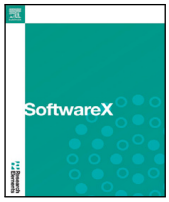
SoftwareX 26 (2024) 101647

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx



Original software publication

GIN: A web-application for constructing synthetic datasets of interconnected networks in bioinformatics

Pietro Cinaglia

Department of Health Sciences, Magna Graecia University of Catanzaro, 88100 Catanzaro, Italy

ARTICLE INFO

Dataset link: https://github.com/pietrocinaglia/gin, https://pietrocinaglia.github.io/gin

Keywords:
Interconnected networks
Synthetic dataset
Network graph
Bioinformatics

ABSTRACT

Networks are crucial in structuring biological systems in terms of interactions, to investigate the related properties and dynamics. Network analysis plays a relevant role in knowledge extraction from complex biological data, where biological objects and their own interactions (or relationships) can be modelled as nodes and edges, respectively. In this context, data is crucial in testing and evaluating process, for designing effective methods and software tools. For instance, network alignment requires many samples to evaluate the performance in node mappings, as well as in contexts where the aim is to reconstruct a missing or perturbed topology. It is common in scientific literature to report a lack of datasets related to interconnected biological networks. Usually, a research group has to design its own dataset for testing a novel method or software tool. This work addressed this issue, presenting a web-based Generator of synthetic Interconnected Networks (*GIN*). Our contribution has the primary aim of supporting the provision of synthetic data, via a user-friendly and effective tool. *GIN* allows for the generation of ad-hoc datasets that are particularly suitable for bioinformatics applications. Furthermore, the possibility of defining the network generation parameters allows a scientist to model (ad-hoc) interconnected networks for reproducing the topological and dynamical properties of a real biological network of interest.

Code metadata

Current code version	1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00763
Permanent link to Reproducible Capsule	
Legal Code License	MIT
Code versioning system used	GIT
Software code languages, tools, and services used	Python, HTML5, CSS, JavaScript
Compilation requirements, operating environments & dependencies	https://github.com/pietrocinaglia/gin/blob/main/src/requirements.txt
If available Link to developer documentation/manual	https://github.com/pietrocinaglia/gin/blob/main/README.md
Support email for questions	cinaglia@unicz.it

Software metadata

Current software version	1.0
Permanent link to executables of this version	https://pietrocinaglia.github.io/gin
Permanent link to Reproducible Capsule	
Legal Software License	
Computing platforms/Operating Systems	Web Application, Platform-independent
Installation requirements & dependencies	
If available, link to user manual - if formally published include a reference to the publication in the reference list	
Support email for questions	cinaglia@unicz.it

E-mail address: cinaglia@unicz.it.

https://doi.org/10.1016/j.softx.2024.101647
Received 10 November 2023; Received in revised form 22 January 2024; Accepted 28 January 2024
Available online 6 February 2024

1. Introduction

Network graph models (or networks) provide a holistic view of the system's structure and dynamics, aiding in studying complex biological phenomena to infer, e.g., novel features or behaviour [1–3].

A network mainly consists of nodes and edges which represent the objects and their own relationships (e.g., interactions), respectively. Basically, it supports a single type of object (i.e., entity), and it develops on a single layer. However, multiple layers can be modelled by using a multilayer model (i.e., multilayer network), whose meaning depends on whether the set of objects belong to the same entity (i.e., homogeneous) or different ones (i.e., heterogeneous). Specifically, the former are called multiplex networks, while the latter are called interconnected networks [4].

In this paper, we focus on interconnected networks, which prove highly effective in scenarios where objects of different entities require a heterogeneity of types of interactions [5,6]. These networks allow for modelling interactions from several biological entities (e.g., genes, proteins, drugs), for purposes such as investigating these in terms of relationships, and before for integrating and unifying different networks to explore insights from multiple sources. To give a non-exhaustive example, a scientist may exploit the advantages provided by network science to focus their experimental investigations on a species genetically simpler than *Homo sapiens*, and then transfer the progress resulting from the mining of knowledge, to the latter.

