SOFTWARE TOOL ARTICLE

Creating, generating and comparing random network models with Network Randomizer [version 1; peer review: 1 approved, 1 approved with reservations]

Gabriele Tosadori^{1,4}, Ivan Bestvina², Fausto Spoto¹, Carlo Laudanna³, Giovanni Scardoni⁴

¹Department of Computer Science, University of Verona, Verona, 37134, Italy

²Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, 10000, Croatia

³Department of Medicine, University of Verona, Verona, 37134, Italy

⁴Center for BioMedical Computing, University of Verona, Verona, 37134, Italy

V1 First published: 17 Oct 2016, 5:2524
<https://doi.org/10.12688/f1000research.9203.1>

Second version: 15 Jun 2017, 5:2524
<https://doi.org/10.12688/f1000research.9203.2>

Latest published: 10 Nov 2017, 5:2524
<https://doi.org/10.12688/f1000research.9203.3>

Abstract

Biological networks are becoming a fundamental tool for the investigation of high-throughput data in several fields of biology and biotechnology. With the increasing amount of information, network-based models are gaining more and more interest and new techniques are required in order to mine the information and to validate the results. We have developed an app for the Cytoscape platform which allows the creation of randomized networks and the randomization of existing, real networks. Since there is a lack of tools for generating and randomizing networks, our app helps researchers to exploit different, well known random network models which could be used as a benchmark for validating real datasets. We also propose a novel methodology for creating random weighted networks starting from experimental data. Finally the app provides a statistical tool which compares real versus random attributes, in order to validate all the numerical findings. In summary, our app aims at creating a standardised methodology for the validation of the results in the context of the Cytoscape platform.

Keywords

random networks, cytoscape, network randomisation, network analysis, network generation

Open Peer Review

Approval Status

	1	2	3
version 3 (revision) 10 Nov 2017	 view		 view
version 2 (revision) 15 Jun 2017	 view		
version 1 17 Oct 2016	 view	 view	

1. **Giovanni Micale** , University of Catania, Catania, Italy
2. **Francesco Russo**, University of Copenhagen., Copenhagen, Denmark
3. **Ankush Sharma** , Erasmus Medical Centre, Rotterdam, The Netherlands

Any reports and responses or comments on the article can be found at the end of the article.

This article is included in the **Cytoscape** gateway.

Corresponding authors: Gabriele Tosadori (gabriele.tosadori@gmail.com), Ivan Bestvina (Ivan.Bestvina@fer.hr)

Competing interests: No competing interests were disclosed.

Grant information: This work was supported by: Italian Association for Cancer Research (AIRC, IG16797), MIUR (PRIN 2009), Nanomedicine project University of Verona and Fondazione Cariverona to Carlo Laudanna.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2016 Tosadori G *et al.* This is an open access article distributed under the terms of the **Creative Commons Attribution License**, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Tosadori G, Bestvina I, Spoto F *et al.* **Creating, generating and comparing random network models with Network Randomizer [version 1; peer review: 1 approved, 1 approved with reservations]** F1000Research 2016, 5:2524 <https://doi.org/10.12688/f1000research.9203.1>

First published: 17 Oct 2016, 5:2524 <https://doi.org/10.12688/f1000research.9203.1>

Introduction

In network analysis many tools have been developed to address different problems related to the extraction of useful information from systems modeled as graphs¹. Cytoscape² is a very well known platform, that supports hundreds of apps, that simplifies the information mining in complex networks, with a specific focus on biological applications. By using Cytoscape it is possible to perform topological analysis, cluster and motifs retrieval, biological enrichment, draw nice graphs, search for ontologies, etc. As the number of apps increases, the possibilities of performing more and more complex analysis grows together with the amount of information that could be used and retrieved. The problem is that this analysis remain, in general, preliminary to further experimental validations and, in this sense, a sort of benchmark for an *in-silico* validation is required³. A possible solution to this issue may come from the biological background of the process, from the literature or from experimental data⁴. But this is not always possible since the network may represent some complex processes whose biology is yet not well understood, or that take advantage of some novel insights that require a different validation. In these cases, when biological evidences are missing or incomplete, an interesting approach allows comparing the real experiments with a set of randomly generated experiments to verify the robustness of the real data⁵. In network analysis this is achieved by comparing the networks we are interested into, with some randomly generated networks.

