Published online 25 May 2021 *Nucleic Acids Research, 2021, Vol. 49, Web Server issue* ***W257–W262***
https://doi.org/10.1093/nar/gkab420

GRAPHERY: interactive tutorials for biological network algorithms

Heyuan Zeng[1,2], Jinbiao Zhang[3], Gabriel A. Preising[2], Tobias Rubel[2], Pramesh Singh[2] and Anna Ritz[2,*]

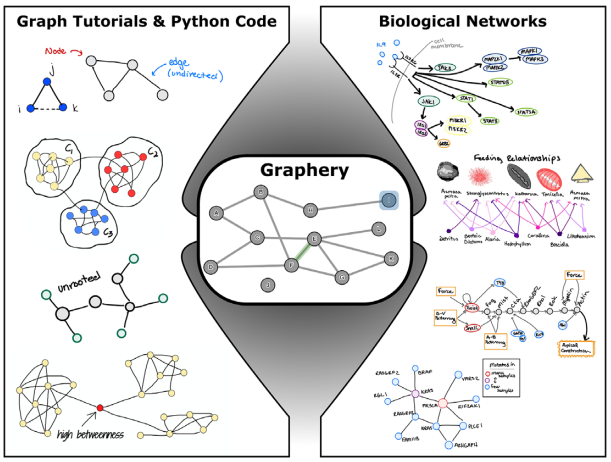
[1]Computer Science Department, Reed College, 3203 SE Woodstock Blvd, Portland, OR 97202, USA, [2]Biology Department, Reed College, 3203 SE Woodstock Blvd, Portland, OR 97202, USA and [3]Information and Communication Technology Department, Xiamen University Malaysia, Jalan Sunsuria, Bandar Sunsuria, 43900 Sepang, Selangor Darul Ehsan, Malaysia

Received March 05, 2021; Revised April 19, 2021; Editorial Decision April 30, 2021; Accepted May 03, 2021

ABSTRACT

Networks have been an excellent framework for modeling complex biological information, but the methodological details of network-based tools are often described for a technical audience. We have developed GRAPHERY, an interactive tutorial web-server that illustrates foundational graph concepts frequently used in network-based methods. Each tutorial describes a graph concept along with executable Python code that can be interactively run on a graph. Users navigate each tutorial using their choice of real-world biological networks that highlight the diverse applications of network algorithms. GRAPHERY also allows users to modify the code within each tutorial or write new programs, which all can be executed without requiring an account. GRAPHERY accepts ideas for new tutorials and datasets that will be shaped by both computational and biological researchers, growing into a community-contributed learning platform. GRAPHERY is available at https://graphery.reedcompbio.org/.

GRAPHICAL ABSTRACT



INTRODUCTION

Computational biologists have long used networks, or *graphs*, to model complex relationships that range in scale from molecular interactions to population and community dynamics. The popularity of graphs for bioinformatics and computational biology is marked by a wealth of review articles dedicated to the topic. For example, graphs have been extensively used to model molecular interactions such as protein interactions, metabolic signaling, and gene regulatory relationships (1–4). Network algorithms have also been developed to study the molecular systems biology of complex diseases, including identifying disease genes and pathways and predicting potential drug targets (5,6). Beyond molecular systems, graphs have been used to model neuron connections in the brain (7), social dynamics in animals (8,9) and energy flux in populations (e.g. food webs) (10,11).

Alongside this development of network-based methods, the amount of available network data (e.g. constructed

*To whom correspondence should be addressed. Tel: +1 503 517 4061; Email: aritz@reed.edu

from experiments or by carefully curating existing literature) has exploded. Graph databases and repositories have been developed to store, query and visualize biological networks (12–18), and biological network visualization is a subfield in its own right (19,20). Web-based network visualization has enabled a lightweight and interactive means for users to explore graphs without downloading a standalone application (21–23). Many graph algorithms that were designed for biological networks offer their own web-based tools, for example, GeneMANIA (24) and SteinerNet (25), or have plugins for existing platforms such as Cytoscape (26).

Recent growth in biological network data has inspired novel graph algorithms that, for example, uncover important components of or identify higher-order organization within complex networks. Many of these methods are extensions of classic and well-studied graph algorithms, such as clustering and community detection, random walks and belief propagation, and finding paths and trees. With the wealth of online datasets, web-based tools, and network graph visualization platforms, researchers can run these graph algorithms on their own data. However, when it comes to interpreting the outputs of these algorithms, biological researchers often face a major obstacle: How can you interpret the outputs of an algorithm if you do not know how the algorithm works?