Essentially, we can study the network of a simpler species and then transfer the knowledge to the network of the more complex one. The latter methodology is called Network Alignment (NA), and it is a well-known topic of network science [7].

NA allows for the mapping of nodes that belong to the same entity from different networks. It may be implemented using several approaches of varying effectiveness and efficiency from the literature. The type of approach and its implementation largely depends on the type of structure and topology of the network [8]. For instance, *Dante* [9] is a method for the pairwise alignment of temporal networks, *FINAL* [10] is able to extract knowledge from attributed networks for improving alignment performance, while *Grad-Align* and *Grad-Align+* [11,12] empower the gradual NA via attribute augmentation. Another fundamental factor is the number of networks involved in NA; for instance, [13] is able to consider multiple networks, thus differing from pairwise alignment. Referring to interconnected networks and more generally to multilayer networks, these can be aligned using *MultiGlobAl* [14] or *DANTEml* [15], both based on pairwise global approach for NA.

There is a clear lack of real datasets of interconnected networks. This issue is reflected in the need for generating ad-hoc datasets. Furthermore, several datasets only consist of static networks [16], and no resources support interconnected networks, especially in biology.

We focused on the bioinformatics field, in order to address the gap in availability of interconnected networks for testing and evaluating methods and software tools. To compensate, synthetic data is often used for experiments; however, to date, no generator is available for interconnected networks.

In this paper, we present a web-based Generator of synthetic Interconnected Networks (*GIN*).

This tool supports the provisioning of synthetic data via a user-friendly and effective tool, aimed at those who need data to be able to test the performance of their methodology. The degree of freedom offered in the parameterization of the network allows the user to be able to model multiple models. In addition, our solution allows the generation of noise, which is particularly important in NA, as well as in contexts where the aim is to reconstruct a missing or perturbed topology [17].

We summarize the key-points related to the main features of *GIN* as follows:

- Generating interconnected network;
- Creation of custom interconnected networks, in order to reproduce given behaviours (e.g. evolutions or associations) of interest.
- Building datasets based on an initial interconnected network and its counterparts for a specific noise level.
- Providing a ready-to-use software tool for users with no expertise in bioinformatics.

2. Methods

In this section, we describe the software and its own implementation details.

GIN generates synthetic data through a random model graph construction, which uses preferential attachment based on Barabási–Albert (BA) model [18].

BA model provides a valuable framework for simulating the growth of complex networks, particularly those exhibiting scale-free characteristics. Briefly, it generates networks that adhere to a power-law distribution, meaning that a few nodes have a significantly higher degree compared to most nodes. BA model is widely used in several fields, including biology and information technology. The key concept behind the BA model is the principle of preferential attachment, which reflects the tendency of nodes to connect to existing nodes, based on their current degree of connectivity. It starts with a small seed network, typically comprising a few nodes connected in a simple configuration. This initial network serves as the foundation for subsequent growth. As the network evolves, new nodes are added one at a time with their own links, that are randomly included proportionally to the number of connections of the node of interest. The growth of the network occurs iteratively, with each new node establishing connections based on preferential attachment.

The model's ability to capture the emergence of hubs and scale-free behaviour makes it a valuable tool for exploring the dynamics of complex systems.

In the BA model, the generative parameters are constant and well-defined by the model itself, therefore, we used an extended implementation of the BA model (i.e., extended BA model), which allows the possibility of being able to customize these. Barabási et al. [19] demonstrated the validity of extended BA model for simulating real biological networks, embarking on an integrated experimental study to define a model able to reproduce the topological and dynamical properties of the various networks that control the behaviour of the biological systems (e.g., molecules and their interactions within a living cell). The model allows for the simulation of different biological scenarios based on the parameters given as input by the operator, and for which numerous studies are available [20–22].

Specifically, the extended BA model allows for the generation of random scale-free networks by applying a preferential attachment mechanism which evolves in accordance with two main coexisting parameters: the first one grows the network by adding new nodes, the second one concerns the preferential attachments for handling network's density. Through this parameterization, it is possible to make the model converge to typical behaviours of biological objects (e.g. genes, proteins, drugs) [23].

