Network Randomizer - User Manual

Gabriele Tosadori and Ivan Bestvina

About Network Randomizer

Network Randomizer is a Cytoscape app for generating random networks, as well as randomizing the existing ones, by using multiple random network models. Further, it can process the statistical information gained from the these networks in order to pinpoint their special, non-random characteristics.

It covers many popular random network models: Erdős–Rényi, Watts–Strogatz, Barabási–Albert, Community Affiliation Graph, edge shuffle, degree preserving edge shuffle, but it also features a new model which is based on the node multiplication.

The statistical module is based on the two-sample Kolmogorov-Smirnov test. It compares random and real networks finding the differences between them and thus providing insight into non-random processes upon which real networks are built.

Development details and FAQ can be found on: github.com/gabrielet/Network-Randomizer

Link to the app: http://apps.cytoscape.org/apps/networkrandomizer

Typical use-scenario

Rather than explaining individual features of the Network Randomizer, this manual walks you through an example of using Randomizer modules starting from a real network, and ending up with valuable statistical insights into it. More information about each random network model used can be found by clicking on the '**?**' button inside the app.

Importing and Preparing Networks

The Network Randomizer is primarily designed to work with multiple real networks simultaneously, but for the sake of simplicity, only one will be analyzed.

First step when analyzing any network with Cystoscape apps is to import the network into Cytoscape. To use a new session, click **File** > **New** > **Session**. Once a new session is created, the network needs to be loaded. Go to **File** > **Import** > **Network** > **File**.

After the network is imported, the information about it needs to be prepared in order to be used by the Network Randomizer. Randomizer's network comparison module is focused on the centrality measures and their distributions. There are multiple ways to fill in this information. One of the most simple ones is to use a Cytoscape's built-in network analysing function. To run it, go to **Tools** > **NetworksAnalyzer** > **Network Analysis** > **Analyze**

Networks. This should automatically fill in the values of various node centrality measures into the network node table.

Another, more advanced approach, is to use a popular centrality measures app CentiScaPe, which can be downloaded from the Cytoscape Apps store. It calculates many more centrality measures than the built-in method does, so it can provide useful additional information.

The third option is to import your own data. But, to work as it was designed to, the Network Randomizer needs the same centrality measures for the real and for the random networks, so both kinds should be imported.

Generating Random Networks

There are two ways in which random networks can be generated:

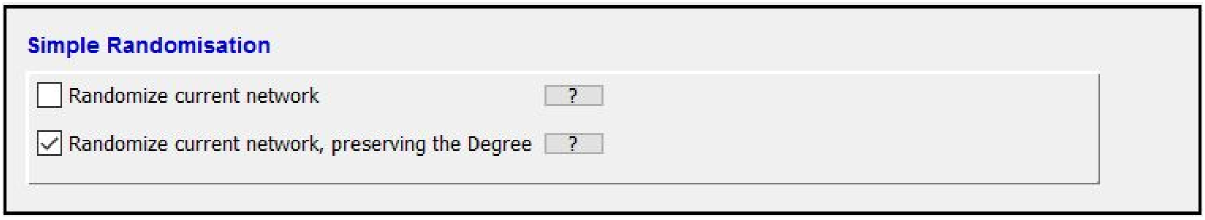
- by using the random network models;
- by randomizing current, real networks.

For best results, random networks should be made as similar as possible to the real ones, as this will allow detecting the most important differences between those two groups.

Randomizing Current Networks

To randomize a current network, one needs to be chosen first. This is done by simply creating a View: right click on the network from the Network Control Panel, and choose **Create View**. If the View already exists, just select the network by clicking on it.

Once the network is selected, there are two randomizing methods that can be used: simple edge shuffle, and degree preserving edge shuffle. Check the box in front of a chosen model, and press **Start randomization**.


Simple Randomisation

☐ Randomize current network ?

☑ Randomize current network, preserving the Degree ?

