Limitations of Chung Lu Random Graph Generation

Christopher Brissette[1], and George Slota[2]

[1] Rensselaer Polytechnic Institute, Troy NY 12180, USA,
`brissc@rpi.edu`
[2] Rensselaer Polytechnic Institute, Troy NY 12180, USA,
`slotag@rpi.edu`

Abstract. Random graph models play a central role in network analysis. The Chung-Lu model, which connects nodes based on their expected degrees, is of particular interest. It is widely used to generate null-graph models with expected degree sequences. In addition, these attachment probabilities implicitly define network measures such as modularity. Despite its popularity, practical methods for generating instances of Chung-Lu model-based graphs do relatively poor jobs in terms of accurately realizing many degree sequences. We perform a theoretical analysis of the Chung-Lu random graph model in order to understand this discrepancy. We approximate the expected output of a generated Chung-Lu random graph model with a linear system and use properties of this system to predict distribution errors. We provide bounds on the maximum proportion of nodes with a given degree that can be reliably produced by the model for both general and non-increasing distributions. We additionally provide an explicit inverse of our linear system and in cases where the inverse can provide a valid solution, we introduce a simple method for improving the accuracy of Chung-Lu graph generation. Our analysis serves as an analytic tool for determining the accuracy of Chung-Lu random graph generation as well as correcting errors under certain conditions.

Keywords: graph theory, random graphs, graph generation

1 Introduction

Say we wish to generate a random simple graph $G = (V, E)$ with a degree distribution $y = \{N_1, N_2, \cdots, N_m\}$ where N_k represents the numbers of nodes with degree k. Here, simple means that there are no self-loops or multi-edges. This is a problem that arises in many network science applications, most notably for the generation of null-models used for basic graph analytics [12]. Generating such simple networks exactly using the explicit configuration model, or erased configuration model is computationally expensive for even moderately large networks [7]. This is in part because the explicit configuration model is difficult to parallelize, and partly because correcting self-loops and multi-edges in the erased configuration model can be cumbersome. As such, we rely on probabilistic methods for large-scale graph generation that only match y in expectation.

The Chung-Lu random graph model [4] is one such widely-used probabilistic model. This model pre-assigns to each node $v_i \in V(G)$ a weight w_i corresponding the the degree we wish for the node to have. It then connects all nodes v_i, v_j pairwise with the probability $p_{ij} = \frac{w_i w_j}{\sum w_k}$. It is known that the degree of each node in the output graph will match its pre-assigned weight in expectation. There are a number of ways that generating such graphs can be done computationally. Some methods generate loops and multi-graphs, while others generate simple graphs. We focus on what is sometimes called the Bernoulli Method for generating Chung-Lu graphs [16], as it is amenable to the edge-skipping technique [11] that allows linear work complexity and near-constant parallel time for scalable implementations [1, 14, 8]. In this method, we implicitly consider all possible pairs of edges between unique nodes and generate edge (v_i, v_j) with $i \neq j$ according to the probability p_{ij}. This generates a simple-graph with degree sequence $\tilde{y}$ where $\mathbb{E}[\tilde{y}] = y$. While we focus our analysis on this specific variation, as it is the one most likely to be used in practice, other methods that generate multi-edges and/or loops have many of the same issues that we discuss in this paper.

The Chung-Lu model, though popular and theoretically sound under the tame condition that $w_i w_j < \sum_{k=1}^{m} w_k$ for all $v_i, v_j \in V$, can produce degree distributions drastically different from the desired expectation in practical settings. This has been widely noted and addressed in the literature [16, 3, 8, 2, 9, 13, 5]. To theoretically motivate why this is the case, consider generating a graph that has many nodes of degree two. While Chung-Lu guarantees that the expected degree of each of these nodes will be two, it makes no other guarantees regarding the probability mass function of these degrees. Indeed in practice, nodes with expected degree two often have degree 0, 1, 3, and beyond. This is particularly challenging when Chung-Lu generation is utilized as a subroutine for more complex graph generation, such as when generating graphs that also match a clustering coefficient distribution (e.g., the BTER model) [10] or a community size distribution for community detection benchmarking [15, 14]. In these and other instances, minimizing error in the degree distribution is critical, as this error can propagate through the rest of the generation stages. In Figure 1 we see an example of the observed error when generating some graphs. As can be seen, the output of Chung-Lu in both cases underestimates the number of degree one nodes, and accrues additional error from other low degree families as well. This suggests that instead of strictly caring about the expected degree of each node in Chung-Lu generation, as is generally done, we should additionally consider deeper statistical properties of the model in application.