2.1. Interconnected network generation

According to the extended BA model, *GIN* allows to define the probabilities on the basis of which new edges, rewired edges, and new nodes take part in the construction of each layer, belonging to the resulting interconnected network. The parameters are described below:

- n is the number of nodes;
- p is the probability that m new edges are added ($p + q < 1$);
- q is the probability that m existing edges are rewired to a random chosen edge and its related node ($p + q < 1$);

- m is the number of (intralayer) edges with which a new node attaches to existing nodes, based on the probability $1 - p - q$;
- l is the number of layers;
- z is the percentage of (interlayer) edges that are randomly created between two layers.

The amount of interlayer edges determined by z is 0% for the one-layer case (i.e., $l = 1$), as the network must have at least 2 layers for these to exist.

Our implementation consists of two main functions, each designed for constructing different types of networks.

The first generates a one-layer network, a particular case in which interlayers do not exist:

- it creates a BA graph with specified parameters and adds it to the graph G;
- it generates a seed file containing node information;
- it writes the edge list of the network to a text file and returns the generated network and log.

The second generates a multi-layer network, including interlayers. Specifically, it builds the layers as static networks, individually. Subsequently, it randomly infers interlayer edges in accordance to z, being careful to change the pairs of nodes through successive and non-redundant selections. Finally, it stores the resulting interconnected networks as an edge list, including the features of interlayer and intralayer edges.

Note that the interlayer edges are missing for $l = 1$, and the value of z becomes irrelevant and set to $null$.

Furthermore, *GIN* provides a list of seeds, defined as pairs of nodes belonging to the initial topology and the related counterpart into noisy ones, respectively. For instance, it can be useful as a starting point for computation by an algorithm based on a local NA method [24].

2.2. Noising

We implemented our own *shuffling* function to introduce noise into the given interconnected network, in order to complete the dataset.

The amount of noise to introduce is defined through a floating-point number, given as input by the user. Note that multiple floating-point values can be provided, and in this case the noising process is repeated.

We describe the implementation of the *shuffling* function by reporting its workflow as follows:

Preprocessing: it identifies and extracts intralayer edges and stores them in a data structure (which we will call E, for explanatory purposes).

Processing: it extracts a random pair of intralayer edges from E: if both edges belong to the same layer, it swaps their endpoints by maintaining the related attribute (i.e., *shuffle*), otherwise, it extracts another random pair. Note that exchanges cannot (and must not) take place between different layers, as each of these is made up of well-defined entity types.

The processing continues until the desired noise level is achieved.

2.3. Packaging

GIN provides the interconnected network and its own noisy versions as a zipped (compressed) file, with a size in the order of tens of kilobits. The resulting package also contains some information formatted through the JavaScript Object Notation (JSON). The latter also reports both the main features of the network (e.g., number of nodes, edges, and layers), as well as information about the noising process (e.g., number of shuffles, and pairs of nodes involved in these). In addition, it includes well-known measures computed between the interconnected networks and its own noisy versions.

We considered three main measures to evaluate between the initial interconnected network and each of its noisy counterparts: *difference* in terms of edges, Graph Edit Distance (*GED*), and number of *shuffles* that affected the initial topology.

Let us define the initial interconnected network as G_1, and a generic its own noisy version as G_2, the mentioned measures can be described as follows:

- *difference*: the total number of intralayer and interlayer edges that differ between G_1 and G_2.
- *GED*: a well-known graph similarity measure [25], which is defined as the minimum cost (or amount of distortions afflicting nodes and edges) that has to be applied to topologically transform G_1 into G_2.
- *shuffles*: amount of shuffles performed to obtain G_2, in accordance with the noise level of interest; this is an input parameter.

2.4. Implementation

GIN consists of a frontend, which collects parameters of interest and sends the request to the backend. The latter processes the output, by providing it to the user as a single package (refer to the previous sections for details).

