, e.g. u_{ik}, would lead to ugly terms like the logarithms of a sum. Instead, we use **Jensen's inequality**:

$$\log\left(\sum_k x_k\right) \geq \sum_k q_k \log \frac{x_k}{q_k} \quad , \tag{5}$$

where q_k's are any probabilities satisfying $\sum_k q_k = 1$ and x_k's are any set of positive numbers. The exact equality can always be achieved imposing:

$$q_k = \frac{x_k}{\sum_k x_k} \quad . \tag{6}$$

Applying this to $\mathcal{L}(u, v, C)$ we get:

$$\mathcal{L}(u, v, C) \geq \sum_{ijkq} A_{ij} q_{ijkq} \log \frac{u_{ik} v_{jq} c_{kq}}{q_{ijkq}} - \sum_{ijkq} u_{ik} v_{jq} c_{kq} := L(u, v, C, q) \quad . \tag{7}$$

The exact equality is obtained when:

$$q_{ijkq} = \frac{u_{ik} v_{jq} c_{kq}}{\sum_{kq} u_{ik} v_{jq} c_{kq}} , \tag{8}$$

hence the double maximization of $L(u, v, C, q)$ w.r.t. q and $\theta = (u, v, C)$ is equivalent to maximize $\mathcal{L}(u, v, C)$ w.r.t. θ. Therefore we now proceed in maximizing the more convenient L.

REMARK 3. *Maximizing L is more convenient because it contains only products inside the logarithms, not sums.*

REMARK 4. *The distribution q plays a role of a variational distribution and nicely integrates into a EM-algorithms. In particular, this is the distribution updated in the E-step.*

REMARK 5. *There are in principle many entries to update for determining q, as this is a $N \times N \times K \times K$, so a $N^2 \times K^2$ complexity. However, the q_{ijkq} terms only enter L through a multiplication by the A_{ij}. This means that we only need to calculate the q_{ijkq} such that $A_{ij} > 0$. In real networks, which are often sparse, this means only accounting for non-zero entries, which are equal to the number of edges $M \sim N$. This means that the complexity is linear in N, not N^2!*

REMARK 6. *The value of q_{ijkq} has a simple physical interpretation: it is proportional to the probability that an edge between i and j exists because of i having color k and j color q.*

Now we can proceed differentiating w.r.t. θ. For instance, taking the derivative w.r.t. u_{ik}.

$$\frac{\partial L}{\partial u_{ik}} = -\sum_{jq} v_{jq} c_{kq} + \frac{1}{u_{ik}} \sum_{jq} A_{ij} q_{ijkq} \quad , \tag{9}$$

setting this to zero leads to:

$$u_{ik} = \frac{\sum_{jq} A_{ij} q_{ijkq}}{\sum_{jq} v_{jq} c_{kq}} \quad . \tag{10}$$

Similarly for the other parameters:

$$v_{jq} = \frac{\sum_{ik} A_{ij} q_{ijkq}}{\sum_{ik} u_{ik} c_{kq}} \tag{11}$$

$$c_{kq} = \frac{\sum_{ij} A_{ij} q_{ijkq}}{\sum_{ij} u_{ik} v_{jq}} \quad . \tag{12}$$

Maximizing the log likelihood is now simply a matter of simultaneously solving Eq. (8) and Eqs. (10) - (12), which can be done iteratively by choosing a random set of initial values and alternating back and forth between the two equations. This is the EM algorithm.

References for this method are Ball *et al.* (2011) and De Bacco *et al.* (2017); Contisciani *et al.* (2020) for further generalizations for multilayer networks and covariates.

2.2. Constraints and regularization

We did not impose any constraints on the parameters. However, we implicitly imposed that u, v, C must be positive (we cannot have negative parameters for the Poisson). In fact, because the updates are all multiplicative, i.e. of the type:

$$u_{ik}^{(t+1)} = \frac{\sum_{jq} A_{ij} q_{ijkq}^{(t)}}{\sum_{jq} v_{jq}^{(t)} c_{kq}^{(t)}} = u_{ik}^{(t)} \times \frac{\sum_{jq} A_{ij} v_{jq}^{(t)} c_{kq}^{(t)}}{\sum_{jq} v_{jq}^{(t)} c_{kq}^{(t)}} \quad , \tag{13}$$

where we made explicit the iteration step with the superscripts and unpacked q_{ijkq}. Hence, if we initialize $u_{ik}^{(0)} \geq 0$ and similarly for the other parameters, we will always obtain positive parameters.