There has been growing recognition that it is important for biological researchers to understand the concepts that underpin current computational methodologies (27–29). While review articles offer a broad view of an application or methodology, they tend to cite original work that may be dense for a non-computational audience. Primers and workshops have been developed to give biologists a deeper understanding of mathematical concepts, including machine learning (30), Bayesian network modeling (31), and deep learning (32). There are also a few network-based primers and textbook chapters dedicated to giving biologists more insight into graph statistics and algorithms (33–36). While these are rich with examples and principles of graph algorithms, they do not tend to be interactive or designed for researchers who want a high-level understanding of graphs.

We present GRAPHERY, a web-based platform that offers graph tutorials, example code, and interactive graphs, all in one place. GRAPHERY is aimed at helping biological researchers understand the core concepts upon which many state-of-the-art graph algorithms are built. The central idea behind GRAPHERY is that users can read tutorials and run the associated Python code on real-world biological networks that may be relevant to their field (Supplementary Figure S1). GRAPHERY's power is in the user's ability to select a specific graph to use when working through the tutorials and run that tutorial's Python code on the graph using a step-through debugger-style interface.

AUDIENCE

The majority of GRAPHERY users will be *visitors* who navigate the tutorials. Visitors can interact with graphs, read through the tutorials, run code, and even edit the code without logging in. We expect most visitors to be trainees and researchers from the biological sciences who commonly use bioinformatic tools to analyze their data. GRAPHERY tutorials provide written descriptions of the fundamental concepts behind common graph algorithms and enable visitors to interact with real-world networks. There is also a click-through interface that steps through Python code line-by-line highlighting important variables.

We hope that visitors will gain an intuition about the general control flow of a piece of code by stepping through it line-by-line. A smaller number of visitors may be more familiar with programming, and GRAPHERY provides features for modifying or writing code alongside the existing tutorials and networks. However, teaching Python programming is outside the scope of our webserver.

In this paper, we describe GRAPHERY from a visitor perspective. We first present tutorials and graphs as a visitor would interact with them. We then provide details about other user roles (such as authors, translators, and administrators), GRAPHERY's implementation, and the potential of GRAPHERY for education and training.

GRAPHERY TUTORIALS

Tutorials are the heart of GRAPHERY, as they are the view where all features come together (Figure 1). The right-hand pane of the tutorial view contains the *tutorial's content*, including text, images, and hyperlinks (Figure 1C). Tutorials are ordered in increasing complexity, beginning with tutorials that orient users, moving on to simple definitions and characteristics of graphs, and then delving into the main concepts behind graph algorithms (Table 1). We intentionally made these tutorials short and brief, with the goal that they build upon each other. In some cases, later tutorials use code that is described in earlier tutorials. Many of our tutorials are written in English, Chinese, and Spanish (Table 1 and inset of Figure 1C), with a goal to add more languages that reach a broader audience of network biology researchers.

GRAPHERY currently supports undirected, unweighted graphs. The tutorials cover a number of fundamental concepts about undirected graphs, including graph definitions (such as nodes, edges, node degree, paths and trees), graph statistics and properties (such as degree distribution, clustering coefficient, and acyclicity), simple graph algorithms (such as shortest paths and spanning trees), and more advanced graph concepts (such as random walks and community detection). These tutorials serve as a foundation for describing more complicated concepts, such as label propagation (e.g. used in GeneMania (24)), Steiner trees (e.g. used in SteinerNet (25)), and k-shortest paths (e.g. used in PathLinker (37)). We are actively developing new GRAPHERY tutorials, and we welcome suggestions and contributions.

GRAPHERY contains an interactive *graph visualization panel* (Figure 1B) that is built upon Cytoscape.js (21). Visitors can manually rearrange nodes in the graph and use the pan and zoom features. The drop-down in Figure 1A allows a visitor to select among a list of available biological networks for this tutorial; the networks themselves are described in more detail in the next section.