The frontend has implemented by using the following technologies: *HTML5* (https://w3.org/html5, accessed on 11 January 2024), *jQuery* (https://jquery.com, accessed on 11 January 2024), and *Bootstrap* (https://getbootstrap.com, accessed on 11 January 2024).

Furthermore, it is designed as a succession of three simple steps: (i) in the first one the user defines the name of the dataset, which will coincide with the package's name provided in output; (ii) in the second one the user provides the aforementioned parameters for constructing the interconnected network (n, p, q, m, l, and z); (iii) in the third one the user defines the set of noise levels to be applied to the network, in order to also generate the related noisy counterparts.

The backend has been implemented in Python, by using Flask (https://flask.palletsprojects.com, accessed on 11 January 2024), a well-known lightweight Web Server Gateway Interface (WSGI) web application framework. The extended BA model is implemented through *NetworkX* [26], a well-known python library. Otherwise, the functions described in the previous sections are self-implemented.

Finally, Fig. 1 shows the preview of *GIN*, through its step-by-step workflow.

3. Illustrative examples

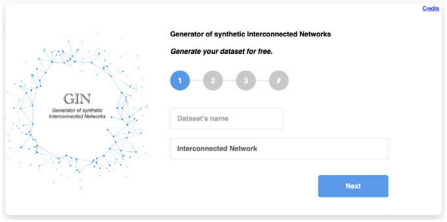
In our experimentation, we generated 10 networks for 4 noise levels (5%, 10%, 15%, and 20%), for an increasing number of layers (1, 2, 3, and 4). Therefore, we analysed the resulting datasets, consisting of 160 interconnected networks.

Figs. 2(a), 2(b), 2(c) show the plots from the descriptive analysis of our experimental results; in all cases the coefficient of variation between the noisy counterparts of the same noise level was negligible. The latter demonstrates how our solution produces noise levels perfectly traceable to the respective input parameter.

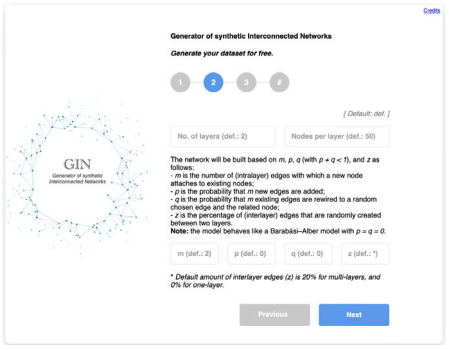
In addition, we performed the statistical assessment of our experimental results, in order to investigate their validity and significance. It shows a statistical significance that corroborates both the consistency in the generation of the networks with the same parameters, and the maintenance of adequate levels, that do not exceed the defined parameterization, as regards the noisy counterparts; the latter deviates from the initial ones in accordance with the noise levels of interest, without creating inconsistencies and indeterminacies in the topology.

Tables 1, 3, and 5 show the One-Way ANOVA tests for *difference*, *GED*, and *shuffles*, respectively; the related descriptives were shown in Tables 2, 4, and 6, respectively.

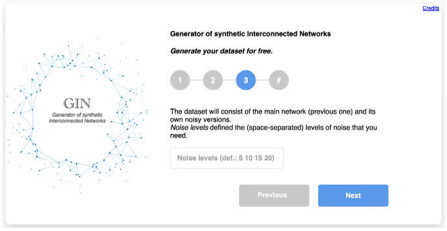
What was previously stated can therefore be checked by considering the related p (or p-value), for each measure; by conventional criteria, the latter is significant for a value less than 0.05 ($p < 0.05$).



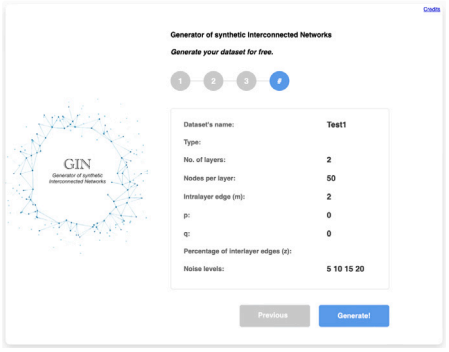
(a) Step 1: initialization; the type is pre-defined.



