Effects of Null Model Choice on Modularity Maximization

Christopher Brissette[1,2], Ujwal Pandey[1,3], and George M. Slota[1,4]

[1] Rensselaer Polytechnic Institute, Troy NY
[2] brissc@rpi.edu, [3] pandeu@rpi.edu, [4] slotag@rpi.edu

Abstract. Given a defined set of communities, modularity is computed by comparing each existing edge with its probability of occurrence in a random graph null model. The heuristic has historically garnered a wealth of attention, and many community detection algorithms have been designed around maximizing modularity. Despite this, there are potential issues with the Chung-Lu null graph model that underpins the heuristic. In this manuscript, we explore the output communities given by modularity maximization when this null model is subject to change. We construct two null models using iterated double edge swapping and maximum likelihood estimation, and we use these models as the basis for new modularity-like heuristics we call *desmod*, and *mlemod*. We compare the clusters output by standard modularity maximization with those output by our methods on a test suite of LFR benchmark graphs and find that changing the null model consistently increases the normalized mutual information scores when the mixing parameter is high.

Keywords: clustering, modularity, random graphs

1 Introduction

Community detection, also synonymously referred to as 'graph clustering', is one of the most well-studied problems in the field of network science [13, 21, 19, 7, 15]. Generally, the goal of community detection algorithms is to partition a graph into disjoint or overlapping [31] vertex partitions. While the exact optimization metric for this partitioning varies, one generally attempts to group vertices with similar attributes into the same community membership. In the absence of metadata, labels, or other non-topological attributes, basic network topology is used as the defining metric for 'similarity'. In this case, the measure of network modularity is a natural optimization metric.

Modularity can be loosely defined as to how *well* a given network topology is divided into a given set of communities relative to a random network [26]. It is an important heuristic for community detection, and it has played a significant role in the literature [15, 10, 18, 6]. For a given set of communities, modularity is defined as the difference between the number of observed and number of expected intra-community edges in a network. If the modularity is high, the former is larger than the latter, and the given communities are considered 'more

clustered than expected'. For a given graph $G = (V, E)$, and defined labels for each node $\{c_i\}_{i \in V}$, modularity is explicitly computed as in Equation 1.

$$Q = \frac{1}{2m} \sum_{u,v \in V} \left[A_{uv} - \frac{d_u d_v}{2m} \right] \delta(c_u, c_v) \tag{1}$$

Here A is the adjacency matrix for the graph G, m is the number of edges in the network, d_u, d_v are the degrees of a nodes $u, v \in V$, and $\delta(\cdot, \cdot)$ is the Kronecker delta function. Despite how notable modularity maximization is in literature, it has some known problems such as the 'resolution limit' [14, 22] in which the method cannot discern communities under a certain size for a given network density. A far less discussed issue with the modularity heuristic is its dependence on the Chung-Lu graph model [8, 30], which assumes a random wiring of edges for a given degree distribution in expectation with self-loops and multi-edges. In essence, computing modularity is simply evaluating the null hypothesis on a given community assignment and network topology with one on a random null model graph provided by Chung-Lu. The contribution of the model comes from the term $\frac{d_u d_v}{2m}$ in Equation 1. There are a significant number of ways to define a null model within a discussion on network topology. An excellent recent review article by Fosdick et al. [16] discusses the many considerations. Changing the implicit null model in modularity maximization has also been shown to produce substantial difference in the obtained communities in the case of geometrically constrained graphs [20].

In our prior work [17, 3, 2] and other related work [27, 12, 4, 11], the general issues with the usage of Chung-Lu probabilities were studied for random graph generation. These problems include the theoretical reality that Chung-Lu generation cannot actually produce a vast majority of possible degree distributions [3], and the considerable error that results when using Chung-Lu probabilities for simple graph generation [4]. Similar issues arise when using Chung-Lu probabilities as an implicit null model in methods such as modularity maximization[1]. Fosdick et al. [16] was the first work we know of in the literature that gave such considerations more than a cursory glance.