Visitors can click through each tutorial's Python code in the *code editor*, which employs a debugger-style format (Figure 1D and Supplementary Figure S2). The tool bar al-

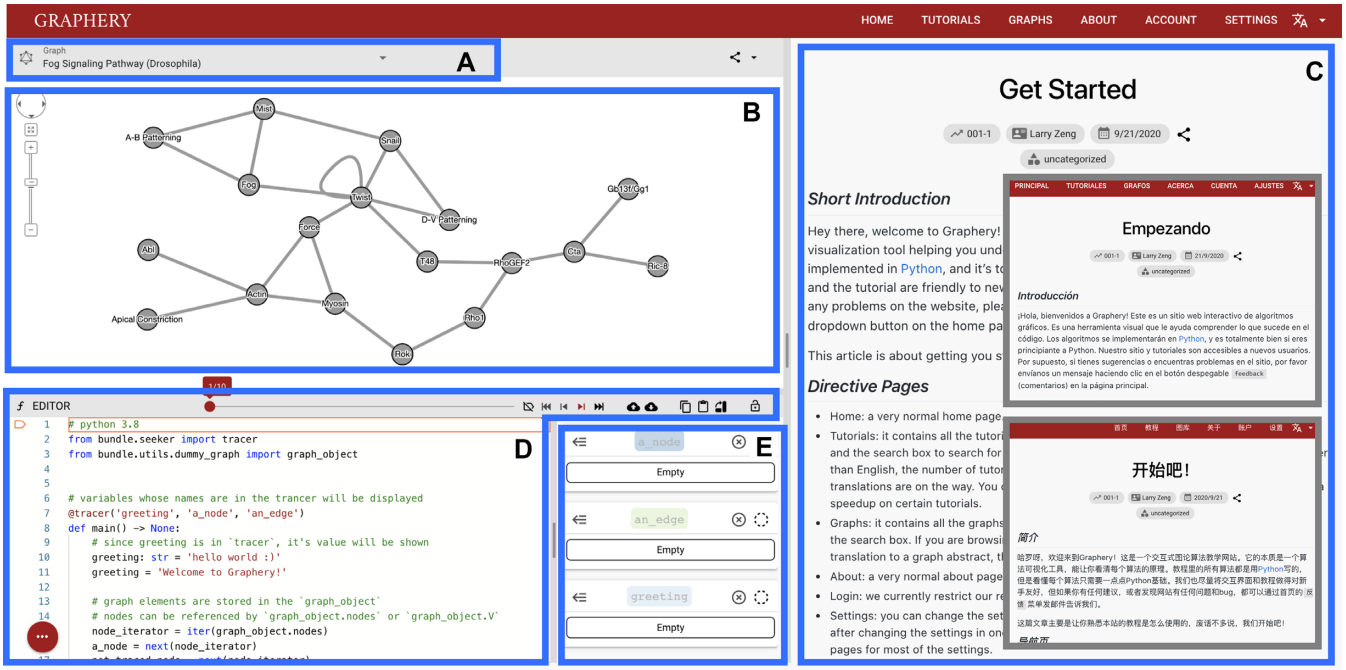


Figure 1. Tutorial view. This page includes (**A**) a drop-down to visualize different graphs, (**B**) an interactive graph visualization, (**C**) tutorial content (which can be switched to Spanish or Chinese translations), (**D**) a debugger-style editor and (**E**) a list of traced variables.

Table 1. Tutorials available in GRAPHERY, along with their available Spanish and Chinese translations (last updated 18 April 2021)

Number	Name	Description	EN-US	ES	ZH-CN
001-1	Get Started	Website structure and how to navigate tutorials	✓	✓	✓
001-2	Graph Primer	Introduction to graphs in Graphery	✓	✓	✓
001-3	Python Primer	Introduction to Python programming	✓	✓	✓
001-4	Programming in Graphery	Overview of the advanced program editor features	✓	✓	✓
101-1	Counting Elements	Introduction to graphs, nodes, and edges	✓	✓	✓
101-2	Depth First Search	Introduction to depth first search traversal	✓		
102-1	Degree Distribution	Degree distribution and network global structure	✓	✓	✓
102-2	Degree Distribution 2	Calculating the degree distribution, faster	✓	✓	✓
103-1	Shortest Paths	Computing the shortest path between two nodes	✓	✓	✓
103-2	Shortest Paths 2	Calculating the average shortest path length of a graph	✓	✓	✓
103-3	Betweenness Centrality	Calculating the betweenness centrality for all nodes	✓		
104-1	Trees & Acyclic Graphs	Introduction to trees and checking for graph acyclicity	✓	✓	✓
104-2	Spanning Trees	Spanning trees on unweighted graphs	✓		✓
105-1	Clustering Coefficient	Calculating a network's global clustering coefficient	✓		
105-2	Clustering Coefficient 2	Calculating the local clustering coefficient of nodes	✓		
201-1	Modularity	Community structure and network modularity	✓		
201-2	Modularity 2	Greedy community detection by maximizing modularity	✓		
202-1	Random Walks	Introduction to random walks	✓		
202-2	Random Walks 2	Random walks with restarts	✓		
203-1	Label Propagation	Introduction to label propagation	✓		

lows visitors to slide the step-counter or take one- or five-step jumps with the next/reverse buttons. Importantly, the code contains traced variables, which appear in a variable list (Figure 1E) and are also highlighted in the graph itself. Figure 2 shows the result after running the code in the *Get Started* tutorial, which selects a random node and a random edge in the graph.