(b) Step 2: parameterization referring to each individual layer.

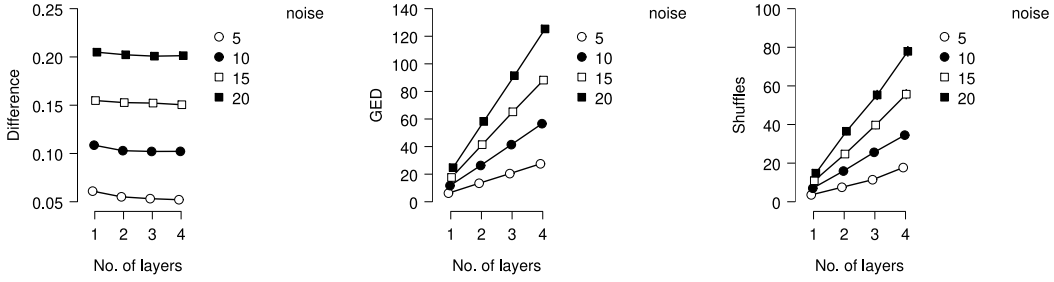


(c) Step 3: noise levels to be applied to the first interconnected network, in order to generate other ones.



(d) Step 4: summary and generation.

Fig. 1. Preview of *GIN* based on its step-by-step workflow.



(a) Difference (b) Graph Edit Distance (GED) (c) Shuffles

Fig. 2. Plots from the descriptive analysis of all GIN-generated datasets.

Table 1
Difference: One-Way ANOVA test.

Cases	Sum of Squares	DF	Mean Square	F	p
layers	8.097×10^{-4}	3	2.699×10^{-4}	53.365	< .001
noise	0.473	3	0.158	31 196.118	< .001
layers * noise	1.292×10^{-4}	9	1.436×10^{-5}	2.838	0.004
Residuals	7.232×10^{-4}	143	5.057×10^{-6}		

Note. DF: Degrees of Freedom.

4. Impact

Interconnected networks allow representing multiple semantic levels consisting of nodes and edges distributed over multiple layers, in order to model real biological systems. In bioinformatics, the methodologies and techniques related to this topic are growing. However, the state-of-the-art presents a relevant gap in tools for testing and evaluation, in terms of data availability, especially. Testing and evaluation are crucial for performance assessment, as well as to demonstrate the effectiveness of a tool or method. In this context, we presented a novel web-based tool for constructing synthetic interconnected network datasets for bioinformatics purposes, mainly. *GIN* allows for the handling of both modelling and generation via a user-friendly graphical interface, regardless of the operator's IT skills, to also be exploited by users with skills purely focused on clinical and biological fields. For instance, its alpha version has been used for constructing the datasets in the experimentation of *MultiGlobAl*. Similarly, other studies could have applied it, we mention some related to the field of network alignment, such as: *MuLaN* [27], *MultiNode2Vec* [28], as well as in inferring

Table 2
Difference: Descriptives.

Layers	Noise	N	Mean	SD	SE	Coefficient of variation
1	10	10	0.109	0.004	0.001	0.036
	15	10	0.155	0.004	0.001	0.027
	20	10	0.205	0.003	0.001	0.017
	5	9	0.061	0.004	0.001	0.065
2	10	10	0.103	0.001	4.583×10^{-4}	0.014
	15	10	0.153	0.001	4.485×10^{-4}	0.009
	20	10	0.202	0.002	5.121×10^{-4}	0.008
	5	10	0.055	0.003	8.492×10^{-4}	0.049
3	10	10	0.102	0.001	4.583×10^{-4}	0.014
	15	10	0.152	4.216×10^{-4}	1.333×10^{-4}	0.003
	20	10	0.201	0.001	3.266×10^{-4}	0.005
	5	10	0.053	0.002	4.899×10^{-4}	0.029
4	10	10	0.102	0.001	3.266×10^{-4}	0.010
	15	10	0.150	8.498×10^{-4}	2.687×10^{-4}	0.006
	20	10	0.201	6.749×10^{-4}	2.134×10^{-4}	0.003
	5	10	0.052	9.944×10^{-4}	3.145×10^{-4}	0.019

Note. N: number of samples. SE: Standard Error. SD: Standard Deviation.