Hence, the primary focus of this work is to experimentally examine the impact of using proper null graph probabilities in place of the Chung-Lu model for the specific problem of community detection via modularity maximization. We additionally utilize methods from our prior work [3, 2], which were in part motivated by the Fosdick et al. review, to derive vertex pairwise attachment probabilities. These attachment probabilities are then used in place of Chung-Lu probabilities within the computation of modularity for a maximization algorithm.

Modularity maximization [26, 10, 18, 6] methods generally take two forms – single or multi-level methods. Single level methods make choices on vertex–community membership for each individual vertex, while multi-level methods

[1] One of the major issues is that a majority of graphs studied in community detection fall squarely in the simple graph space. We note this applies to LFR and similar benchmark graphs and a large proportion of the real-world graphs with defined communities; e.g., those listed in the SNAP repository.

assign vertices to communities and then iteratively coarsen communities into single vertices to improve computation time. We will specifically consider the former category of algorithms in this paper, although we will also discuss how our methods can be extended to multi-level algorithms such as the popular Louvain algorithm [1].

Generally, the outputs of community detection algorithms on networks are evaluated in two primary ways [21, 7]. If some *ground truth* community definitions exists for the network, the output of the algorithm is compared to this ground truth via computing a metric such as normalized mutual information (NMI). Absent an explicit ground truth, evaluation can be done by comparing some computed metric derived solely from the community assignments and network structure. In the latter case, modularity is one such popular measure. However, in the context of this paper, we are unable to directly use modularity scores, since our experimental differences are in *the way we compute modularity*. Hence, our experimental evaluations focus on comparisons to network ground truth with the NMI metric. To generate a suitably large number and variety on test instances, we utilize the common Lancichinetti-Fortunato-Radicchi (LFR) benchmark generator [23]. We will discuss our experimental setup in more detail later in this manuscript.

In summary, our contributions are as follows:

1. We are the first work to extensively study the usage of a more appropriate null graph model within the context of modularity maximization.
2. We detail our approach to computing attachment probabilities and their effective utilization within a modularity maximization framework.
3. We observe that this change in attachment probabilities can improve computed NMI scores by up to fifty percent on average for some data sets.
4. We discuss how this work might be applied in future efforts, such as with multi-level community detection algorithms.

2 Methods

Ultimately, including custom probabilities into the computation of modularity is trivial once one determines these probabilities. All one needs to do is replace the Chung-Lu term '$\frac{d_u d_v}{2m}$' with a general term for the connection probability 'p_{uv}' between nodes u and v as in Equation 2. Algorithmically, however, there are some unique considerations which have to be made when using custom null models for modularity maximization.

$$Q = \frac{1}{2m} \sum_{u,v \in V} \left[A_{uv} - p_{uv}\right] \delta(c_u, c_v) \tag{2}$$

When the modularity maximization algorithm does not affect the underlying structure of the network, one can use a straightforward modularity maximization algorithm such as Clauset-Newman-Moore's algorithm [9]. As such, we implemented this algorithm to utilize any arbitrary p_{uv} and validated it against

NetworkX's implementation[2] with Chung-Lu probabilities. However, for multi-level methods such as the well-established Louvain maximization algorithm [1], null model discovery would need to be performed at every step. We will discuss future work intended to address this current drawback. Below, we will describe two methods for determining the p_{uv} attachment probabilities of the underlying null model of an arbitrary graph.