In this sense, our app was created to address the problem of creating a validation layer which allows simulating random experiments. With randomly created networks it becomes possible to compare, by using a specific statistical test, the numerical results that come from a common network analysis. To do so, our Network Randomizer allows randomizing existing networks by using a simple shuffling algorithm and a degree preserving algorithm. Moreover it allows creating Erdős–Rényi, Barabási–Albert, Watts–Strogatz, Lattice, and Community Affiliation models. We also implemented the Multiplication model which is designed to generate weighted networks where nodes are multiplied, i.e. represented in different copies which have the same topological characteristics, to fit quantitative data. Finally a statistical module, based on the two-sample Kolmogorov–Smirnov test, compares networks in order to check whether the values come from the same distribution or if they should be considered different. The original idea behind the app was to compare topological attributes, the so called centralities, computed by using CentiScape⁶ or with the Cytoscape's built-in Network Analyzer, but it supports the comparison of all kinds of numerical attributes in order to be useful to all users.

Methods

Implementation

Network Randomizer follows a modular structure, making it easy to add additional random network models in future releases. Each model is initialized with some user's specified parameters and once the generation is done the app deploys the network into the Network Control Panel, making it instantly available to the user for future use. Since the user may want to generate a number of random networks, the network views are not created by the app, in

order to avoid pop-ups of networks in the Cytoscape window during the algorithm's computation.

The app is divided in several classes which allow a very flexible and modular structure. The CyActivator class runs the application and communicates with Cytoscape. The RandomizerCore class is used as a model of the current Cytoscape state (network handling etc.). The MenuAction class allows app initiation. The OptionsMenu class refers to the main GUI which is used to interact with the app. The ThreadEngine allows creating and handling threads and multithread tasks. Finally the AbstractModel is an abstract class which defines the basic random model and offers several methods that could be useful when defining a new model. The other classes in the app refer to the models we implemented. In order to add a new model, a new class should be instantiated by following the AbstractModel implementation. Then it is necessary to modify the GUI in order to let the new model become part of the app.

The statistical module that we implemented is a two-sample Kolmogorov–Smirnov test⁷. It takes each pair of real and random networks, and for each of the attributes used, it calculates the difference between their distribution. The K–S definition of the distribution difference is the maximum gap between the cumulative probability functions of those probabilities. Although it relies on the cumulative probability functions, they never need to be explicitly calculated. In fact, the algorithm only needs to sort the two lists of attribute's values and then run through them in parallel, summing the normalized (divided by the sum) values along the way and comparing the two sums, saving only the largest difference.

Operation

System requirements. To install and run the Network Randomizer app, the Cytoscape software must first be installed. Once this is done, no additional system requirements exist since the app does not use any external library nor does it significantly increase the memory consumption. An updated Java version is suggested, since the last Cytoscape version works only with Java 8.

It is important to note that, when using the Multiplication model, the number of nodes that are generated could result in a huge network. This is due to the model itself which, starting from an user defined attribute, creates an array of random weights which will result in a number of nodes. It works by creating a range between the minimum and the maximum values found in the file, hence in the worst case the algorithm will create a network with *#max* copies of nodes for each node. In numbers this means that if the attribute file varies in a range [0 – 100] and the real network is made of 10 nodes, then the algorithm in the worst case may generate a network with $(100 * 10) + 10$ nodes. Memory usage problems may also arise within the other models, depending on the number of nodes the user selects for creating a random network.