Table 3
Graph Edit Distance (GED): One-Way ANOVA test.

Cases	Sum of Squares	DF	Mean Square	F	p
layers	77 298.474	3	25 766.158	91 986.256	< .001
noise	73 996.811	3	24 665.604	88 057.232	< .001
layers * noise	18 964.686	9	2107.187	7522.746	< .001
Residuals	40.056	143	0.280		

Note. DF: Degrees of Freedom.

Table 4
Graph Edit Distance (GED): Descriptives.

Layers	Noise	N	Mean	SD	SE	Coefficient of variation
1	10	10	11.700	0.483	0.153	0.041
	15	10	17.600	0.516	0.163	0.029
	20	10	24.700	0.483	0.153	0.020
	5	9	6.222	0.441	0.147	0.071
2	10	10	26.300	0.483	0.153	0.018
	15	10	41.400	0.516	0.163	0.012
	20	10	58.200	0.632	0.200	0.011
	5	10	13.400	0.699	0.221	0.052
3	10	10	41.400	0.516	0.163	0.012
	15	10	65.200	0.422	0.133	0.006
	20	10	91.400	0.516	0.163	0.006
	5	10	20.400	0.516	0.163	0.025
4	10	10	56.600	0.516	0.163	0.009
	15	10	88.200	0.422	0.133	0.005
	20	10	125.300	0.675	0.213	0.005
	5	10	27.500	0.527	0.167	0.019

Note. N: number of samples. SE: Standard Error. SD: Standard Deviation.

Table 5
Shuffles: One-Way ANOVA test.

Cases	Sum of Squares	DF	Mean Square	F	p
layers	30 414.894	3	10 138.298	2318.447	< .001
noise	28 666.444	3	9555.481	2185.168	< .001
layers * noise	7262.733	9	806.970	184.540	< .001
Residuals	625.322	143	4.373		

Note. DF: Degrees of Freedom.

biological associations having multiple instances over interconnected layers (e.g., gene–disease associations) [29]. Finally, we expect its use in research fields related to network science, and bioinformatics or biology, e.g., focused on biological objects mapping between different species, for knowledge transfer.

5. Conclusions

Scientific literature frequently reports a lack of datasets related to interconnected biological networks. Usually, a research group designs its own dataset for testing a new method or software tool for network analysis (e.g., network alignment). The proposed solution helps solve this issue. We investigated this topic, by designing a web-based Generator of synthetic Interconnected Networks (*GIN*), which may address this need.

Our contribution has the primary aim of supporting the provision of synthetic data, via a user-friendly and effective tool which allows generating synthetic datasets. This is particularly suitable for bioinformatics applications that need to evaluate their performance.

Our software is available both as source-code, and as a ready-to-use web-application for demonstration purposes only.

Table 6
Shuffles: Descriptives.

Layers	Noise	N	Mean	SD	SE	Coefficient of variation
1	10	10	7.000	0.816	0.258	0.117
	15	10	10.900	1.370	0.433	0.126
	20	10	14.800	1.549	0.490	0.105
	5	9	3.556	0.726	0.242	0.204
2	10	10	15.900	0.994	0.314	0.063
	15	10	24.700	1.636	0.517	0.066
	20	10	36.500	2.461	0.778	0.067
	5	10	7.500	0.707	0.224	0.094
3	10	10	25.600	2.271	0.718	0.089
	15	10	39.700	2.627	0.831	0.066
	20	10	55.300	3.335	1.055	0.060
	5	10	11.400	0.966	0.306	0.085
4	10	10	34.500	2.550	0.806	0.074
	15	10	55.700	3.268	1.033	0.059
	20	10	77.900	3.414	1.080	0.044
	5	10	17.700	0.949	0.300	0.054

Note. N: number of samples. SE: Standard Error. SD: Standard Deviation.