We can also add further constraints, for instance the normalization $\sum_k u_{ik} = 1$. This could be enforced using Lagrange multipliers. It would lead to similar multiplicative updates but with different denominators.

Alternatively, one can take a Bayesian approach and impose a prior on the parameters to impose a regularization. For instance, if we want the entries of u and v to be small so that nodes do not belong to many communities, i.e. sparsity constraints, then we can assume an exponential prior:

$$P(u_{ik}|a_{ik}) = a_{ik} e^{-a_{ik} u_{ik}} \tag{14}$$

$$P(v_{ik}|b_{ik}) = b_{ik} e^{-b_{ik} v_{ik}} \quad , \tag{15}$$

where a, b are hyper-priors given in input. One should then use MAP estimate instead of MLE and maximize the log-posterior:

$$L(u, v, C, q) - \sum_{i,k} a_{ik} u_{ik} - \sum_{i,k} b_{ik} v_{ik} \quad . \tag{16}$$

Notice that if we assume uniform $a_{ik} = a$ and $b_{ik} = b$, then we get a L1-regularization:

$$L(u, v, C, q) - a \sum_{i,k} u_{ik} - b \sum_{i,k} v_{ik} \quad . \tag{17}$$

A similar result can be obtained with a gamma prior.
Exercise: prove this.

2.3. PMF algorithmic updates: EM

The algorithmic updates to optimize the log-likelihood alternate between an E-step updating the variational distributions q_{ijkq} and the M-step updating the parameters u, v, C are shown in Algorithm 1.

Algorithm 1: Expectation-Maximization for Poisson Matrix Factorization

Input: Data A.

Initialize u, v, C randomly.

while *change in L is above some threshold* **do**

E step.

For each pair of nodes such that $A_{ij} > 0$, update the variational distributions:

$$q_{ijkq} = \frac{u_{ik} v_{jq} c_{kq}}{\sum_{kq} u_{ik} v_{jq} c_{kq}}$$

M step.

For each node, update the out-going membership parameters:

$$u_{ik} = \frac{\sum_{jq} A_{ij} q_{ijkq}}{\sum_{jq} v_{jq} c_{kq}}$$

update the in-coming membership parameters:

$$v_{jq} = \frac{\sum_{ik} A_{ij} q_{ijkq}}{\sum_{ik} u_{ik} c_{kq}}$$

For each pair k, q, update the affinity matrix parameters:

$$c_{kq} = \frac{\sum_{ij} A_{ij} q_{ijkq}}{\sum_{ij} u_{ik} v_{jq}}$$

end

Output: point-estimates of the parameters (u, v, C).

3. Model selection

Often, we have more than one model variant at our disposal. For instance, we can impose constraints on the affinity matrix C to allow only diagonal entries to be non-zero (assortative) or keep them free (disassortative). Similarly, we can vary the number of groups K and obtain different results.

Question: How do we evaluate which model is the best?

We need to use some model selection criteria.
There are various possibility for this, here we focus the particular choice of predictive performance. In fact, one possibility is to choose the model that better predicts missing information. In network, as the basic unit of information are links, we can setup a link prediction task and then measure predictive performance.

3.1. Cross-validation and link prediction

A standard procedure is cross-validation, where one splits the entries of the adjacency matrix into train and test. Then, the training set is used to infer the parameters, and the test is used to measure predictive performance.

a. Split A into train and test. E.g. build a binary mask of the same dimension as A. $A_{test} = A[mask]$, $A_{train} = A[\neg mask]$ (here $\neg$ is the symbol for logical NOT).
b. Train a model by giving in input as data A_{train}. The output is the set of parameters θ.
c. Calculate a score per edge using the learned parameters. In probabilistic models, this is usually $P(A_{ij}|\theta)$, for any (i, j) belonging to train or test.
d. Compute the desired performance metric only using A_{test}. For instance, heldout log-likelihood is $\sum_{i,j \in mask>0} \log P(A_{ij}|\theta)$, i.e. considering only the pairs (i, j) in the test set.
e. Repeat for all the possible masks. In cross-validations, you build N_{fold} masks that do not intersect.