Workflow. The usual Network Randomizer workflow consists of a few distinct steps. First of all, the user needs to load one or more real networks. Depending on whether one or multiple

networks are loaded, the final output will be different. It consists of a file which shows the results of the statistical analysis. If there is only one real network, the output file will present more details about it. If there are multiple real networks, then the output file will be a summary of all the statistical tests that were carried out.

After loading the networks, the randomization step creates random networks. The random networks may be created either by randomizing real networks, using the edge shuffle algorithm (degree preserving or completely random) (Figure 2), or by generating new random networks by using one of available models (Figure 3 and Figure 4).

After real networks are loaded and random networks generated, it is possible to compare their shared attributes. Since the app was initially designed to be used in conjunction with centralities parameters, we recommend using the CentiScaPe app which is completely compatible with our Network Randomizer. It is important to note that the app allows comparing the attributes with the same name in all the networks that are selected within the statistical module. Basically, once the networks are selected, only the shared attributes across all networks would appear as possible

attribute choices. If there are no attributes that are shared, the user should check the names in the Node Table panel of Cytoscape.

Finally, once all the data are available, the statistical test (Figure 1) compares the selected attributes, pinpointing their differences and similarities and giving a textual file which summarizes the results. Each part of the app has a question mark button which provides help to the users, making the Network Randomizer easier to use, as well as appropriate for educational purposes.

Use cases

Data preparation. In order to use the app we present a typical use case to guide the users through the network analysis process, using the Network Randomizer.

The first step when analyzing any network with Cytoscape apps is to import the network into the Cytoscape software. After that, as said, some information is required, and even though we focused on topological centralities, it is possible to use any kind of numerical information. There are multiple ways to fill in this information. It is possible to import a .csv file, to create a new attribute or to use a built-in network analyzing function.

Figure 1. Statistical module: after data are generated, the statistical module allows comparing all the networks attributes in order to find important patterns. The attribute to be compared must have the same name in all the networks that are selected (it is also case-sensitive).

Figure 2. Randomization interface: the users can choose between a simple edge shuffle, or the degree preserving version. Both of them are intended for randomizing an already existing networks.

☒ ErdosRenyi model

☒ $G(n,p)$ $n =$ N must be greater than 0
☐ $G(n,M)$ $p =$ p must be comprised in [0-1]
 $M =$ M must be less than $(n*(n-1))/2$

☒ Watt-Strogatz model

$N =$ N must be greater than 0
 $K =$ $0 < K < (n/2)$
 $\beta =$ Beta must be comprised in [0-1]

☒ Barabasi-Albert model

$N =$ N must be greater than 0
 $m =$ $\ll N$ $0 < M < (n/2)$

Figure 3. Random network models: Erdős–Rényi, Watts–Strogatz, Barabási–Albert. They require some parameters which are inserted by the user. Some labels, in red, help the user in order to correctly fill the fields.

☒ Multiplication model

Multiply current network, using:

☐ the network is directed
☐ generate networks

☒ Lattice model

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

Dimensions must be grater than 1

☐ Generate hyper torus

☒ Community Affiliation model

Community data file:

Figure 4. Multiplication model network generator, lattice generator and Community Affiliation Graph model. The Multiplication and the Community Affiliation require a file as input, in order to generate random networks.

A more advanced approach is to use a popular centrality measures app, i.e. CentiScaPe, which can be downloaded from the Cytoscape Apps store. It provides the users (and consequently the Network Randomizer itself) with more information about the networks, and, finally, results in a much wider perspective on differences between them. It is important to note that by using these apps one obtains standard names for the attributes. Caution should be taken when using user defined attributes, that the same attribute names are used for the real and for the random networks otherwise no attribute will appear while performing the statistical analysis.

Generating random networks. There are two main methodologies for generating random networks. By using the random network models or by randomizing current, real networks. For obtaining better 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.

To randomize an existing 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. Once the network is selected, there are two randomizing methods that can be used: simple edge shuffle, and degree preserving edge shuffle. Users can choose one or both models by checking the boxes (Figure 2). Once the module is selected, the Start button runs the randomization.