To better understand the output of Chung-Lu, consider grouping all nodes by expected degree. That is, take degree families $d_k = \{v_i \in V : w_i = k\}$ and consider connections between them. From the point of view of a single node $v_j \in V$ with expected degree w_j the number of connections it has to each degree family d_k is binomially distributed with mean $\frac{k w_j}{\sum w_i} |d_k|$. Therefore the degree distribution of the nodes in d_{w_j} is the sum of m independent binomial random processes where m is the maximum expected degree of the graph. This allows us

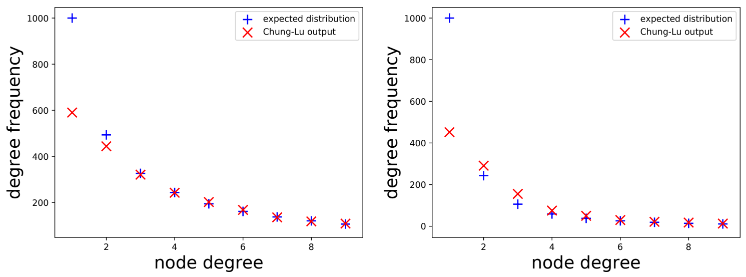

Fig. 1. Distribution error of Chung-Lu. We consider the degree classes between one and nine for two different power law distributions. On the left is a power-law distribution with exponent $\beta = 1.0$ and on the right is a power-law distribution with exponent $\beta = 2.0$. In the top two plots, crosses represent the input distribution and x's represent the average distribution for 20 instances of Chung-Lu graphs given the power law distribution distribution as input. We can see that the Chung-Lu generated graphs drastically under-represent degree one nodes. This is a phenomenon that commonly occurs in application and can greatly affect generation accuracy.

to go beyond simply guaranteeing the mean of each degree family, and instead predict the probability mass function for the degrees of each of these families, and by extension predict degree distribution errors.

Since the degree distribution of each degree family d_k in our graph is binomially distributed, we may apply a further approximation. Because the limiting case of the binomial distribution is the Poisson distribution, we approximate the number of connections between nodes in a given degree family with all other nodes as a sum of Poisson distributions, which is again Poisson. We note that often times a desired degree distribution will be such that certain degree classes will not have the number of nodes required for this approximation to have guaranteed accuracy. In fact, power-law degree distributions will in general have degree families d_k where $k \approx m$ such that $|d_k| \approx 1$. However, we also note that this additionally means the node-wise error contributed by those families is relatively small, so we are willing to sacrifice some accuracy in lieu of a cleaner description.

Say that X_{ij} is the Poisson distribution representing the degree of each node in d_i if d_i only connected to nodes in d_j. Additionally, take the mean of this Poisson distribution to be γ_{ij}. Because the means of independent Poisson distributions are additive, we have the following linear system, describing the means of each distribution.

$$\begin{bmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1m} \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{m1} & \gamma_{m2} & \cdots & \gamma_{mm} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_m \end{bmatrix} \tag{1}$$

This matrix provides additional rationale for our Poisson approximation. Since we assumed the distributions were Poisson we may now add means of Poisson distributions directly as opposed to computing with more complex independent binomial distributions. In the case of the Chung-Lu model, each $\gamma_{ij} = \frac{w_i w_j}{\sum w_k}$. This, perhaps as expected, gives the right hand means of $\mu_k = k$. This means that the degrees within each degree family d_k should be approximately Poisson distributed about k. Before moving on we note that a similar analysis can be done for any connection probabilities. While we are focusing on Chung-Lu probabilities, this model also describes the output degree sequence for any set of chosen p_{ij} between degree families, albeit with potential changes to the means μ_i.