Algorithm 1 Computing double edge swap probabilities.

procedure DESSAMPLE($G = (V, E), k$)
  $P \leftarrow \frac{1}{k}$`get_adjacency`(G)
  $E' \leftarrow$ `permute`(E)
  for $i \in [0, \cdots, k-1]$ **do**
    for $j \in [0, \cdots, |E|-1]$ **do**
      if `valid_swap`(G, E_j, E'_j) **then**
        $G \leftarrow$ `perform_swap`(G, E_j, E'_j)
    $P \leftarrow P + \frac{1}{k}$`get_adjacency`$(G)$

Our first method, termed *desmod*, uses *double edge swaps* to randomly alter the graph topology while keeping the degree sequence the same. Double edge swaps take two edges and 'swap' the endpoints of each edge, permuting the edge-list of the graph while keeping the degree distribution consistent. The main idea behind the *desmod* method is to randomly sample a large number of realized instances of graphs with a fixed degree distribution to discern average degree–degree pairwise connection probabilities. We utilize a sampling method modified from our prior work [17, 28], which is based on established techniques [16, **?**]. This method, as outlined in the Algorithm 1, involves first permuting the edge-list and matching each edge in the permuted list with an edge in the original list. These will be our potential double edge swap partners. Of course, some of these swaps will not be *feasible* (i.e., results in a multi-edge or self loop), so we check each potential swap for viability using the primitive `valid_swap()`. For more details on this method and how valid swaps are chosen, we refer the reader to Fosdick et al. [16] and our prior work Garbus et al. [17].

Our second method, termed *mlemod*, is based on our recent prior work [2], which shows that degree distributions can be better approximated by Chung-Lu like methods with connection probabilities determined by maximum likelihood estimation. For detailed explanations of how the probabilities are computed, please see the referenced manuscript. The overarching idea behind the algorithm is that a degree sequence can be seen as a probability distribution, and the degrees of nodes with common weights will be distributed as Poisson distributions. To match this probability distribution, and hence the degree sequence we want, we can use maximum likelihood estimation to express this distribution as a sum of Poisson distributions from which nodal weights can be discerned for a Chung-Lu-like graph model. The *mlemod* method has some unique issues to be considered when compared with *desmod*. With *desmod*, each probability

[2] Implemented via the `greedy_modularity_communities(·)` function.

Algorithm 2 Computing maximum likelihood probabilities.

procedure MLEPROBS($G = (V, E)$)
  $D \leftarrow \texttt{degrees}(G)$
  $w \leftarrow \texttt{mle\_weights}(D)$
  $D \leftarrow \texttt{sort}(D)$
  $w \leftarrow \texttt{sort}(w)$
  $P \leftarrow \texttt{zeros}(|V|, |V|)$
  $m \leftarrow \frac{1}{2}\texttt{sum}(w)$
  for $u \in V$ **do**
    for $v \in V$ **do**
      $P_{uv} \leftarrow \frac{w_u w_v}{\sum_{x \in V} w_x}$

is assigned explicitly to an edge, meaning that the probabilities output can be directly applied to perform modularity maximization using Equation 2. In *mle-mod* we only obtain a list of weights, from which probabilities can be derived once they are assigned to explicit nodes. This means that *mlemod* requires us to assign each node the 'most probable' label.

For a given graph $G = (V, E)$, with $|V| = n$, and weights returned from a process such as MLE $w = \{w_1, \cdots, w_n\}$, the labeling problem amounts to finding a bijection $\phi : V \to w$ such that the probability of observing the given edge set $P(E|\phi(V))$ is maximized. If we call the sum of weights $\sum_{z \in V} \phi(z) = S$ we get Equation 5.

$$P(E|\phi(V)) = \left[\prod_{uv \in E} P(e_{uv}|\phi(u), \phi(v))\right] \left[\prod_{xy \notin E} (1 - P(e_{xy}|\phi(x), \phi(y)))\right] \tag{3}$$

$$= \left[\prod_{uv \in E} \frac{\phi(u)\phi(v)}{S}\right] \left[\prod_{xy \notin E} \left(1 - \frac{\phi(x)\phi(y)}{S}\right)\right] \tag{4}$$

$$\propto \left[\prod_{uv \in E} \phi(u)\phi(v)\right] \left[\prod_{xy \notin E} (S - \phi(x)\phi(y))\right] \tag{5}$$