The best model is the one with highest average performance metric over the folds. In fig. 2 we show example plots that can be used to evaluate what model performs the best.

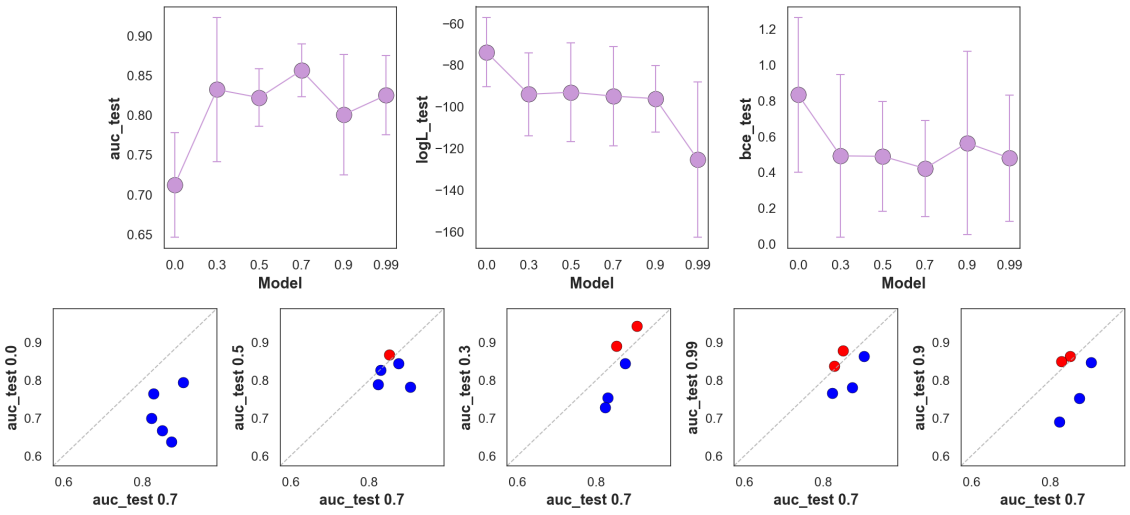

FIGURE 2. Example plots to evaluate predictive performance as model selection criterium. Top) average and standard deviations of three performance metrics calculated over 5 test set folds, against various models (x-axis). Bottom) fold-by-fold comparison of the AUC between one method (y-axis) and a reference method (x-axis). Blue markers are for points where the reference method outperforms the comparison one.

There are other model selection approaches beyond measuring predictive performance. Here we list some of them:

- Compression as in information theory, e.g. minimum description length (MDL), if you have a Bayesian method.
- Posterior predictive checks Gelman *et al.* (1996, 2013), if you have a Bayesian method.
- Sample from the learned model new data and measure topological properties on the samples. Checks if these are similar as in the original dataset. This is valid if you have a generative model, where you can sample from a learned distribution.
- Information criteria as BIC or AIC.

See Vallès-Català *et al.* (2018); Peixoto (2019) for discussions about compression vs prediction.

4. Adding extra information

In some cases, a network dataset comes with additional information beyond the input set of edges. For instance, one can have node attributes or different types of edges.

Question: How do we use this extra information?

The idea is that when extra information is correlated with community structure, it should help finding a good partition. This is particularly useful in case of sparse data, where there is little information to start with. There are various ways to add this information into a probabilistic model like the ones explored so far, but the starting information is that now we have this extra data X, in addition to A. We can treat also X as a random variable and ask what is the probability $P(X|\Theta)$ of observing it given the parameters, similarly as done for A. Now, to be able to learn the Θ that best measure $P(A|\Theta)$ while using X, the key is to have some of the parameters in Θ shared between the two likelihoods. This can be done by setting up a joint probability:

$$P(A, X|\Theta) \quad , \tag{18}$$

and learning Θ using statistical inference techniques like one does when fitting only the data A. However, setting up a joint likelihood for two possibly different types of datasets (different dimensions, data types, etc...) can be challenging and it is not clear a priori. Instead, it may be easier to adopt one of two more intuitive approaches.