Given the description offered by Equation 1 we now have the tools to estimate the output of Chung-Lu through a simple linear system. Consider an input degree distribution $y = [N_1, N_2, \cdots, N_m]^T$ as a vector in $\mathbb{R}^m$ with the number of nodes being, $N = \sum_{i=1}^{m} N_i$. Additionally take $poiss(k)$ to be the probability density function of the Poisson distribution with mean k. We can calculate the expected output $\tilde{y}$ of this as follows.

$$\mathbf{Q}y = \begin{bmatrix} | & | & & | \\ poiss(1) & poiss(2) & \cdots & poiss(m) \\ | & | & & | \end{bmatrix} \begin{bmatrix} N_1 \\ N_2 \\ \vdots \\ N_m \end{bmatrix} = \tilde{y} \tag{2}$$

This construction works in the following way. Each column of our matrix represents the probability mass function of degrees within each degree family. Taking the inner product of a row r of this matrix with a vector of degree family sizes amounts to adding together the expected number of degree r nodes produced by each degree family under the Chung-Lu algorithm. So, by considering the action of the entire matrix, we are considering the action of Chung-Lu as a whole. Note here we are assuming $poiss(k)$ is the full, discrete version of the Poisson distribution with mean k. This implies that the system in Equation 2 maps $\mathbb{R}^m$ to an infinite dimensional space. This is not computationally useful. We therefore truncate the Poisson matrix $\mathbf{Q}$ to be square in $\mathbb{R}^{m\times m}$ by removing the first row corresponding to degree zero nodes, as well as everything below the m^{th} row. We will denote this matrix by $\mathbf{P}$. Our justification for this truncation is two-fold. One, we are inputting a degree distribution in $\mathbb{R}^m$, and we mainly only care about error with regards to those output degrees between one and m. Two, making the matrix square allows for us to invert the matrix which will be useful for generating Chung-Lu graphs with more accurate degree sequences. Note that truncating $\mathbf{Q}$ to some dimension m amounts to ignoring nodes with degree zero as well as nodes with degree higher than m. If we wish to obtain error information for higher degrees as well we can easily append zeros to the end of our input distribution and consider $\mathbf{P}\in\mathbb{R}^{n\times n}$ where $n > m$ and m is the maximum degree of our desired distribution. Then, for large enough n, our error is only ignoring nodes of degree zero. In a practical setting, these nodes would possibly be thrown out and ignored, anyways. The rest of this paper discusses properties of $\mathbf{P}$, the limitations these properties suggest, and how the matrix can be used to improve the accuracy of Chung-Lu outputs in some cases.

1.1 Our Contributions

As noted, while Chung-Lu graph generation is a useful tool for many theoretical purposes and is used widely in fields such as social network analysis, it often does a poor job of approximating distributions at the ends. The specific issue considered in this paper is that Chung-Lu generated networks will often under-represent low degree nodes. In Figure 1, we can clearly see that actual Chung-Lu realizations may easily contain less than 60 percent of the desired number of degree one nodes. This can lead to a great deal of inaccuracy for distributions with particularly large numbers of low degree nodes, such as power-law distributions. In practice, this generally means that generated graphs will have many vertices of degree zero, so one way of resolving this issue is to connect these nodes to the graph in order to inflate the number of degree one nodes. Depending on the degree distribution, this can easily skew other degree classes without careful choice of where these nodes are connected. This may also require considerable computation. For this reason, it is far simpler for applications to throw away degree zero nodes. Other proposed methods might artificially inflate the input distribution in terms of degree one nodes so the output better matches the desired input [10], but this also presents similar challenges.

For this reason, we suggest the matrix model referenced in the introduction. The standard input distribution for Chung-Lu is simply the desired output distribution y. We suggest a "shifted" Chung-Lu algorithm where, given a matrix model $\mathbf{P}$ for the output of the Chung-Lu algorithm, we take our desired output distribution y and solve for $x = \mathbf{P}^{-1}y$. Then the input to a Chung-Lu graph generator is x as opposed to the desired output. This is particularly compelling since the matrix $\mathbf{P}^{-1}$ only depends on the maximum degree of our desired output distribution and once computed allows for drastic accuracy improvement at negligible algorithmic cost. While useful in certain special cases, we find that such an algorithm is not possible in general. We prove several the matrix $\mathbf{P}$ is invertable and show that many distributions do not have non-negative inverses. We investigate these cases and classify some instances in which an inverse distribution is guaranteed to have negative entries. Most interestingly, we provide tight bounds on the expected maximum number of nodes that may belong in each degree family for both non-increasing as well as general distributions. These bounds suggest that there exists a vast number of graphs that Chung-Lu generation is ill-equipped to generate.