Apart from the randomization option, the main part of the Randomizer consists of multiple random network models.

Currently implemented models are: Erdős–Rényi⁸, Watts–Strogatz⁹, Barabási–Albert¹⁰, lattices and Community Affiliation Graph¹¹. There is also a new model which we called the Multiplication model.

Erdős–Rényi model (Figure 3) generates fully random networks by either uniformly choosing M pairs of nodes to connect, or by connecting each pair with probability p . Watts–Strogatz model (Figure 3) generates networks which show the so called small-world phenomenon. Here, although the network is not dense, the average shortest path is still significantly low. Networks generated by using the Barabási–Albert model (Figure 3) are scale-free networks, meaning that their degree distribution follows a power law distribution. The model uses the preferential attachment growth, hence each new node is connected to m other nodes, choosing them with a probability which is proportional to their degree. Once a model is selected the users are informed about the parameter constraints by a red label (Figure 3). The labels update their values, in order to be more helpful, once they are filled in correctly and after pressing the Enter button.

Multiplication model (Figure 4) generates randomly weighted network (Figure 5). The algorithm generates a random array which defines a weight for each node belonging to an existing, user-defined network, starting from an attribute file which contains quantitative information about the nodes. Then the algorithm creates a new network which contains a number of copies that depend on the random weights that were assigned to each node. This random weighted network represents a possible experimental

Figure 5. The algorithm for the multiplication of the nodes, starting from the network on the left, creates a random array for each new random network that will be generated. By using the values in the random array a number of nodes, in green, are added to the original network, represented by the yellow nodes and some new networks are created.

setup which derives from the attribute file. In this sense it is possible to generate a number of randomly weighted networks, in order to simulate a set of experiments. The new nodes, that are added as copies of existing nodes, have the same neighbours of the original node and share an edge with the original node and each of its copies.

Lattice model (Figure 4) generates regular, multidimensional lattices. Multidimensional lattices are grids of nodes, each of which is connected only to its first neighbours. For example, the one-dimensional lattice is a path graph, while the two-dimensional lattice is a square grid. Additionally, users have the option of generating torus-shaped lattices, where there are no end-nodes. For example, the one-dimensional torus lattice is a cycle graph.

Finally, the Community Affiliation model (Figure 4) generates random networks from the community information: given a list of communities, their members, and the probability of having an edge between two members of each community, it randomly generates realistic social networks.

Some models require special input files which define their behavior. The two models that currently require an input file are *Multiplication* and *Community Affiliation*. The Multiplication file must contain numerical values, one for each row. It is expected, but not mandatory, that the number of rows is equal to the number of nodes and the Randomizer will pop up a dialog which asks the user if the number of values found in the file is correct. The Community Affiliation model requires a file which contains a set of rows where each line represents a community. The rows starts with a p-value, i.e. the probability of an edge between two members of the community, and is followed by the names of the nodes inside that community, separated by spaces.

One last point concerns the fact that, once a model is selected in the app panel, every time the button Start Randomization is pressed each selected model is created in a number of copies chosen by the user (Figure 6). If the number of networks is not specified, by using the form, the algorithm will generate a network for each selected model. Otherwise the selected number of newtworks will be generated, for each selected model. This means that, after a computation, we suggest removing the check mark from all the models in order to avoid creating a number of not required new networks, every time Network Randomizer runs.

Comparing networks. Once random networks are generated, before comparing their attributes to the real ones, at least a numerical attribute should be present in all the networks that will be

compared. To run the statistical comparison module, it is necessary to specify which networks it will use. To do this, users need to select the real networks in the Networks Control Panel and press Selected, and then do the same for the random ones. After selecting the networks, a list of all the node attributes shared within all the selected networks is provided. Users can now select the node attributes they would like to compare. If there is not a shared attribute then a dialog will appear to tell the user to check again the attributes. Selection is done using the 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, output file name and directory need to be specified, and the comparison can be executed by clicking the Start Statistics button.