Broadly speaking, the above equation states the following. The optimal way to match weights with nodes is to maximize the product of weights across existing edges, while minimizing the product of weights among non-existent edges. In general, this is a difficult optimization problem. However, we can heuristically come up with an easy approximate solution by only considering the leftmost term, which corresponds to maximizing the likelihood of existing edges.

$$OBJ = \max_{\phi} \left[\prod_{uv \in E} \phi(u)\phi(v)\right] = \max_{\phi} \left[\prod_{u \in V} \phi(u)^{d_u}\right] \tag{6}$$

In this case, we have the objective outlined in Equation 6. Fortunately, this is a very easy to maximize objective, as it only requires sorting the degrees of

our graph in descending order and matching them with the associated weights, also sorted in descending order. This process can be observed in Algorithm 2.

3 Results

Our experiments were run on the server *Bella* at RPI. *Bella* has 2× AMD Epyc 7742 processors with 64 cores at 2.25 GHz and 2 TB DDR4 at 2666 MHz, and it is running Ubuntu 20.04.6 with Python version 3.8.10 and NetworkX version 3.1.

We ran four sets of tests with various topological differences. For these experiments, all graphs were generated using the `LFR_benchmark_graph(·)` function available in `Python`'s `NetworkX` library. A range of parameters were input, and instances in which the generator failed regarding combinations of those parameters were skipped. The definitions for the various adjusted input parameters are given in Table 1. We compare community outputs using NMI values as well as the number of communities generated in comparison to the ground truth. We performed two primary experiments, which we will define below.

Table 1. Variable Definitions for Experimental LFR Generation.

Variable	Definition
N	Number of nodes in the network.
$\langle k \rangle$	Average degree of nodes in the network.
k_{max}	Maximum degree of nodes in the network.
k_{min}	Minimum degree of nodes in the network.
s_{min}	Minimum community size in the network.
τ_1	Power-law exponent of node degree distribution.
τ_2	Power-law exponent of community size distribution.
μ	Average proportion of edges that are external to communities.

3.1 General Experimental Set of Networks

We tested the results for a general set of graphs with a lower bound on community size. For these tests, we generated LFR's with the following parameters: $N \in \{4000, 8000, 16000\}$, $\tau_1 \in \{2, 2.5\}$, $\tau_2 \in \{1.1, 1.25, 1.5\}$, $\mu \in \{0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$, $s_{min} = 6$, $k_{min} = 5$, and $k_{max} \in \{\frac{N}{20}, \frac{6N}{100}, \frac{7N}{100}, \frac{8N}{100}, \frac{9N}{100}, \frac{N}{10}\}$. The NMI comparison between modularity maximization via the probabilities of Chung-Lu (chung), *desmod* (samp), and *mlemod* (mle) can be observed in Figure 1. We also show the number of communities obtained by each of these three methods plus the ground truths for this dataset in Figure 2. We show this due to possible 'NMI hacking' which can occur for randomly chosen communities as described in work by Vinh et. al. [29] where higher numbers of communities may sometimes yield higher NMI values without necessarily being better at matching the ground truth communities.

As Figure 1 demonstrates, our proposed null model choices both improve upon the NMI computed on outputs using baseline Chung-Lu probabilities, on