2 Properties of the matrix model

From the introduction, we use the assumption that the degree distribution of each family is approximately Poisson distributed to form a matrix that will transform input distributions into approximate output distributions from the Chung-Lu model. Assume that our input distribution has degrees in $\mathbb{N}_m = \{1, \cdots, m\}$

and is represented by $x = [N_1, N_2, \cdots, N_m]^T$ where N_k represents the number of nodes with expected degree k and $N = \sum_{k=1}^m N_k$. We can represent our matrix $\mathbf{P}$ in terms of the following factorization.

$$\begin{aligned}\mathbf{P} &= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \frac{1}{2!} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{m!} \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & m \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{m-1} & \cdots & m^{m-1} \end{bmatrix} \begin{bmatrix} e^{-1} & 0 & \cdots & 0 \\ 0 & 2e^{-2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & me^{-m} \end{bmatrix} \\ &= \mathbf{AVB}\end{aligned} \tag{3}$$

When this factorization is multiplied out, we obtain exactly the $\mathbf{P}$ matrix discussed in the introduction. Note that realizing a Chung-Lu graph model amounts to computing $\mathbf{P}x$ for some pre-defined x. We instead look at the inverse problem of determining $x \in \mathbb{R}^{+m}$ given $\mathbf{P} \in \mathbb{R}^{m \times m}$ and desired output $y \in \mathbb{R}^{+m}$. Here $\mathbb{R}^{+m}$ is the element-wise positive region of $\mathbb{R}^m$. This amounts to solving the linear system $\mathbf{P}x = y$. One may be tempted to simply invert this matrix using any number of computational methods, and this is reasonable for small m. However, given the factorization in Equation 3, we have that $\mathbf{P} = \mathbf{AVB}$ with $\mathbf{V}$ a Vandermonde matrix. Due to the extremely poor conditioning of both $\mathbf{A}$ and $\mathbf{V}$, using a computational method for inverting $\mathbf{P}$ is not advised. Fortunately $\mathbf{A}$ and $\mathbf{B}$ are diagonal, meaning they are easy to invert, so finding the inverse of $\mathbf{P}$ only requires finding an inverse to $\mathbf{V}$. Again, we do not want to compute this using standard computational methods, since Vandermonde matrices are the textbook examples of nearly uninvertible matrices. Fortunately, our Vandermonde matrix is such that it has a special structure yielding a somewhat simple closed-form inverse given in [6]. It relates each entry in the matrix to associated binomial coefficients and Stirling numbers of the first kind. Explicitly, each entry is expressed as follows.

$$\mathbf{V}_{ij}^{-1} = (-1)^{i+j} \sum_{k=max(i,j)}^{n} \frac{1}{(k-1)!} \binom{k-1}{i-1} \begin{bmatrix} k \\ j \end{bmatrix} \tag{4}$$

For distributions with entry-wise positive inverses we can now compute the input of Chung-Lu that will best approximate the desired output according to $\mathbf{P}^{-1}y = x$. The actual implementation of this would look like the pseudocode given in Algorithm 1 where $\lfloor \cdot \rfloor$ represents element-wise rounding down to the nearest integer.

2.1 Not all solutions are positive

We now concern ourselves with cases where Algorithm 1 will fail. These cases will occur exactly when $\mathbf{P}^{-1}y$ has negative entries. To understand why this is the case, consider that $\mathbf{P}^{-1}y$ represents a degree distribution. Negative entries in this vector therefore represent a meaningless value as an input to the Chung-Lu algorithm. Matrix $\mathbf{P}$ has only positive real entries. This implies that for any

Algorithm 1 ShiftedChungLu (y,d_{max})

```
1: S ← ComputeStirlingMatrix($d_{max}$)
2: $A^{-1}$ ← ComputeDiagInverse(A)
3: $B^{-1}$ ← ComputeDiagInverse(B)
4: $V^{-1}$ ← ComputeVandermondeInverse(S,$d_{max}$)
5: $\tilde{x}$ ← $\lfloor B^{-1}V^{-1}A^{-1}y \rfloor$
6: $G$ ← GenerateChungLu($\tilde{x}$)
7: return $G$
```

element-wise positive vector x, $\mathbf{P}x$ is also positive. While this implies that any positive input will yield an approximately valid result, it does not exclude the possibility of vectors with negative entries also mapping into the positive region of $\mathbb{R}^m$ under the action of $\mathbf{P}$. This means that we may not be able to use the output of $\mathbf{P}^{-1}y = x$ as the input of $\mathbf{P}x$ since x has the possibility of containing negative elements. In Figure 2, we can see what the action of $\mathbf{P}$ looks like on a sample of random vectors for $\mathbf{P}{\in}\mathbb{R}^4$. Notice how, as expected, it "squishes" the positive region into a small sliver.