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

If 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. The first field indicates how different is each random network from the real one by using the average difference across all centralities. The second field represents the difference between the real network and its most similar random network, according to each centrality measure individually. The last field provides more in-depth information, specifying the difference between the real network and each random one, for each centrality measure.

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 shows the most similar random network with respect to the real one, for each real network and each centrality, and specifies their difference. The last field is the real-random distance matrix, with distances defined as average difference across all centralities. A value of 0 indicates that the distributions are completely the same, up to a normalization factor. Normalization factor, in this sense, is the number of elements in the series. So, for example, the series [1, 2, 3] and [1, 1, 2, 2, 3, 3] would

Multiple network generation ?

☒ generate networks be careful when choosing the number of networks you're creating!

Figure 6. This form allows the user in order to choose how many networks will be generated, once the Randomizer runs, for each selected model.

be completely the same. A value which is close to 1 indicates the existence of an important difference between the distributions. This result is rarely achieved in real datasets, and it happens when the elements of one series are all greater than the elements of the other.

Summary

To summarize, our app allows generating and creating random networks and is useful whenever a validation benchmark is required. Starting from real networks it is possible to compare their attributes with randomly generated attributes obtained by analyzing random networks. The main issue concerns the fact that there is not a specific model that should be suggested in a specific setup or with some specific kinds of data. It is up to the user to select which model best describes the network and its mathematical characteristics.

Concluding, our app is designed to be as general as possible having a very wide range of applications and to be completely user-friendly, giving the possibility to perform a simple, but meaningful, statistical analysis and a readable output.

Software availability

1. <http://apps.cytoscape.org/apps/networkrandomizer>
2. Source code as at the time of publication: <https://github.com/gabrielet/Network-Randomizer> authors Gabriele Tosadori and Ivan Bestvina

3. [10.5281/zenodo.159271](https://doi.org/10.5281/zenodo.159271)¹²

4. Software license: Apache License, Version 2.0

Author contributions

GT defined and implemented the random multiplication model, designed and developed the app, and wrote the manuscript, IB designed and developed the app, and wrote the manuscript. Finally GS, FS and CL tested the app and gave interesting comments that improved our work.

Competing interests

No competing interests were disclosed.

Grant information

This work was supported by: Italian Association for Cancer Research (AIRC, IG16797), MIUR (PRIN 2009), Nanomedicine project University of Verona and Fondazione Cariverona to Carlo Laudanna.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Acknowledgments

We would like to acknowledge the National Resource for Network Biology (NRNB) Training Program which allowed the actual development of the app.

References

1. Saito R, Smoot ME, Ono K, *et al.*: **A travel guide to Cytoscape plugins.** *Nat Methods*. 2012; **9**(11): 1069–1076.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
2. Cline MS, Smoot M, Cerami E, *et al.*: **Integration of biological networks and gene expression data using Cytoscape.** *Nat Protoc*. 2007; **2**(10): 2366–2382.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
3. Sah P, Singh LO, Clauset A, *et al.*: **Exploring community structure in biological networks with random graphs.** *BMC Bioinformatics*. 2014; **15**(1): 220.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
4. Halbe-Kains B, Emmert-Streib F: **Quantitative assessment and validation of network inference methods in bioinformatics.** *Frontiers Media SA*, 2015.
[Publisher Full Text](#)
5. Newman ME, Watts DJ, Strogatz SH: **Random graph models of social networks.** *Proc Natl Acad Sci U S A*. 2002; **99**(suppl 1): 2566–2572.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
6. Scardoni G, Petterlini M, Laudanna C: **Analyzing biological network parameters with CentiScaPe.** *Bioinformatics*. 2009; **25**(21): 2857–2859.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
7. Wilcoxon RR: **Some practical reasons for reconsidering the Kolmogorov-Smirnov test.** *Br J Math Stat Psychol*. 1997; **50**(1): 9–20.
[Publisher Full Text](#)
8. Erdős P, Rényi A: **On Random Graphs I.** *Publ Math Debrecen*. 1959; **6**: 290–297.
[Reference Source](#)
9. Watts DJ, Strogatz SH: **Collective dynamics of ‘small-world’ networks.** *Nature*. 1998; **393**(6684): 440–442.
[PubMed Abstract](#) | [Publisher Full Text](#)
10. Barabasi AL, Albert R: **Emergence of scaling in random networks.** *Science*. 1999; **286**(5439): 509–512.
[PubMed Abstract](#) | [Publisher Full Text](#)
11. Yang J, Leskovec J: **Community-affiliation graph model for overlapping network community detection.** In *Data Mining (ICDM), 2012 IEEE 12th International Conference on IEEE*. 2012; 1170–1175.
[Publisher Full Text](#)
12. Gabriele T, Bestvina I: **F1000Research/Network-Randomizer: F1000Research/Network-Randomizer [Data set].** *Zenodo*. 2016.
[Data Source](#)