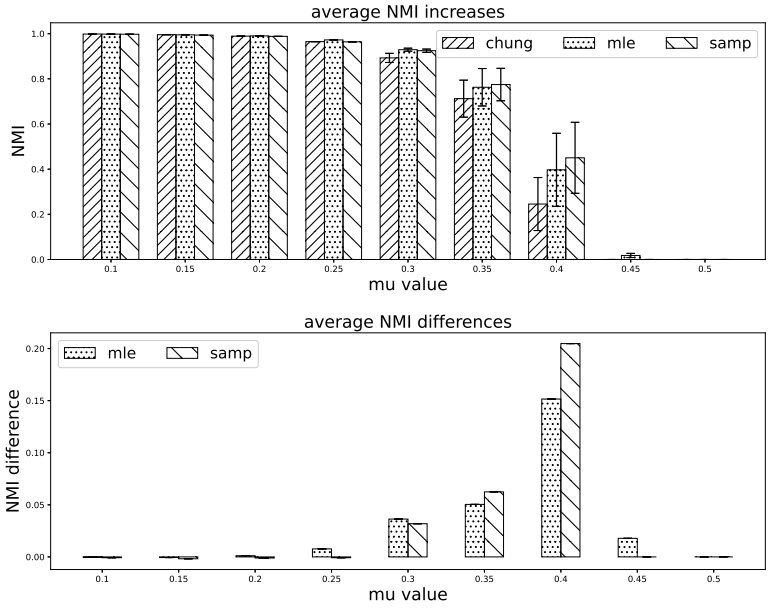

Fig. 1. NMI for general test set: NMI results for standard modularity maximization, *desmod*, and *mlemod* varying the μ parameter on LFR graphs with minimum degree 5, and minimum community size 6. We can see a sharp improvement in cluster quality near the $\mu = 0.4$ bound, implying that desmod and mlemod may perform better than standard modularity maximization in these test instances.

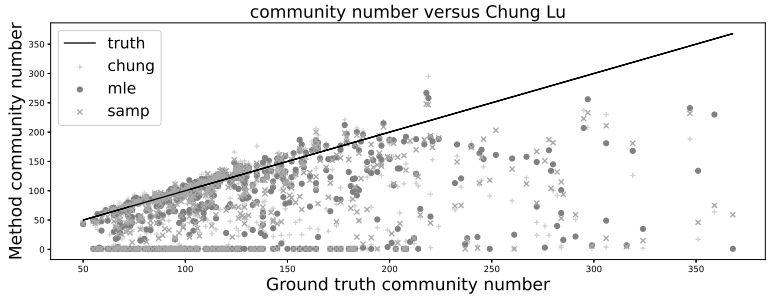

Fig. 2. Community numbers for general tests : The number of communities output by various methods in comparison to the ground truth communities. Each point is a specific graph, and the marker denotes the method used to obtain that graphs clustering. We can see that in most cases the number of communities is near, or below the expected number.

average. This difference is most notable at μ values approaching $\mu = 0.5$, the point at which communities become difficult to discern for all methods. This is expected, as $\mu = 0.5$ implies that the each node on average has as many edges external to their ground truth community as they have internal edges. We also

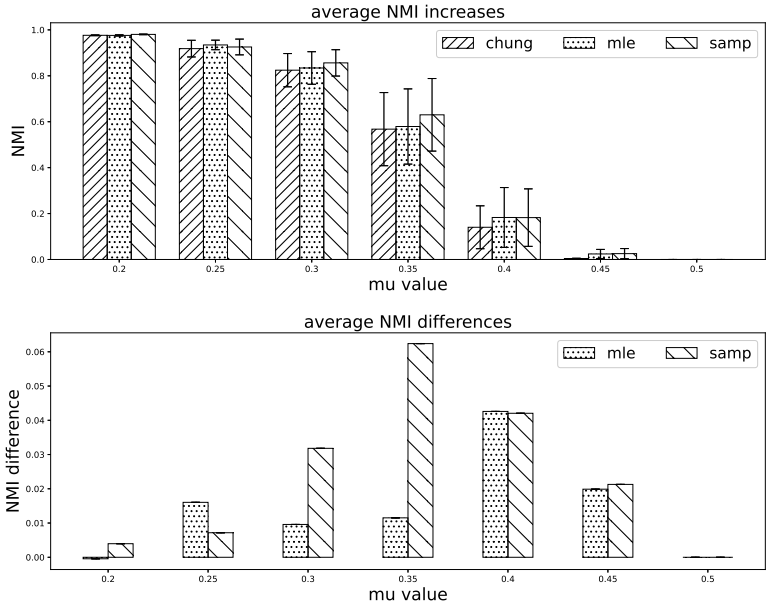

Fig. 3. NMI for τ_2=1.1 : NMI results for a narrow band of LFR graphs with $\tau_2 = 1.1$. Note that both *mlemod* and *desmod* have varying behavior in their NMI-difference plots in comparison to Figure 1. Regardless, we can still see that both *mlemod* and *desmod* outperform standard modularity maximization for every test instance, on average.

note that our outputs closely match the ground truth in terms of the number of communities, as shown in Figure 2. Generally, our proposed null models either closely match the ground truth, or it results in fewer communities output. This is possibly a consequence of the resolution limit, where multi-level methods, described later, would be able to improve upon these results in future work.