Some visitors may be interested in learning more about how the Python code carries out the specified tasks. The code editor offers additional features for visitors to gain familiarity with programming in the context of the displayed tutorial. After unlocking the editor (by clicking the lock button on the toolbar or changing GRAPHERY's settings), visitors can change the variables being traced, change hard-coded parameters to observe their effects, or write completely new functions within the editor. Whenever the code is changed, visitors just need to execute the code with a one-button click and the new modifications will be applied. GRAPHERY provides remote execution capabilities (the cloud button with an up arrow), or local execution capabilities on the user's machine if a back-end server is running (the cloud button with a down arrow). Through these features, GRAPHERY visitors with different programming experience levels can learn from each tutorial.

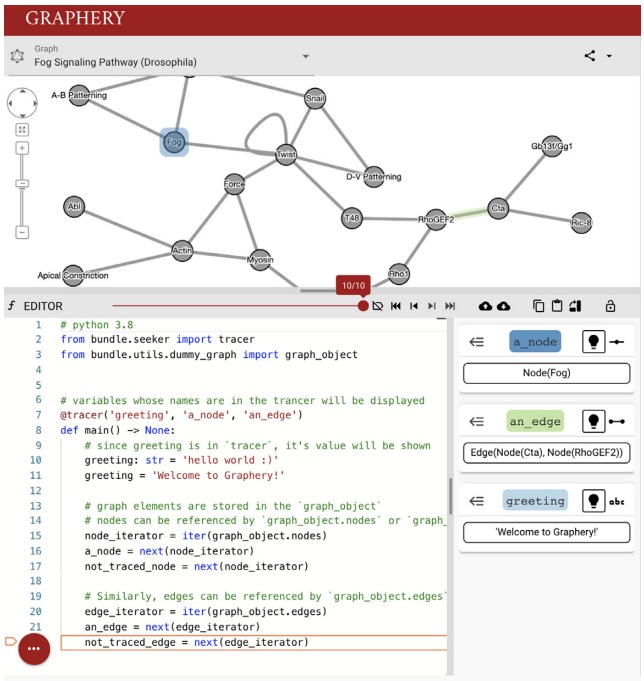


Figure 2. Example of the code after execution (panels D and E from Figure 1). The list of traced variables is now populated and graph variables are highlighted in the interactive panel.

GRAPHERY GRAPHS

If tutorials are the heart of GRAPHERY, biological networks are the cardiovascular system. The tutorials would not be useful on their own without the ability for visitors to walk through the code using graphs. Further, these graphs are not only examples, but come from real-world datasets and applications. Each graph contains a description of the underlying dataset (Supplementary Figures S1 and S2). The drop-down allows users to specify other graphs on which the Python code can be executed (Figure 1A). GRAPHERY also offers a playground feature where users can open a biological network with a code editor and interact with the graph independent of any specific tutorial (Supplementary Figure S3).

The biological networks in GRAPHERY span different scales of biology, from molecular interactions to population dynamics (Table 2). Further, they also capture networks relevant to different diseases (such as tuberculosis spread in a badger population, tumor cell evolution, and dysregulated protein networks). There is a current focus on protein-protein interactions and signaling pathways due to the authors' research areas, and we are continuing to expand the selection of biological networks to other domains and network topologies. Just like tutorials, we are actively posting new GRAPHERY graphs, and we welcome both suggestions and contributions.

OTHER ROLES IN GRAPHERY

While visitors are the most common type of users, GRAPHERY provides other roles that support the development and maintenance of the webserver. *Authors* are users who contribute new tutorials and/or graphs. Authors have control over which graphs are utilized in their tutorial, tutorial/network graph categories and whether their content is published (viewable by visitors). *Translators* are users who write translations of the existing tutorials. Translators have flexibility in their translated content, for example by including references to additional or alternate sources that are more readily accessible in some countries. *Administrators* have control over all content published on GRAPHERY, including those published by other authors and translators. These user roles require an account, which is maintained by an administrator.