Open Peer Review

Current Peer Review Status:

Version 1

Reviewer Report 01 November 2016

<https://doi.org/10.5256/f1000research.9906.r17030>

© 2016 Russo F. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Francesco Russo

The Novo Nordisk Foundation Center for Protein Research, University of Copenhagen.,
Copenhagen, Denmark

The authors proposed a new Cytoscape App to generate random networks and compare them with real data. It consists of the most common approaches for network randomisation. Moreover, the authors proposed a novel method to generate random weighted networks.

There are few implemented apps available in Cytoscape for this purpose, so this app will be very useful and simple to use in a well known framework.

The manuscript is well written and clear. It could be nice and useful to have a figure representing the workflow of the proposed app, if it is compatible with the number of figures allowed for this journal.

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Reviewer Report 31 October 2016

<https://doi.org/10.5256/f1000research.9906.r17046>

© 2016 Micale G. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Giovanni Micale

Department of Math and Computer Science, University of Catania, Catania, Italy

In the paper, Tosadori et al. describe a Cytoscape app for creating random networks according to different random models and comparing them with one or more input networks on the basis of different topological metrics.

The main goal of the app is to help the user in the creation of a benchmark for validating some properties of a real network or a dataset of real networks.

I found the app very interesting and I think that the paper is suitable for being indexed. However, there are several major issues, especially regarding the potential applications of the tool and the statistical part of the app, that need to be addressed before indexing.

1. The actual usefulness of the app is not fully explained. In the Abstract and in the Introduction the authors point out that the app should serve as a framework to validate some numerical results of one or more input networks, but it is not clear how the results of comparing values of topological attributes of real and random networks can be exploited by the user to draw conclusions. I would suggest to insert (for example in the Introduction) one or more examples of usage of this tool. For instance, an interesting application of the tool could be the following: generate different random networks (one or more for each random model) with a similar number of nodes and edges of the real network, then compare these networks with the real one according to some statistics (for example clustering coefficient) in order to find the random model that best fits the input network. This can tell a lot about the features of the network and how it has been generated.
2. The statistical module of the app allows to compare two distribution of values on some topological property of the nodes (for example centrality) by using the two-sample Kolmogorov-Smirnov test. The aim of this comparison should be carefully clarified. Is the aim to establish if topological properties are maintained with respect to the random model used? Does it make sense when such a value is obtained comparing only a pair of networks, the real and the random ones?

Let's consider a set of N networks generated using the preferential attachment and compare them to a real network according to the values of some metric M . The reviewer expects that will be sensible variations on the Kolmogorov-Smirnov distance. It would be more reasonable to give the user the possibility to compare expected values of some metric in ensembles of real and random networks. In some context, this kind of analysis can be preferable and would result more robust.