3.2 Fixed Community Size Distribution Experimental Set

Additionally, we explored the quality of clusters for LFR graphs with a fixed τ_2 parameter. This can be seen in Figure 3. We generated LFR's with the following parameters: $N \in \{1000, 2000, 4000, 8000\}$, $\mu \in \{0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$, $\tau_1 \in \{2, 2.5\}$, $\tau_2 \in \{1.1\}$[3], $\langle k \rangle \in \{20, 25, 30, 35\}$, and $k_{max} \in \{100, 150, 200, 250, 300\}$. The goal for these experiments was to observe how the methods perform as the distribution of clusters remains close to linear, which we noticed made community detection much more 'difficult' for our modularity maximization methods. This makes these graphs topologically unique among the generated LFR graphs, since their largest and smallest communities vary far less drastically.

We give the results of this experimental set in Figure 3, again as a comparison between output NMI values. As shown in Figure 3, we once more observe that

[3] We chose $\tau_2 = 1.1$, as NetworkX failed to generate graphs using $\tau_2 = 1.0$.

desmod (samp) and *mlemod* (mle) result in consistently higher NMI scores than standard Chung-Lu (chung) probabilities. However, in these tests, we find that the output NMI takes a different form to the general case. Here, the difference in NMI can be seen to peak at $\mu = 0.35$, and only for *desmod*. This implies that sampling may perform better under certain topological features. Meanwhile, *mlemod* under-performs in comparison to *desmod* for many μ values, implying that it too may suffer as a consequence of certain topological properties.

4 Discussion

The results in this manuscript indicate that the choice of null model has a significant impact on the observed cluster quality for many graphs. We find that the results for Chung-Lu, sampled (*desmod*), and MLE probabilities (*mlemod*) are consistently better for general LFR graphs with broadly-varying parameters and size. On average, across a wide range of tests, sampled and MLE probabilities achieve better NMI results than those of standard modularity maximization. In the particular cases of larger LFR graphs and LFR graphs with smaller τ_2 values, both methods consistently out-perform Chung-Lu based modularity maximization. The differences are particularly notable near the $\mu = 0.4$ boundary.

The observed results suggest that *mlemod* and *desmod* outperform standard modularity maximization in general, particularly when the connectivity between communities is relatively high. This implies these null models are more robust to network density than the Chung-Lu random graph model. The primary question the reader might have is: **Why?**

The explanation can actually be summarized quite succinctly. In our prior work [17], we have observed that Chung-Lu probabilities can over-estimate real attachment probabilities[4] between pairs of average degree nodes and pairs of high degree nodes within graphs with skewed degree distributions; low degree probabilities are otherwise similar. As a consequence, the baseline modularity maximization biases *against* assortativity, while most real networks and benchmark networks actually exhibit a considerable amount of assortative degree mixing within communities [24, 25, 5]. The use of appropriate null model probabilities 're-biases' towards assortative mixing within communities when performing modularity maximization.

We finally note one specific concern that may arise to the reader, in that our modularity maximization method used for experimentation is relatively naive compared to more modern modularity maximization algorithms. In the following subsection, we provide a theoretical justification for how one may extend a multi-level Louvain-type algorithm for use with our proposed null models.