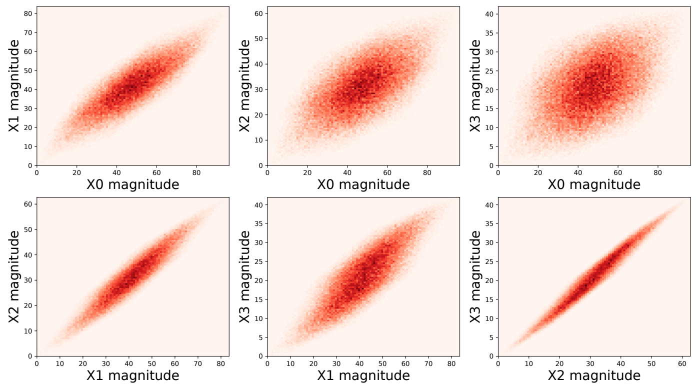

Fig. 2. Action of P on the positive hypercube. Here we can see plots of projections of random vectors under the action of $\mathbf{P}$ as a heat map. The sample consists of 100,000 random vectors with random integer entries selected to be within $\{0, \cdots, 100\}$ under the action of $\mathbf{P} \in \mathbb{R}^{4\times 4}$. The output vectors are then projected onto each canonical unit vector $e_j \in \mathbb{R}^4$ and plotted pairwise. These vectors are referred to as Xi in the axis labels. Intuitively this shows all feasible output from a Poisson random graph model with node degrees limited to those in $\{1, 2, 3, 4\}$. We can see that all positive vectors remain inside the positive region as expected, and we also see how sharply limiting this is for finding positive solutions of $\mathbf{P}^{-1}y$ for y positive.

Given a number of nodes N we look to bound how many nodes of each degree are feasible. That is, if we have some degree distribution x with L1-norm

$\|x\|_1 = N$ we wish to find lower and upper bounds, l_i and u_i respectively on $|(\mathbf{P}x)_i|$ such that $l_i \leq |(\mathbf{P}x)_i| \leq u_i$. We want to do this for every degree family. Take the projector $\rho_i = e_i^T e_i$ where e_i is the i^{th} canonical unit vector in $\mathbb{R}^m$. Then we know $|(\mathbf{P}x)_i| = \|\rho_i \mathbf{P}x\|_1$. This directly implies from the structure of $\mathbf{P}$ that we have,

$$N \underset{k}{min} |\mathbf{P}_{ik}| \leq \|\rho_i \mathbf{P}x\|_1 \leq N\mathbf{P}_{ii} \,\forall\, x \in \mathbb{R}^m \,:\, \|x\|_1 = N \tag{5}$$

Under the necessary, but reasonable, assumption that $N > m$, Equation 5 gives us a tight upper bound on the number of nodes we can reliably generate of a given degree based on only the number of nodes in our distribution. This bound is realized precisely when all of the nodes in our distribution have input degree $\frac{N}{i}$. We may be interested in what outputs a more narrow space of input distributions can reliably generate. Consider bounding the number of nodes with given degrees in a special case. Namely we pick degree family sizes such that the following is true.

$$N_1 \geq N_2 \geq \cdots \geq N_m \tag{6}$$

That is, the size of the families are non-increasing with respect to input degree. This classifies a wide variety of networks ranging from those with identical family sizes, to power-law distributions. We wish to upper bound the number of nodes we can generate in a given degree family j with a distribution following the Property 6. This problem can be expressed in terms of finding coefficients satisfying Equation 7. Here we may take coefficients $\|a\|_1 = 1$ and then generalize by taking $x = Na = [N_1, N_2, \cdots, N_m]^T$.