4.1. Factorized likelihoods

One approach is to factorize the joint likelihood into two separate parts, one per dataset. Then, one needs at least one shared parameters to be able to correlate the two types of information:

$$P(A, X|\Theta) = P(A|\theta_A, \theta)\, P(X|\theta_X, \theta) \quad , \tag{19}$$

where θ_A, θ_X are parameters specific to each dataset, while θ is a shared set of parameters, the most important one to be able to learn from one another. The key assumption is *conditional independence* between the two types of data, given the set of parameters. An

example of this, that uses an underlying Poisson factorization as in section 2, is MTCOV Contisciani *et al.* (2020). In that approach, one models:

$$P(A|\theta_A, \theta) = \text{Pois}\left(A_{ij}; \sum_{k,q} u_{ik} v_{jq} w_{kw}\right) \quad , \tag{20}$$

similarly to what done in eq. (2).
Instead, the matrix X of dimension $N \times Z$ representing categorical node attributes, where $x_{iz} = 1$ if node i has attribute of category z, 0 otherwise, is modeled with a multinomial distribution:

$$P(X_i = x_i|\theta) = P(X_{i1} = x_{i1}, \dots, X_{iZ} = x_{iZ}|\theta) = \pi_{i1}^{x_{i1}} \dots \pi_{iZ}^{x_{iZ}} \quad , \tag{21}$$

where the parameters:

$$\pi_{iz} = \frac{1}{2} \sum_{k=1^K} \beta_{kz} \left(u_{ik} + v_{ik}\right) \quad , \tag{22}$$

represent the probability that node i has attribute of category z. Here β is a $K \times Z$ matrix measuring the probability of observing a particular category z together with a community k.
Overall, we have $\theta_A = \{W\}$, $\theta_X = \{\beta\}$ and $\theta = \{U, V\}$. Hence, we see that the membership vectors are the shared parameters. They appear in both likelihood terms and are therefore learned using both types of information. In addition, now the parameters U, V need to be normalized to 1 as $\sum_k u_{ik} = 1$, to make sure that they are valid parameters for the multinomial.
In practice, inference is performed via maximum likelihood and for this one considers a sum of two log-likelihood terms, weighted by an hyperparameter $\gamma \in [0, 1]$:

$$\mathcal{L}(U, V, W, \beta) = (1 - \gamma)\, \mathcal{L}_A(U, V, W) + \gamma\, \mathcal{L}_X(U, V, \beta) \quad , \tag{23}$$

where $\mathcal{L}_A = \log P(A)$ and $\mathcal{L}_X = \log P(X)$. This parameter is necessary to weigh the two dataset differently, as they can contain different amount of data, both in terms of quantity and quality. This should be learned by fitting to data, e.g. using cross-validation, analogously to what done to learn K.
In Fig. 3 we can see an example result of what can be obtained by using a node attribute in the WTO trade network.
Another example of this type of factorization in presence of different types of data can be found when combining an hyperlink network of documents to the text that generates them Zhu *et al.* (2013).

4.2. Attributes as prior

A different approach is to factorize the joint probability of the two datasets in eq. (18) as a product of prior and likelihood, where the auxiliary dataset (e.g. node attributes) enters in the prior:

$$P(A, X|\theta) = P(A|X, \theta)\, P(X|\theta) \quad . \tag{24}$$

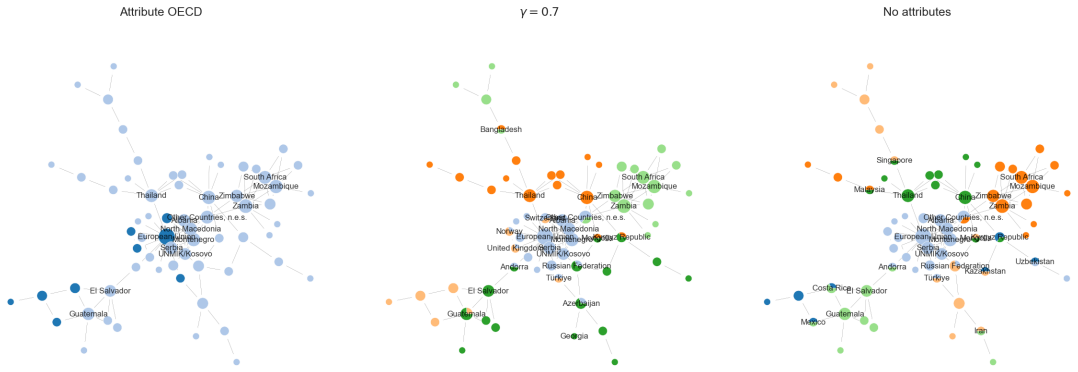

FIGURE 3. Community structure using node attributes in input on the WTO dataset. Left) Node attributes, binary, representing countries belonging to OECD (Organisation for Economic Co-operation and Development). Center) community structure found with MTCOV with $K = 6$ and the best $\gamma = 0.7$ found with 5-fold cross-validation ($AUC = 0.75 \pm 0.07$, over a grid search of values $\gamma \in \{0.0, 0.3, 0.5, 0.7, 0.9, 0.99\}$). Right) community structure found with MTCOV with $K = 6$ and no attributes ($\gamma = 0.0$, $AUC = 0.71 \pm 0.07$).

En example of this is the model in Newman and Clauset (2016), where they model the node (hard)-membership u_i as generated by the node attributes as the prior:

$$P(u|\Gamma, X) = \prod_i \gamma_{u_i x_i} \quad , \tag{25}$$

where the entries $\gamma_{u_i x_i}$ of a $K \times Z$ matrix Γ represent the probability that a node in community u_i has attribute x_i. Then, this enters as a prior to be multiplied against the likelihood, which depends explicitly only on u:

$$P(A, X|\theta) = \sum_u P(A|C, u)\, P(u|\Gamma, X) \quad , \tag{26}$$

where the likelihood is the SBM Bernoulli likelihood and C is the $K \times K$ affinity matrix (with degree correction).
In this case, the parameter Γ may also be of interest, similarly to the parameter γ in eq. (23). The Γ in the prior tells us how and to what extent the metadata are correlated with the communities. If the metadata are uncorrelated with the network communities, the prior probabilities become constant, independent of the metadata. They have no impact on the posterior probabilities of the communities. Similarly, if the network is large and has strong community structure, the prior probabilities will have little effect on the results and the algorithm will find the communities without help from the metadata. Both these case will be equivalent to infer a $\gamma \approx 0$ in MTCOV.

4.3. SBM part B: summary

- SBM can be extended to mixed-membership, allowing for EM algorithmic updates.
- Choosing the best number of groups K or between models can be done using model selection criteria. Cross-validation is one common approach.

- Extra information, e.g. node attributes, can be added into the formulation in different ways. This information will be used provided it is correlated with community structure, otherwise it should be discarded and algorithms should be able to learn this from data.

Further reference for this lecture are Peel *et al.* (2017) for discussions about the role of metadata in community detection.

References

B. Ball, B. Karrer, and M. E. Newman, Physical Review E **84**, 036103 (2011).

C. De Bacco, E. A. Power, D. B. Larremore, and C. Moore, Physical Review E **95**, 042317 (2017).

M. Contisciani, E. A. Power, and C. De Bacco, Scientific reports **10**, 1 (2020).

A. Gelman, X.-L. Meng, and H. Stern, Statistica sinica , 733 (1996).

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis* (CreateSpace, United States, 2013).

T. Vallès-Català, T. P. Peixoto, M. Sales-Pardo, and R. Guimerà, Physical Review E **97**, 062316 (2018).

T. P. Peixoto, Advances in network clustering and blockmodeling , 289 (2019).

Y. Zhu, X. Yan, L. Getoor, and C. Moore, in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013) pp. 473–481.

M. E. Newman and A. Clauset, Nature communications **7**, 11863 (2016).

L. Peel, D. B. Larremore, and A. Clauset, Science advances **3**, e1602548 (2017).