4.1 Extension to Explicit multi-level Methods

One issue that arises when using bespoke null models defined by methods such as maximum likelihood estimation or sampling is that it excludes simple implementations of explicit multi-level schemes or approaches that otherwise modify the

[4] Here, 'real attachment probabilities' are those determined for an appropriate simple graph null model under an appropriate sampling methodology.

underlying network structure before maximizing modularity. E.g., in standard Louvain methods for modularity maximization, small communities are found on the original graph according to a greedy approach such as the one that we use in this paper. These communities are then coarsened into a single vertex in a weighted graph or multigraph, where modularity maximization is again performed. This allows for information from multiple scales to be considered, and it leads to generally better results. Louvain and many other more modern algorithms using this, and related approaches appear throughout the literature [7, 19].

Regardless, multi-level schemes may be implemented using custom graph-specific null models if we allow for extra computational overhead. Consider maximizing modularity the same way it is currently done in this paper. Then, a coarsened multi-graph may still be obtained, as in Louvain. Because we are not using Chung-Lu probabilities in this case, we do not have straightforward access to a null model for the coarsened graph. However, we can consider the probability of connection between the communities given by our initial clustering. In this case, the probability of connection between two clustered 'supernodes' can be thought of as the sum of connection probabilities between their comprising nodes. This can be seen in Equation 7, where C_i is a community of nodes given by the first round of modularity maximization.

We check that this is consistent with the behavior we expect from the Chung-Lu random graph model. This is shown in Equation 8, where we show that using Chung-Lu probabilities in accordance to the prior Equation 7 yields the associated Chung-Lu probabilities of the coarsened graph.

$$p_{C_iC_j} = \sum_{u \in C_i} \sum_{v \in C_j} p_{uv} \tag{7}$$

$$= \frac{1}{2m} \sum_{u \in C_i} \sum_{v \in C_j} d_u d_v = \frac{|C_j|}{2m} \sum_{u \in C_i} d_u \langle d_{C_j} \rangle = \frac{d_{C_i} d_{C_J}}{2m} \tag{8}$$

While this suggests the proposed method may be consistent with that of Chung-Lu, experimental validation should be performed to ensure that this provides meaningful communities. This is a primary topic of investigation for our ongoing and future work.

5 Conclusion

In this manuscript, we investigated the effects of null model choice on modularity maximization. We did this through a series of experiments using the LFR benchmark with varying parameters. We developed two different methods for altering modularity maximization for custom graph null models. One method, which we call *desmod*, uses double edge swaps on the input graph to generate probabilities. The other method, which we call *mlemod*, uses a maximum likelihood method along-side a greedy labeling to determine attachment probabilities. By replacing

standard modularity maximization with *desmod* and *mlemod*, we found that the normalized mutual information of output communities relative to the ground truth was better in many cases. In particular, both *mlemod* and *desmod* yield better results than Chung-Lu methods when inter-community connections are dense. We additionally suggest that this technique may be extended to the case of multi-level schemes such as Louvain, and we provide some theoretical justification for how that may be done. Despite this, such studies are left for future work and remain an open problem. This work represents a 'first foray' into the practical use of null model choice for community detection on general graphs, and it is the authors' hope that this spurs an interest in the topic for the broader community.