$$\max_a \sum_{k=1}^{m} k^j \frac{e^{-k}}{j!} a_k \tag{7}$$

We can see the maximum occurs when a_j has a maximal population. This means that, perhaps as expected, the way to achieve the maximum number of nodes with degree j is to maximize the number of input nodes with degree j. Since our function is nonincreasing this means this maximum occurs when $a_1 = a_2 = \cdots = a_j$ and $a_{j+1} = a_{j+2} = \cdots = a_m = 0$. This directly implies that we will get the most nodes of degree j when the following is true for $\|a\|_1 = 1$.

$$a_1 = a_2 = \cdots = a_j = \frac{1}{j} \tag{8}$$

Therefore the maximum number of nodes we should expect in a given degree class can be approximated as follows.

$$\frac{1}{j!j} \sum_{k=1}^{j} k^j e^{-k} \approx \frac{1}{j!j} \int_1^j x^j e^{-x} dx \tag{9}$$

$$\approx \frac{1}{j!j} \gamma(j+1, j) \tag{10}$$

Equation 9 gives us both the exact upper bound and continuous approximation. Equation 10 can be used as a quick approximation of this value in terms of the incomplete gamma function from 0 to j. This gives a far tighter bound than is provided by Equation 5 when we have a non-increasing degree distribution. It should be noted that one may improve upon the accuracy of these bounds for even more restrictive families of distributions by including a lower bound as well as a tighter upper bound on the size of each degree family.

We can glean useful information from these bounds. For instance, if one desires an output distribution where more than a tenth of the nodes have degree five, there are no non-increasing inputs for which we should expect that property in output. In terms of the inverse matrix $\mathbf{P}^{-1}$, inputting such a vector will yield negative family sizes in some indices. This is incredibly limiting since this is independent of node number.

3 Results

We wish to determine how well $\mathbf{P}$ models the output of the Chung-Lu algorithm for a given input distribution. In Figure 3 we compare the naïve output distribution to the outputs of both Chung-Lu generation and our model taking that distribution as input. For this simple example we find that our model predicts the output node degree frequency remarkably well.

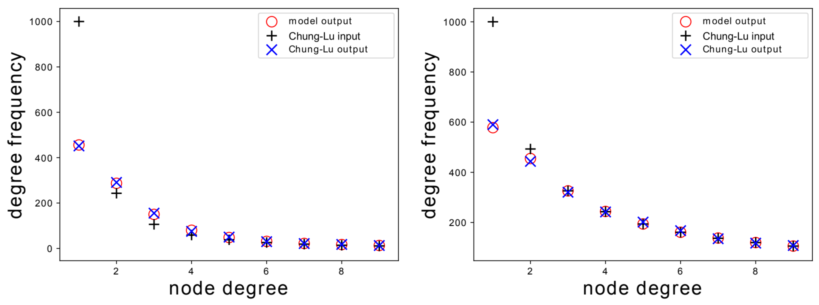

Fig. 3. Model distribution versus Chung-Lu outputs. We consider the degree classes between one and nine for two different power law distributions. On the left is a power-law distribution with exponent $\beta = 1.0$ and on the right is a power-law distribution with exponent $\beta = 2.0$. In the top two plots, black crosses represent the naïve input $1000 \times k^{-\beta}$, red circles represent the distribution our model estimates will be the output of Chung-Lu generation, and blue x's represent the average distribution for 20 instances of Chung-Lu graphs given the black crosses as input. We can see that the Chung-Lu generated graphs match our model output remarkably closely.

Additionally we aim to determine how much proportional L1 accuracy is gained by using the vector $x = \mathbf{P}^{-1}y$ as opposed to y itself as an input to Chung-Lu. Specifically, we consider generating a set of graphs $\{y_i\}$ using the Chung-Lu algorithm with the naïve inputs $\{y_i\}$, and the shifted inputs $\{x_i = \mathbf{P}^{-1}y_i\}$. We plot the proportional L1 errors $\frac{\|y_i - \tilde{y}_i\|_1}{\|y_i\|_1}$ and $\frac{\|y_i - \tilde{x}_i\|_1}{\|y_i\|_1}$ in Figure 4 where $\tilde{y}_i$ and $\tilde{x}_i$ are the output distributions of Chung-Lu for the naïve input and shifted input respectively. we choose our set $\{y_i\}$ such that these are guaranteed to be invertible distributions in the sense that $x \in \mathbb{R}^{+m}$. For this we use the variable precision toolbox in MATLAB with the digits of precision set to 100. The results of this can be seen in Figure 4. We find that our shifted input drastically decreases the proportional L1 error between the output of Chung-Lu and the desired output.