Authors, translators, and administrators add content to GRAPHERY through a user-friendly control panel (Supplementary Section S2 and Supplementary Figure S4). The control panel supports image upload, Markdown-style editing with a live-update preview mode, and summaries of the posted tutorials and graphs. Authors and translators receive attribution on the content they contribute, as seen by the author tag in Figure 1.

Any user, including contributors such as authors/translators or visitors who interact with tutorials, can run Python code locally instead of in the cloud. Running the code locally gives users more control over the execution process, for example working with local files and using external Python modules. Supplementary Section S3 and Supplementary Table S1 provides more implementation details and instructions for local execution. The code is available on GitHub at https://github.com/Reed-CompBio/Graphery.

GRAPHERY FOR EDUCATION AND TRAINING

GRAPHERY has potential to be adapted for computational biology education and training at multiple levels, from high school students to bioinformaticians new to systems biology. In its current state, we believe that the majority of GRAPHERY visitors will be (a) trainees in the biological sciences who may commonly use tools with graph algorithms (b) undergraduate science majors without substantial programming experience. With these groups of trainees in mind, we considered core competencies described by two overlapping but distinct societies: the Curriculum Task Force of the International Society for Computational Biology (ISCB) Education Committee (29) and The Network for Integrating Bioinformatics into Life Sciences Education (NIBLSE) (47). ISCB's competencies are focused on bioinformatics training, whereas NIBLSE focuses on computational competencies designed for trainees in the life sciences. We mapped common GRAPHERY tasks (reading tutorials, interacting with graphs and code, modifying code, and writing code) with core competencies laid out by ISMB and NIBLSE (Supplementary Figure S5). All visitors will be able to evaluate graph algorithms in the context of systems biology and use bioinformatics tools to explore biological interactions and networks. As visitors engage more with the tutorials by learning about the code, they will begin to understand the algorithms and try adapting them with minor changes to the code. More experienced visitors who write new programs will learn about computing requirements needed to solve a problem. We note that there is no

Table 2. Biological networks available as GRAPHERY graphs (last updated 18 April 2021)

Name	Network type	Refs	EN-US	ES	ZH-CN
Badger Social Network	disease; social network	(38)	✓	✓	
Colorectal Cancer Evolution Tree	disease; evolution	(39)	✓	✓	
Competition Graph of Yellowstone Food Web	ecology; food web	(40)	✓	✓	
Fog Signaling Pathway (*fly*)	protein interactions; signaling pathway	(41,42)	✓	✓	✓
Food Web of Intertidal Species in WA	ecology; food web	(43)	✓	✓	✓
Interleukin-9 Signaling Pathway (*human*)	protein interactions; signaling pathway	(44)	✓	✓	
Pan-Cancer Network	disease; protein interactions	(45)	✓	✓	
Protein Transport Complex (*fly*)	protein interactions	(46)	✓		
Tutorial Network	*base network for tutorials*		✓	✓	✓

assessment in place for GRAPHERY tutorials, so carefully designed user studies are needed to fully determine GRAPHERY's potential in education and training.

DISCUSSION

GRAPHERY provides an interactive environment to learn about concepts in graph algorithms that form the foundation of many state-of-the-art methods. The combination of tutorials, code, and real-world graphs encourages learning in a more engaging way than the tutorials alone. We believe that GRAPHERY has the potential to become a mainstay learning tool for biological researchers at any career stage, from students to principal investigators.

We have come across some limitations in implementing our first tutorials and graphs. First, GRAPHERY visualizes undirected, unweighted graphs, which we acknowledge is only a subset of the type of graphs used in biological applications. We plan to extend GRAPHERY to support directed graphs, weighted graphs, and multigraphs (all of which Cytoscape.js supports). Second, we have found that even classic graph algorithms require a fair amount of background knowledge. While the tutorial content can describe this knowledge, the associated code must not incur an unreasonable number of steps for the code editor. For example, while graph clustering is intuitive to describe at a high level, the code to cluster graphs (and trace the relevant variables) may end up taking hundreds of thousands of steps, even for the smallest graphs. Our approach to this challenge is to ensure that the tutorials describe small, modular components of algorithms that can be combined for later tutorials. We also plan to implement additional useful step jumping tools to the execution set to help navigate code with many steps. Finally, we believe that the real-world biological networks in GRAPHERY are a fantastic way for biologists to better understand graph algorithms, but finding small and relevant graphs that describe real biological systems has been more difficult than we initially expected. Some of our networks have been suggested by biological experts which has proved to be a successful way to add graphs to GRAPHERY.