References

1. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment 2008(10), P10008 (2008)
2. Brissette, C., Liu, D., Slota, G.M.: Correcting output degree sequences in chung-lu random graph generation. In: International Conference on Complex Networks and Their Applications. pp. 69–80. Springer (2022)
3. Brissette, C., Slota, G.M.: Limitations of chung lu random graph generation. In: International Conference on Complex Networks and Their Applications. pp. 451–462. Springer (2021)
4. Britton, T., Deijfen, M., Martin-Löf, A.: Generating simple random graphs with prescribed degree distribution. Journal of Statistical Physics 124(6), 1377–1397 (2006)
5. Catanzaro, M., Caldarelli, G., Pietronero, L.: Assortative model for social networks. Physical review e 70(3), 037101 (2004)
6. Chen, M., Kuzmin, K., Szymanski, B.K.: Community detection via maximization of modularity and its variants. IEEE Transactions on Computational Social Systems 1(1), 46–65 (2014)
7. Chunaev, P.: Community detection in node-attributed social networks: a survey. Computer Science Review 37, 100286 (2020)
8. Chung, F., Lu, L.: The average distances in random graphs with given expected degrees. Proceedings of the National Academy of Sciences 99(25), 15879–15882 (2002)
9. Clauset, A., Newman, M.E., Moore, C.: Finding community structure in very large networks. Physical review E 70(6), 066111 (2004)
10. Despalatović, L., Vojković, T., Vukicević, D.: Community structure in networks: Girvan-newman algorithm improvement. In: 2014 37th international convention on information and communication technology, electronics and microelectronics (MIPRO). pp. 997–1002. IEEE (2014)
11. Drobyshevskiy, M., Turdakov, D.: Random graph modeling: A survey of the concepts. ACM Computing Surveys (CSUR) 52(6), 1–36 (2019)
12. Durak, N., Kolda, T.G., Pinar, A., Seshadhri, C.: A scalable null model for directed graphs matching all degree distributions: In, out, and reciprocal. In: 2013 IEEE 2nd Network Science Workshop (NSW). pp. 23–30. IEEE (2013)
13. Fortunato, S.: Community detection in graphs. Physics reports 486(3-5), 75–174 (2010)

14. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. Proceedings of the national academy of sciences 104(1), 36–41 (2007)
15. Fortunato, S., Hric, D.: Community detection in networks: A user guide. Physics reports 659, 1–44 (2016)
16. Fosdick, B.K., Larremore, D.B., Nishimura, J., Ugander, J.: Configuring random graph models with fixed degree sequences. SIAM Review 60(2), 315–355 (2018)
17. Garbus, J., Brissette, C., Slota, G.M.: Parallel generation of simple null graph models. In: The 5th IEEE Workshop on Parallel and Distributed Processing for Computational Social Systems (ParSocial) (2020)
18. Good, B.H., De Montjoye, Y.A., Clauset, A.: Performance of modularity maximization in practical contexts. Physical review E 81(4), 046106 (2010)
19. Jin, D., Yu, Z., Jiao, P., Pan, S., He, D., Wu, J., Yu, P., Zhang, W.: A survey of community detection approaches: From statistical modeling to deep learning. IEEE Transactions on Knowledge and Data Engineering (2021)
20. Kishore, R., Gogineni, A.K., Nussinov, Z., Sahu, K.K.: A nature inspired modularity function for unsupervised learning involving spatially embedded networks. Scientific reports 9(1), 2631 (2019)
21. Lancichinetti, A., Fortunato, S.: Community detection algorithms: a comparative analysis. Physical review E 80(5), 056117 (2009)
22. Lancichinetti, A., Fortunato, S.: Limits of modularity maximization in community detection. Physical review E 84(6), 066122 (2011)
23. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical review E 78(4), 046110 (2008)
24. Newman, M.E.: Assortative mixing in networks. Physical review letters 89(20), 208701 (2002)
25. Newman, M.E.: Mixing patterns in networks. Physical review E 67(2), 026126 (2003)
26. Newman, M.E.: Modularity and community structure in networks. Proceedings of the national academy of sciences 103(23), 8577–8582 (2006)
27. Slota, G.M., Berry, J., Hammond, S.D., Olivier, S., Phillips, C., Rajamanickam, S.: Scalable generation of graphs for benchmarking HPC community-detection algorithms. In: IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC) (2019)
28. Slota, G.M., Garbus, J.: A parallel LFR-like benchmark for evaluating community detection algorithms. In: The 5th IEEE Workshop on Parallel and Distributed Processing for Computational Social Systems (ParSocial) (2020)
29. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: is a correction for chance necessary? In: Proceedings of the 26th annual international conference on machine learning. pp. 1073–1080 (2009)
30. Winlaw, M., DeSterck, H., Sanders, G.: An in-depth analysis of the chung-lu model. Tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States) (2015)
31. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: The state-of-the-art and comparative study. Acm computing surveys (csur) 45(4), 1–35 (2013)