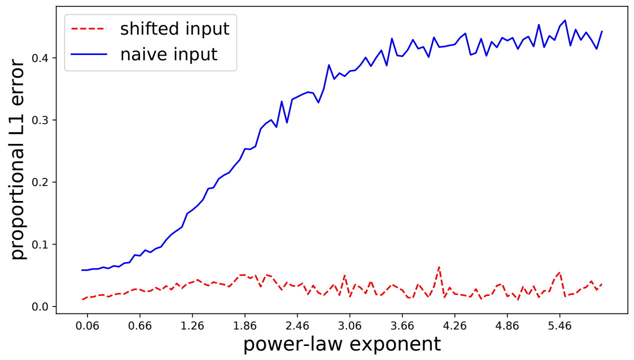

Fig. 4. Error of naïve Chung-Lu input versus shifted Chung-Lu input. We consider 100 input distributions y_i such that $\mathbf{P}^{-1}y_i = x_i$ where the distribution x_i is the power-law distribution $1000 \times k^{-\frac{6i}{100}}$ with k ranging between 1 and 40. For each of the 100 inputs, 30 graphs were generated and their degree distributions were averaged using the input y_i for Chung-Lu. The proportional L1 error between this output and the desired output y_i is shown as the solid blue line. Additionally 30 graphs were generated and their degree distributions were averaged using the input x_i for Chung-Lu. The proportional L1 error between this output and the desired output y_i is shown as the dashed red line. We can see that the "shifted" input we get using our model drastically reduces error for the sample.

4 Conclusion

We have provided a simple method for estimating the output of Chung-Lu random graph generators with far lower proportional L1 error than that given by the traditional assumption that output distributions will resemble input distributions. Our method utilized a Poisson estimate for the number of nodes of given degrees and we used this to define an invertible matrix $\mathbf{P}$ that models the expected output from Chung-Lu generators. This allowed us to "solve the problem in reverse" and take a desired output y and solve for the Chung-Lu input

x that will result in y. We called this the shifted Chung-Lu input. We showed $\mathbf{P}$ predicts that many degree distributions simply are not feasible for Chung-Lu generators, however we provide conditions for classifying a large portion of these distributions.

There are several avenues for further research. For instance, this work lends itself to analysis and improvement of graph generation. Methods which use naïve Chung-Lu generation as a subroutine may gain both accuracy and insight into possible distribution errors through the kind of analysis done in this paper. Further work may also be done on how altering connection probabilities between degree classes may be used to fine tune the matrix $\mathbf{P}$ in order to produce graphs which are inadvisable for naïve Chung-Lu generation.

References

1. Alam, M., Khan, M., Vullikanti, A., Marathe, M.: An efficient and scalable algorithmic method for generating large-scale random graphs. In: SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 372–383. IEEE (2016)
2. Britton, T., Deijfen, M., Martin-Löf, A.: Generating simple random graphs with prescribed degree distribution. Journal of Statistical Physics 124(6), 1377–1397 (2006)
3. Chodrow, P.S.: Moments of uniform random multigraphs with fixed degree sequences. SIAM Journal on Mathematics of Data Science 2(4), 1034–1065 (2020)
4. 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)
5. 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)
6. Eisinberg, A., Franzé, G., Pugliese, P.: Vandermonde matrices on integer nodes. Vandermonde matrices on integer nodes 1(80), 75–85 (1998)
7. 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)
8. 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)
9. van der Hofstad, R.: Critical behavior in inhomogeneous random graphs. Random Structures & Algorithms 42(4), 480–508 (2013)
10. Kolda, T.G., Pinar, A., Plantenga, T., Seshadhri, C.: A scalable generative graph model with community structure. SIAM Journal on Scientific Computing 36(5), C424–C452 (2014)
11. Miller, J.C., Hagberg, A.: Efficient generation of networks with given expected degrees. In: International Workshop on Algorithms and Models for the Web-Graph. pp. 115–126. Springer (2011)
12. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science 298(5594), 824–827 (2002)
13. Pfeiffer III, J.J., La Fond, T., Moreno, S., Neville, J.: Fast generation of large scale social networks with clustering. arXiv preprint arXiv:1202.4805 (2012)

14. 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)
15. 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)
16. 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)