Future works may concern the design of an environment more similar to a standardized testing-space for the execution and evaluation of methods related to network analysis, which comes alongside ad-hoc dataset building.

CRediT authorship contribution statement

Pietro Cinaglia: Writing – review & editing, Writing – original draft, Visualization, Supervision, Software, Methodology, Investigation, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

GIN's source code and supplementary data are available at https://github.com/pietrocinaglia/gin, under MIT License. In addition, GIN is freely accessible as ready-to-use web-application at https://pietrocinaglia.github.io/gin.

Acknowledgements

This work was funded by the Next Generation EU - Italian NRRP, Mission 4, Component 2, Investment 1.5, call for the creation and strengthening of 'Innovation Ecosystems', building 'Territorial R&D Leaders' (Directorial Decree n. 2021/3277) - project Tech4You - Technologies for climate change adaptation and quality of life improvement, n. ECS0000009. This work reflects only the authors' views and opinions, neither the Ministry for University and Research nor the European Commission can be considered responsible for them.

References

[1] Guzzi PH, Roy S. Complex network models. In: Guzzi PH, Roy S, editors. Biological network analysis. Academic Press; 2020, p. 53–75. http://dx.doi.org/10.1016/B978-0-12-819350-1.00010-4.

[2] Ma'ayan A. Introduction to network analysis in systems biology. Sci Signal 2011;4(190):tr5.

[3] Charitou T, Bryan K, Lynn DJ. Using biological networks to integrate, visualize and analyze genomics data. Genet Sel Evol 2016;48(1):27.

[4] Hammoud Z, Kramer F. Multilayer networks: aspects, implementations, and application in biomedicine. Big Data Analytics 2020;5(1). http://dx.doi.org/10.1186/s41044-020-00046-0.

[5] Barrenäs F, Chavali S, Alves AC, Coin L, Jarvelin M-R, Jörnsten R, et al. Highly interconnected genes in disease-specific networks are enriched for disease-associated polymorphisms. Genome Biol 2012;13(6):R46.

[6] Lv Y, Huang S, Zhang T, Gao B. Application of multilayer network models in bioinformatics. Front Genet 2021;12:664860.

[7] Zhang S, Tong H. Network alignment: Recent advances and future directions. In: Proceedings of the 29th ACM international conference on information & knowledge management. New York, NY, USA: Association for Computing Machinery; 2020, p. 3521–2. http://dx.doi.org/10.1145/3340531.3412168.

[8] Cinaglia P, Cannataro M. Network alignment and motif discovery in dynamic networks. Netw Model Anal Health Inf Bioinf 2022;11. http://dx.doi.org/10.1007/s13721-022-00383-1.

[9] Cinaglia P, Cannataro M. A method based on temporal embedding for the pairwise alignment of dynamic networks. Entropy 2023;25(4). http://dx.doi.org/10.3390/e25040665, URL https://www.mdpi.com/1099-4300/25/4/665.

[10] Zhang S, Tong H. FINAL: Fast attributed network alignment. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. KDD '16, New York, NY, USA: Association for Computing Machinery; 2016, p. 1345–54. http://dx.doi.org/10.1145/2939672.2939766.

[11] Park J-D, Tran C, Shin W-Y, Cao X. GradAlign+: Empowering gradual network alignment using attribute augmentation. In: Proceedings of the 31st ACM international conference on information & knowledge management. CIKM '22, New York, NY, USA: Association for Computing Machinery; 2022, p. 4374–8. http://dx.doi.org/10.1145/3511808.3557605.

[12] Park J, Tran C, Shin W, Cao X. On the power of gradual network alignment using dual-perception similarities. IEEE Trans Pattern Anal Mach Intell 2023;45(12):15292–307. http://dx.doi.org/10.1109/TPAMI.2023.3300877.

