Multi-edge Randomizer: Additional details on simulations

O. Sagarra[1] and F. Font-Clos[2, 3]

[1]*Departament de Física Fonamental, Universitat de Barcelona, 08028 Barcelona, Spain*
[2]*Centre de Recerca Matemàtica, Campus de Bellaterra, Edifici C - 08193 Bellaterra (Barcelona), Spain*
[3]*Departament de Matemàtiques, Universitat Autònoma de Barcelona, Edifici C - 08193 Bellaterra (Barcelona), Spain*

I. GENERATION OF SYNTHETIC NETWORKS ON DIFFERENT ENSEMBLES AND THE EFFECT OF SELF-LOOPS

This document constitutes the manual for the **Multi Edge Randomizer** software, where details on the implementation of the simulated ensembles are provided. For more details see [1] and [2].

The code accompanying the present documents allows for the generation of both directed and undirected multi-edge networks with a prescribed strength sequence (optionally including or excluding self-loops) in the various ensembles. In the following section a description of the simulation procedure for each ensemble is given. The code for simulating any ensemble can be downloaded from [3]. It must be noted that all node related quantities computed are averaged only over the realizations where a given node exists ($s_i \neq 0$), since the probability for a node to have strength 0 is not zero in the GC and Canonical (C) ensembles (albeit rapidly decreases with $\hat{s}$).

A. Micro canonical Ensemble

To generate a multi-edge network in the micro-canonical ensemble, one just needs to apply the well-known configurational model schema [4] (this method was also used in [5]) and *rewire* the link connections permitting multiple connections between nodes.

In the case of allowing self-loops, the expected number of events joining nodes i and j reads,

$$\langle t_{ij} \rangle^{\text{micro}} = \begin{cases} \frac{\hat{s}_i \hat{s}_j}{\hat{T}} & \text{if } i \neq j \\ \frac{\hat{s}_i(\hat{s}_i - 1)}{\hat{T}} & \text{otherwise.} \end{cases} \tag{1}$$

And so for each node, the error committed comparing with the other ensembles is only due to self-loops, $\Delta_{\langle s \rangle} = \frac{\hat{s}_i}{\hat{T}} \leq 1$ which can only be an important quantity for large $\hat{s}$, but in this case the relative importance of self-loops with respect to the strengths is completely negligible.

It is very important to consider that despite existing many algorithms to *reshuffle* networks preserving the strength sequence, only some of them explore the phase space in an unbiased (or almost unbiased) manner. In our case, the only problem the configurational model has is that it does not allow for self-loops on nodes with strength 1, since they appear always in couples. However, this is not a big issue since the presence of self-loops is only important for high strength nodes while its probability of appearance for nodes with strength 1 is effectively zero.

If one wishes not to accept self-loops, then two alternative approaches are available: one can chose just to not connect the connections corresponding to self-edges (thus not fixing the strength sequence exactly) if rewiring fail repeatedly for sufficiently large number of trials or discard entirely the configurations for which a self-loop event exists.

The first approach is the option chosen in this software, but note that this induces node correlations in the model and hence the ensemble space is not explored evenly (losing the strict micro canonical nature), it does not not fix the strength distribution exactly and furthermore for exponents $\gamma \leq 2$ the number of rejections can grow significantly, making simulations extremely lengthy.

The second approach, on the other hand, is unfeasible in finite time: The probability for a given node of strength $\hat{s}_i$ in the configuration model allowing self-loops while joining stubs to have a self-loop is $p^i_r = \frac{\hat{s}_i - 1 - r}{\hat{T}}$ being r the number of already existing self-loops for the considered node. The number of self-events per node sl_i is thus the result of a random incremental process with step depending probabilities p^i_r. The probability to have no self-loops at the end of the process for each stub is approximately $p^i_0 = \left(1 - \frac{\hat{s}_i - 1}{\hat{T}}\right)$ and approximating the states (also self-states) as independent variables, the probability to obtain exactly 0 self loops $SL = \sum_i sl_i$ in the network reads,

$$P(SL = 0) = \prod_i \left(1 - \frac{\hat{s}_i - 1}{\hat{T}}\right)^{\hat{s}_i} \tag{2}$$

(but this is only an upper bound since as stubs get connected, $\hat{T}$ decreases in "time"). In any case, the earlier expression rapidly vanishes due to the large number of terms in the l.h.s. product which are strictly smaller than 1, hence a configuration approach discarding absolutely the configurations with self-edges is not feasible in practice.