In addition to adding new graphs and tutorials, our team has plans for larger updates to help users better understand the provided code. The current graph API was originally developed for GRAPHERY tutorials, and users who wish to modify or write code must have a basic understanding of object-oriented programming and some knowledge of the underlying graph objects. While this information is described in the *Programming in Graphery* tutorial and on our documentation webpage (Supplementary Section S3), we plan to swap our custom API with a graph package such as `networkx` (48). This will help users already familiar with `networkx` to be able to modify code easily, and there will be a large amount of existing resources for programming support. While GRAPHERY is not explicitly designed to teach users how to program in Python, this modification will help users understand the programs provided in the tutorials.

Lastly, GRAPHERY will become a better webserver with more involvement from the network biology community. In addition to suggesting new tutorials and graphs, future versions will provide citations for current papers that use the concepts in each tutorial. We welcome anyone who wishes to contribute new content for the website; they will be acknowledged as an author on the networks and tutorials. GRAPHERY has potential to reach a broad audience through the translated pages, and we are actively seeking more translators to help us with this task (especially for languages that are not yet in place). With these goals in mind, GRAPHERY will help bridge the computational and biological worlds of systems biology.

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online.

FUNDING

National Science Foundation [BIO-ABI #1750981 to A.R. and BIO-MCB #1716964 to A.R. as co-PI]; G.A.P. was supported by a National Institutes of Health supplement PA-20-222 to [NIH-NIGMS #R15-DK116224-01 to S.C.P.R.]. Funding for open access charge: National Science Foundation [1750981].
Conflict of interest statement. None declared.

REFERENCES

1. Aittokallio,T. and Schwikowski,B. (2006) Graph-based methods for analysing networks in cell biology. *Brief. Bioinform.*, **7**, 243–255.
2. Cowen,L., Ideker,T., Raphael,B.J. and Sharan,R. (2017) Network propagation: a universal amplifier of genetic associations. *Nat. Rev. Genet.*, **18**, 551.
3. McGillivray,P., Clarke,D., Meyerson,W., Zhang,J., Lee,D., Gu,M., Kumar,S., Zhou,H. and Gerstein,M. (2018) Network analysis as a grand unifier in biomedical data science. *Annu. Rev. Biomed. Data Sci.*, **1**, 153–180.
4. Mitra,K., Carvunis,A.-R., Ramesh,S.K. and Ideker,T. (2013) Integrative approaches for finding modular structure in biological networks. *Nat. Rev. Genet.*, **14**, 719–732.
5. Cho,D.-Y., Kim,Y.-A. and Przytycka,T.M. (2012) Network biology approach to complex diseases. *PLoS Comput. Biol.*, **8**, e1002820.