[13] Heimann M, Shen H, Safavi T, Koutra D. REGAL: Representation learning-based graph alignment. In: Proceedings of the 27th ACM international conference on information and knowledge management. CIKM '18, New York, NY, USA: Association for Computing Machinery; 2018, p. 117–26. http://dx.doi.org/10.1145/3269206.3271788.

[14] Cinaglia P, Cannataro M. MultiGlobAl: Global alignment of multilayer networks. SoftwareX 2023;24:101552. http://dx.doi.org/10.1016/j.softx.2023.101552.

[15] Cinaglia P, Milano M, Cannataro M. Multilayer network alignment based on topological assessment via embeddings. BMC Bioinformatics 2023;24(1):416.

[16] Zitnik M, Sosič R, Maheshwari S, Leskovec J. BioSNAP Datasets: Stanford biomedical network dataset collection. 2018, http://snap.stanford.edu/biodata.

[17] Chow K, Sarkar A, Elhesha R, Cinaglia P, Ay A, Kahveci T. ANCA: Alignment-based network construction algorithm. IEEE/ACM Trans Comput Biol Bioinform 2021;18(2):512–24. http://dx.doi.org/10.1109/tcbb.2019.2923620.

[18] Albert R, Barabási A-L. Topology of evolving networks: Local events and universality. Phys Rev Lett 2000;85:5234–7. http://dx.doi.org/10.1103/PhysRevLett.85.5234, URL https://link.aps.org/doi/10.1103/PhysRevLett.85.5234.

[19] Barabási AL, Oltvai ZN. Network biology: understanding the cell's functional organization. Nat Rev Genet 2004;5(2):101–13.

[20] Assenov Y, Ramírez F, Schelhorn S-E, Lengauer T, Albrecht M. Computing topological parameters of biological networks. Bioinformatics 2007;24(2):282–4. http://dx.doi.org/10.1093/bioinformatics/btm554.

[21] Krishnan J, Torabi R, Schuppert A, Napoli ED. A modified ising model of Barabási-Albert network with gene-type spins. J Math Biol 2020;81(3):769–98.

[22] Santolini M, Barabási A-L. Predicting perturbation patterns from the topology of biological networks. Proc Natl Acad Sci USA 2018;115(27):E6375–83.

[23] Zhong Y, Li J, He J, Gao Y, Liu J, Wang J, et al. Twadn: an efficient alignment algorithm based on time warping for pairwise dynamic networks. BMC Bioinformatics 2020;21(Suppl 13):385.

[24] Milano M, Cinaglia P, Guzzi PH, Cannataro M. Aligning cross-species interactomes for studying complex and chronic diseases. Life 2023;13(7). http://dx.doi.org/10.3390/life13071520.

[25] Abu-Aisheh Z, Raveaux R, Ramel J-Y, Martineau P. An exact graph edit distance algorithm for solving pattern recognition problems. In: Proceedings of the international conference on pattern recognition applications and methods - volume 1. ICPRAM 2015, Setubal, PRT: SCITEPRESS - Science and Technology Publications, Lda; 2015, p. 271–8. http://dx.doi.org/10.5220/0005209202710278.

[26] Hagberg AA, Schult DA, Swart PJ. Exploring network structure, dynamics, and function using networkx. In: Varoquaux G, Vaught T, Millman J, editors. Proceedings of the 7th python in science conference. Pasadena, CA USA; 2008, p. 11–5.

[27] Milano M, Cinaglia P, Guzzi PH, Cannataro M. A novel local alignment algorithm for multilayer networks. Inform Med Unlocked 2024;44:101425. http://dx.doi.org/10.1016/j.imu.2023.101425, URL https://www.sciencedirect.com/science/article/pii/S235291482300271X.

[28] Wilson JD, Baybay M, Sankar R, Stillman PE. Fast embedding of multilayer networks: An algorithm and application to group fMRI. 2018, ArXiv, arXiv:1809.06437.

[29] Cinaglia P, Cannataro M. Identifying candidate gene-disease associations via graph neural networks. Entropy (Basel) 2023;25(6).