3. Among the random models, the authors presented a new model called Multiplicative Model. It is not clear from the text how such a model works. There is a picture in the paper that shows three networks generated with three different weighted arrays, but it is not clear how these weighted arrays are generated. Which are the input parameters of the model? Please clarify how the weights are determined. It is not clear i) if the user has to provide a minimum value X and a maximum value Y and then weighted arrays are generated by randomly picking one value within interval $[X,Y]$ for each node or ii) if just has to provide a text file with one weight for each node and generate a random network with exactly that sequence of weights. By using the app, it seems that a text file is required but it is not clear what should be its content. In general, authors should describe more clearly the model both in the paper and in the documentation of the app.

4. It looks like the Multiplication model is not properly working: I tried to generate a random network using the Multiplication Model starting from an input network of N nodes. As a parameter for the model, I loaded an input file with N rows and one number for each row (in each row there is 0 or 1 as weight). A window appeared saying which was the maximum number of edges admitted for the network. However, at the end of the process, a random network with exactly the same number of nodes and edges (maybe a copy) of the input network was created without any error or warning message. I expected as output a network with a different number of nodes and edges. Please clarify this point.
5. In the Abstract and in the Introduction authors claim that there are no tools for generating and randomizing networks. Actually, there are two Cytoscape apps that authors should cite in the paper: Randomnetworks plugin (<http://apps.cytoscape.org/apps/randomnetworks>) and NetMatch* (<http://alpha.dmi.unict.it/netmatchstar/netmatchstar.html>). Furthermore, since your app implements a comparison part between networks, I would also recommend to cite some other Cytoscape app to compare networks, such as GASOLINE (<http://apps.cytoscape.org/apps/gasoline>) and DyNet (<http://apps.cytoscape.org/apps/dynet>), remarking the different kind of approach to compare networks that you are following, which is the new research contribution.
6. On page 6, third paragraph, when authors talk about the two shuffling methods (with or without preservation of degrees) they say that users can choose one or both models. Authors should describe which is the effect when both options are selected.
7. What is the maximum number of nodes in a network generated with the multiplication model? In the example at page 3 for a network of 10 nodes and weights in [0,100] is $100 \times 10 + 10$, but in the figure of page 6 it seems to be just the number of nodes multiplied by the maximum value of the range (in the figure of page 6 it seems 7×3 and not $7 \times 3 + 7$). Please clarify this point.
8. In the paper there is no indication of which sizes represent the Lattice Model (Figure 4). I finally deduced this information from the documentation. This should be reported with a sentence also in the main paper.
9. There is a difference between the panel of Multiplication Model depicted in Figure 4 and the Multiplication Model panel present in the Cytoscape App. In the app I found an option called "Graphical version" which is not reported in the paper. Please clarify and make the app and the paper coherent on such a point.
10. When the user generates random networks or run the statistical analysis there is no indication on the status of the computation. It would be recommended to notify the user with logs and progress bars.

11. I tried to execute a pipeline of statistical experiments with different input networks and/or random networks. However, after the first experiment, it was impossible to me to change the set of selected input or random networks for a second experiment. I had to close and open again the app to make other tests. Please fix this bug.
12. In the Statistical analysis panel there is a "Save as" button to select the path where to save output results. However, there is no text field next to the button that indicates the selected output folder. Please insert such a field.

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 12 Jun 2017

Gabriele Tosadori, University of Verona, Verona, Italy

Dear Giovanni,

Thank you for all the very useful comments. We amended the paper as you suggested and we hope it is now more clear and useful.

I would like to answer you with respect to the points 10, 11 and 12.

Point 10: we decided not to add the progress bar since the computation takes a very little time and we believe it is not a necessary feature. If you really think that it is needed, then we will add it for our newest release otherwise, we'll keep the app as it currently is

Point 11: we are not able to obtain the same bug you're mentioning. Please give us more feedback about this issue and we will see how to fix this.

Point 12: we decided not to add the path to the folder since it was creating a lot of issue with the Mac version of the Java library. In order to avoid bugs for the Mac users we removed this feature.

If you have further questions and suggestions, we will be very happy to hear everything.

Thank you,

Gabriele Tosadori

Competing Interests: No competing interests were disclosed.

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research