Using Random Network Models

Network Randomizer was primarily developed as an aggregation of different random network models. To use them, check the boxes in front of the models, fill in the parameters, and press **Start randomization**.

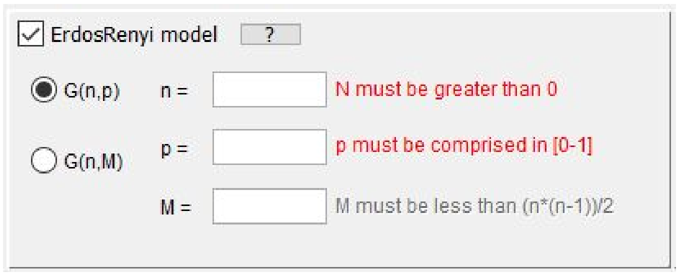
Be careful because, since we are using checkboxes, once a model is selected it remain selected for all the times you run the app. This means that, if you run the app to randomize an existing network and then, after that you would like to generate a new network without removing the check sign from the previously selected model, then the app will perform two operations: creating the new model and randomizing the existing, selected network again.

The models

Erdős–Rényi

The Erdős–Rényi model is the simplest version of random network generators. There are two distinct variants of the model:

1. G(n,M) model - the input for the G(n,M) model consists of the number of nodes (n) and the number of edges (M). After generating n nodes, it adds M edges to the network, choosing the endpoints of each edge uniformly from n generated nodes.
2. G(n,p) model - the G(n,p) model also uses the number of nodes as input, but instead of the number of edges, it uses the probability (p) of there being an edge between each pair of nodes.



N must be greater than 0

p must be comprised in [0-1]

M must be less than (n*(n-1))/2

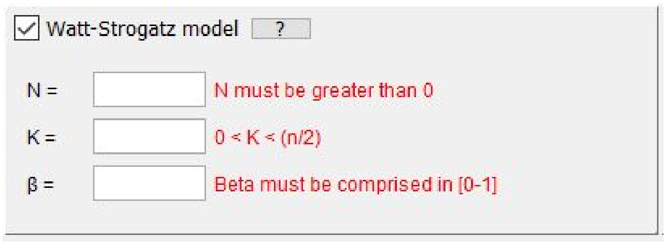
Watts–Strogatz

The Watt-Strogatz model is one of the first models to give rise to the so called *small-world phenomenon*. The following parameters are used as the input:

- N – number of nodes
- K – mean node degree, must be even
- β – probability of edge rewiring

The algorithm starts by constructing a regular ring lattice of N nodes, in which each node is connected to K/2 other nodes on each side. If K = 2, this produces a standard cycle graph. Then, for each node n_i, with probability β, algorithm rewires the edges connecting it with nodes n_j, j > i.

For β = 0, this model produces a regular ring lattice. For β = 1, it produces a completely random graph, equivalent to the Erdős–Rényi ones.



N = N must be greater than 0

K = 0 < K < (n/2)

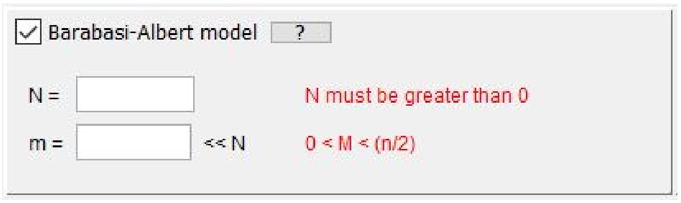
β = Beta must be comprised in [0-1]

Barabási–Albert

The Barabási–Albert model generates a scale-free network, meaning that it has a power-law degree distribution. Such networks are very often found in real-world data, which makes this model versatile. Input parameters are:

- N – total number of nodes
- m_0 – number of initial nodes (m_0 << N)
- m – initial node degree (m <= m_o)

The algorithm begins by constructing a connected graph with m_0 nodes and m*m_0/2 edges. It then iteratively adds one node at a time until there are N nodes in total. Each new node has the initial degree of m, with its m neighbours chosen with probabilities proportional to their degrees. This type of node addition is called the "preferential attachment", meaning that the more connected the node is, the more likely it is to receive new neighbours.