The complexity of this algorithm is of order $\mathcal{O}(T)$ (the length of the stub sequence). In this scenario, the occupation numbers obtained are, in general, correlated.

B. Canonical Ensemble

For the canonical ensemble, the statistics of occupation numbers is multinomial with associated probabilities $p_{ij} = \frac{\hat{s}_i \hat{s}_j}{\hat{T}^2}$ and $\hat{T}$ trials. To avoid self-edges one can set $p_{ii} = 0 \, \forall i$. This method has a limited applicability with system size, since the generation of multinomial distributed variables is not independent and requires a large amount of memory, due to the fact that the occupation numbers generated are correlated,

$$\sigma_{t_{ij},t_{kl}} = \begin{cases} -Tp_{ij}p_{kl} & ij \neq kl \\ Tp_{ij}(1-p_{kl}) & ij = kl \end{cases} \tag{3}$$

C. Grand Canonical Ensemble

The grand canonical ensemble can be implemented in two alternative yet equivalent approaches. One can generate a Poisson distributed number τ with mean T and then generate a collection of occupation numbers $\{t_{ij}\}$ using a multinomial distribution of τ trials and probabilities $p_{ij} = \frac{\hat{s}_i \hat{s}_j}{\hat{T}^2}$ or alternatively one can generate a sequence of L independent Poisson occupation numbers with mean $\langle t_{ij} \rangle = \frac{\hat{s}_i \hat{s}_j}{\sum \hat{s}_i}$. We have chosen the latter approach to avoid memory overload problems (in this case the occupation numbers are independent and can be generated accordingly). The complexity of this algorithm scales with the number of possible states L. Note that in this case self-edges can be manually avoided by setting $\langle t_{ii} \rangle = 0$ (or equivalently $p_{ii} = 0$) $\forall i$.

II. SYNTHETIC POWER LAWS WITH PRESCRIBED AVERAGE STRENGTH

In the folder `strengths` we provide the strength sequences used in **??**. Since in the ensemble approach used the thermodynamic limit is defined in terms of $T = \sum s_i$, we use synthetic distributions generated with tunable average strengths. This fact poses problems for the case of power law distributed strengths, since,

$$\bar{s}(\gamma, s_{min}, s_{max}) = \frac{1}{\sum_{s_{min}}^{s_{max}} s^{-\gamma}} \sum_{s_{min}}^{s_{max}} s_i^{1-\gamma}. \tag{4}$$

In such case one has two scenarios: $\gamma > 2$ and $\gamma \leq 2$. For the first scenario, the effect of s_{max} is negligible, and one can achieve an (approximately) desired $\bar{s}$ by setting an appropriate s_{min}. For the second scenario, the opposite happens and one needs to apply a cut-off on s_{max} to limit the average strength of the sequence (which would be unbounded in the case of infinite sampling).

[1] O. Sagarra, C. J. Pérez Vicente, and A. Díaz-Guilera, Phys. Rev. E **88**, 062806 (2013).
[2] O. Sagarra and F. Font-Clos, arXiv Prepr. arXiv . . . , 1 (2014), arXiv:arXiv:1404.3697v1 .
[3] O. Sagarra, "Multi Edge Network Generator. https://github.com/osagarra/Multi_edge_randomizer," (2014).
[4] M. Molloy and B. Reed, Random Struct. Algorithms **6**, 161 (1995).
[5] M. Á. Serrano, in *AIP Conf. Proc.*, Vol. 776 (AIP, 2005) pp. 101–107.