6. Creixell,P., Reimand,J., Haider,S., Wu,G., Shibata,T., Vazquez,M., Mustonen,V., Gonzalez-Perez,A., Pearson,J., Sander,C. *et al.* (2015) Pathway and network analysis of cancer genomes. *Nat. Methods*, **12**, 615.
7. Bassett,D.S., Zurn,P. and Gold,J.I. (2018) On the nature and use of models in network neuroscience. *Nat. Rev. Neurosci.*, **19**, 566–578.
8. Sah,P., Leu,S.T., Cross,P.C., Hudson,P.J. and Bansal,S. (2017) Unraveling the disease consequences and mechanisms of modular structure in animal social networks. *Proc. Natl. Acad. Sci. U.S.A.*, **114**, 4165–4170.
9. Brugere,I., Gallagher,B. and Berger-Wolf,T.Y. (2018) Network structure inference, a survey: Motivations, methods, and applications. *ACM Comput. Surv. (CSUR)*, **51**, 24.
10. Girvan,M. and Newman,M.E. (2002) Community structure in social and biological networks. *Proc. Natl. Acad. Sci. U.S.A.*, **99**, 7821–7826.
11. Dunne,J.A., Williams,R.J. and Martinez,N.D. (2002) Food-web structure and network theory: the role of connectance and size. *Proc. Natl. Acad. Sci. U.S.A.*, **99**, 12917–12922.
12. Have,C.T. and Jensen,L.J. (2013) Are graph databases ready for bioinformatics? *Bioinformatics*, **29**, 3107.
13. Struck,A., Walsh,B., Buchanan,A., Lee,J.A., Spangler,R., Stuart,J.M. and Ellrott,K. (2020) Exploring integrative analysis using the BioMedical evidence graph. *JCO Clin. Cancer Informatics*, **4**, 147–159.
14. Fabregat,A., Korninger,F., Viteri,G., Sidiropoulos,K., Marin-Garcia,P., Ping,P., Wu,G., Stein,L., D'Eustachio,P. and Hermjakob,H. (2018) Reactome graph database: efficient access to complex pathway data. *PLoS Comput. Biol.*, **14**, e1005968.
15. Slenter,D.N., Kutmon,M., Hanspers,K., Riutta,A., Windsor,J., Nunes,N., Mélius,J., Cirillo,E., Coort,S.L., Digles,D. *et al.* (2018) WikiPathways: a multifaceted pathway database bridging metabolomics to other omics research. *Nucleic Acids Res.*, **46**, D661–D667.
16. Pratt,D., Chen,J., Welker,D., Rivas,R., Pillich,R., Rynkov,V., Ono,K., Miello,C., Hicks,L., Szalma,S. *et al.* (2015) NDEx, the network data exchange. *Cell Syst.*, **1**, 302–305.
17. Sah,P., Méndez,J.D. and Bansal,S. (2019) A multi-species repository of social networks. *Sci. Data*, **6**, 44.
18. Mering,C.V., Huynen,M., Jaeggi,D., Schmidt,S., Bork,P. and Snel,B. (2003) STRING: a database of predicted functional associations between proteins. *Nucleic Acids Res.*, **31**, 258–261.
19. Gehlenborg,N., O'donoghue,S.I., Baliga,N.S., Goesmann,A., Hibbs,M.A., Kitano,H., Kohlbacher,O., Neuweger,H., Schneider,R., Tenenbaum,D. *et al.* (2010) Visualization of omics data for systems biology. *Nat. Methods*, **7**, S56–S68.
20. Hu,Z., Mellor,J., Wu,J., Kanehisa,M., Stuart,J.M. and DeLisi,C. (2007) Towards zoomable multidimensional maps of the cell. *Nat. Biotechnol.*, **25**, 547–554.
21. Franz,M., Lopes,C.T., Huck,G., Dong,Y., Sumer,O. and Bader,G.D. (2016) Cytoscape.js: a graph theory library for visualisation and analysis. *Bioinformatics*, **32**, 309–311.
22. Lopes,C.T., Franz,M., Kazi,F., Donaldson,S.L., Morris,Q. and Bader,G.D. (2010) Cytoscape Web: an interactive web-based network browser. *Bioinformatics*, **26**, 2347–2348.
23. Bharadwaj,A., Singh,D.P., Ritz,A., Tegge,A.N., Poirel,C.L., Kraikivski,P., Adames,N., Luther,K., Kale,S.D., Peccoud,J. *et al.* (2017) GraphSpace: stimulating interdisciplinary collaborations in network biology. *Bioinformatics*, **33**, 3134–3136.
24. Franz,M., Rodriguez,H., Lopes,C., Zuberi,K., Montojo,J., Bader,G.D. and Morris,Q. (2018) GeneMANIA update 2018. *Nucleic Acids Res.*, **46**, W60–W64.
25. Tuncbag,N., McCallum,S., Huang,S.-S.C. and Fraenkel,E. (2012) SteinerNet: a web server for integrating 'omic' data to discover hidden components of response pathways. *Nucleic Acids Res.*, **40**, W505–W509.
26. Lotia,S., Montojo,J., Dong,Y., Bader,G.D. and Pico,A.R. (2013) Cytoscape app store. *Bioinformatics*, **29**, 1350–1351.
27. Carey,M.A. and Papin,J.A. (2018) Ten simple rules for biologists learning to program. *PLoS Comput. Biol.*, **14**, e1005871.