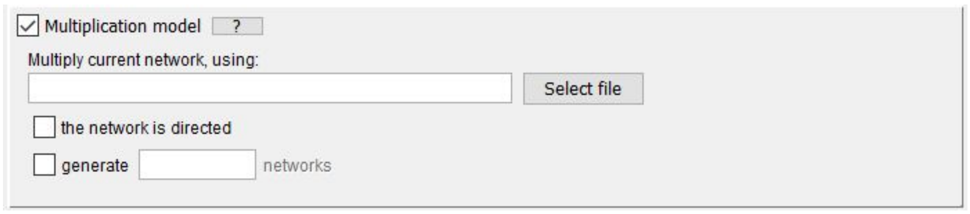


N = N must be greater than 0

m = << N 0 < M < (n/2)

Multiplicative Model (Centiscape is not ready yet to compute centralities on multiplied networks)

The multiplicative model allows creating one or more networks starting from an existing, pre-loaded network. The idea is to create networks starting from a weighting attribute. The weighting attribute defines the number of copies that can further describe the role a node has in a specific process. The point is to simulate, for instance, a proteomic experiment that is able to quantify the total number of molecules present in a specific point in time. Each node has a quantitative value which is represented in a graph by adding copies of itself in order to model the quantitative information. Once each node is multiplied, the network finely represents the high throughput data obtained by a specific technology and allows the user to investigate the properties of a weighted network.

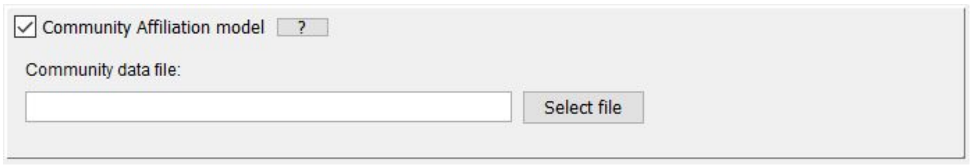


When randomising a multiplied network, the model creates a range of values in which the new network will be created. The range is built up by analysing the multiplying attribute in order to get the minimum and the maximum multiplying factor. Then it creates an array with length equal to the number of nodes in the network to randomly assign a new weight to each node which defines the new, randomly weighted network. It is possible to choose the number of networks to create, starting from a single original network.

It also allows creating multiplied networks starting from a single, non multiplied network. Here, the algorithm requires an attribute which defines the range. After defining the range, the algorithm it is able to create a number of randomly multiplied networks.

Community Affiliation Graph

The Community Affiliation Model is a recently constructed model which provides a deeper insight into communities of social networks. It generates random networks from a set of nodes and a set of communities to which those nodes belong.



Each community has a defined p value - the probability of an edge existing between each two nodes in the community. Nodes which share many common communities are more likely to be connected.

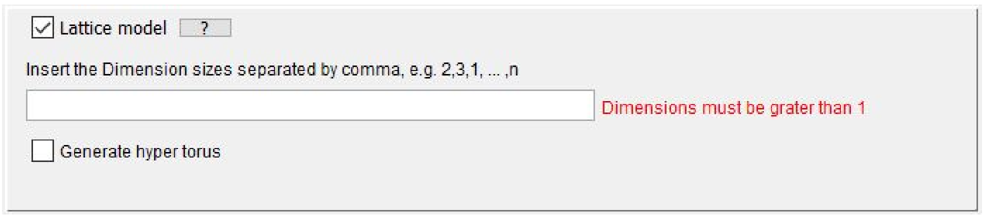
For details about this model’s input, see the in-app explanation.

Lattice graphs

The Lattice Model creates an n-dimensional lattice. Although this does not construct a random network, it is useful as a common baseline for other results.

If the "Generate torus" feature is not selected, the model generates an n-dimensional hypercube with a given set of edge lengths (dimension sizes). Nodes are connected if and only if their coordinates in the hypercube are different by 1 in only one value, and equal in all the others. In other words, they are connected if and only if they share a common side.