28. Wilensky,U., Brady,C.E. and Horn,M.S. (2014) Fostering computational literacy in science classrooms. *Commun. ACM*, **57**, 24–28.
29. Mulder,N., Schwartz,R., Brazas,M.D., Brooksbank,C., Gaeta,B., Morgan,S.L., Pauley,M.A., Rosenwald,A., Rustici,G., Sierk,M. *et al.* (2018) The development and application of bioinformatics core competencies to improve bioinformatics training and education. *PLoS Comput. Biol.*, **14**, e1005772.
30. Mu,F., Magnano,C., Treu,D., Gitter,A. and QUBES Educational Resources (2019) The ml4bio workshop: machine learning literacy for biologists. *GLBIO2019 Special Session on Bioinformatics Education*. doi:10.25334/Q44Q97.
31. Needham,C.J., Bradford,J.R., Bulpitt,A.J. and Westhead,D.R. (2007) A primer on learning in Bayesian networks for computational biology. *PLoS Comput. Biol.*, **3**, e129.
32. Webb,S. (2018) Deep learning for biology. *Nature*, **554**, 555–557.
33. Dong,X., Yambartsev,A., Ramsey,S.A., Thomas,L.D., Shulzhenko,N. and Morgun,A. (2015) Reverse enGENEering of regulatory networks from big data: a roadmap for biologists. *Bioinformatics Biol. Insights*, **9**, 61–74.
34. Pevzner,P. and Shamir,R. (2011) In: *Bioinformatics for Biologists*. Cambridge University Press.
35. Klipp,E., Liebermeister,W., Wierling,C. and Kowald,A. (2016) In: *Systems Biology: A Textbook*. John Wiley & Sons.
36. Junker,B.H. and Schreiber,F. (2011) In: *Analysis of Biological Networks*. John Wiley & Sons, Vol. **2**.
37. Ritz,A., Poirel,C.L., Tegge,A.N., Sharp,N., Simmons,K., Powell,A., Kale,S.D. and Murali,T. (2016) Pathways on demand: automated reconstruction of human signaling networks. *NPJ Syst. Biol. Applic.*, **2**, 16002.
38. Weber,N., Carter,S.P., Dall,S.R., Delahay,R.J., McDonald,J.L., Bearhop,S. and McDonald,R.A. (2013) Badger social networks correlate with tuberculosis infection. *Curr. Biol.*, **23**, R915–R916.
39. Dang,H.X., Krasnick,B.A., White,B.S., Grossman,J.G., Strand,M.S., Zhang,J., Cabanski,C.R., Miller,C.A., Fulton,R.S., Goedegebuure,S.P. *et al.* (2020) The clonal evolution of metastatic colorectal cancer. *Science Advances*, **6**, eaay9691.
40. Koirala,P. *et al.* (2018) Food Webs, Competition Graphs, and a 60-Year Old Unsolved Problem. In: *Teaching and Learning Discrete Mathematics Worldwide: Curriculum and Research*. Springer pp. 165–181.
41. Manning,A.J. and Rogers,S.L. (2014) The Fog signaling pathway: insights into signaling in morphogenesis. *Dev. Biol.*, **394**, 6–14.
42. Peters,K.A., Detmar,E., Sepulveda,L., Del Valle,C., Valsquier,R., Ritz,A., Rogers,S.L. and Applewhite,D.A. (2018) A cell-based assay to investigate non-muscle myosin II contractility via the folded-gastrulation signaling pathway in Drosophila S2R+ cells. *JoVE*, **138**, e58325.
43. Hui,D. (2012) Food web: concept and applications. *Nat. Educ. Knowl.*, **3**, 6.
44. Kandasamy,K., Mohan,S.S., Raju,R., Keerthikumar,S., Kumar,G.S.S., Venugopal,A.K., Telikicherla,D., Navarro,J.D., Mathivanan,S., Pecquet,C. *et al.* (2010) NetPath: a public resource of curated signal transduction pathways. *Genome Biol.*, **11**, R3.
45. Leiserson,M.D., Vandin,F., Wu,H.-T., Dobson,J.R., Eldridge,J.V., Thomas,J.L., Papoutsaki,A., Kim,Y., Niu,B., McLellan,M. *et al.* (2015) Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nat. Genet.*, **47**, 106–114.
46. Guruharsha,K., Rual,J.-F., Zhai,B., Mintseris,J., Vaidya,P., Vaidya,N., Beekman,C., Wong,C., Rhee,D.Y., Cenaj,O. *et al.* (2011) A protein complex network of Drosophila melanogaster. *Cell*, **147**, 690–703.
47. Wilson Sayres,M.A., Hauser,C., Sierk,M., Robic,S., Rosenwald,A.G., Smith,T.M., Triplett,E.W., Williams,J.J., Dinsdale,E., Morgan,W.R. *et al.* (2018) Bioinformatics core competencies for undergraduate life sciences education. *PLoS One*, **13**, e0196878.
48. Hagberg,A., Swart,P. and S Chult,D. (2008) Exploring network structure, dynamics, and function using NetworkX. Los Alamos National Lab.(LANL), Los Alamos, NM.