If the torus feature is selected, model generates an (n+1)-dimensional torus of the same size as the hypercube would be. The only difference is that, with torus selected, the opposite nodes are also connected.



Insert the Dimension sizes separated by comma, e.g. 2,3,1, ... ,n

Dimensions must be grater than 1

For example, if n = 1, choosing a torus feature would produce a cycle, if n = 2, torus would produce a real 3D torus. Not using the torus feature would result in the model producing a chain if n = 1, or a simple square lattice if n = 2.

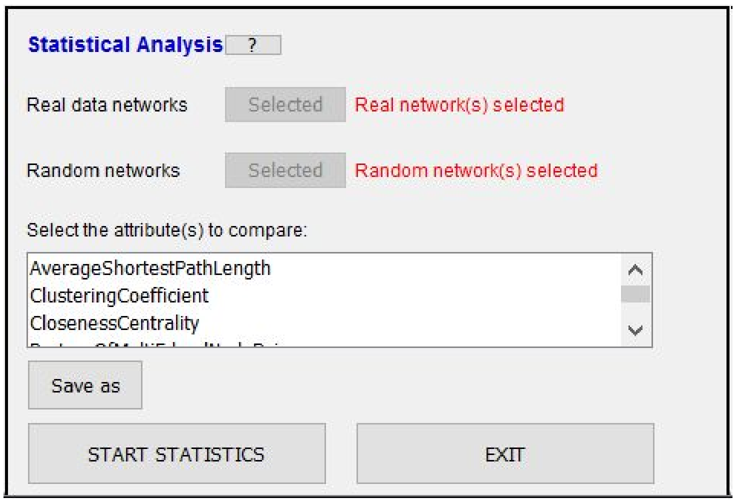
Input is given in the form of comma-separated integers, all greater than one. For example, input "3,2,3", without the torus feature, produces a 3x2x3 cuboid.

Comparing Networks

Once random networks are generated, before comparing them to the real ones, all of the centrality measures to be used should be calculated in advance. Once all the data is prepared and collected, comparing can start. First, a brief explanation of how to run the statistical analysis module will be given. After that, instructions on how to interpret the obtained results will be provided.

Running the Statistical Analysis Module

To run the statistical comparison module, it is necessary to specify which networks to use. To do this, select the networks in the Networks Control Panel and press **Selected**, once for real networks, and then once again for the random ones. After selecting the networks, a list of all the node attributes present in all of the selected networks will be provided. The user can select the node attributes he wishes to use. Selection is done in the natural manner using left-click and additional keys: **Ctrl** for one-by-one multiple selection, **Shift** for range selection, and **Ctrl+A** to select all. Once the attributes are selected, fill in the output filename and path, and click **Start statistics**.



Interpreting the Results

Every output file consists of multiple comma-separated-values fields, each beginning with an explanation marked by the '**>**' symbol. First few fields define the names of the networks used. Other fields differ depending on whether only one, or multiple real networks are used.

Single Real Network

When there is only one real network, output is either fragmented into centrality measures used, or into random networks to which the real one is compared. First field indicates how different is each random network from the real one by using the average difference across all centralities. In the second field, the Network Randomizer picks out a random network most similar to the real one according to each centrality measure individually, and specifies the difference.

The last field provides users with in-depth information, specifying the difference between real network and each random one, for every centrality measure.

Multiple Real Networks

Showing all of the generated data to the user, in the case of multiple real networks, would result in a very unreadable output. To avoid this, only the most interesting points are chosen: either the pairs of real and random networks, or the pairs of real networks and centrality measures which show the least statistical differences. This way, users can check their networks for important non-random processes.

The first field specifies the average difference across all centralities between real networks and their most similar random network. The second field chooses a random network most similar to the real one for each real network and each centrality, and specifies their difference. The last field is a real-random distance matrix, with distance defined as average difference across all centralities.