SoftwareX 33 (2026) 102459

Contents lists available at ScienceDirect

SoftwareX

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

StatGraph: an R package for complex network statistical analyses based on spectrum

Grover Enrique Castro Guzman [a], Diogo Ricardo da Costa [a], Eduardo Silva Lira [a], Suzana de Siqueira Santos [b], Taiane Coelho Ramos [c], Daniel Yasumasa Takahashi [d], Andre Fujita [a, e, *]

[a] Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Rua do Matão, 1010, São Paulo, SP, 05508-090, Brazil
[b] Center for Mathematics, Computing and Cognition, Federal University of ABC, São Paulo, SP, Brazil
[c] MidiaCom Lab, Institute of Computing - UFF, Niteroi, RJ, Brazil
[d] Brain Institute, Federal University of Rio Grande do Norte, Av. Nascimento de Castro, 2155—Morro Branco, Natal, RN, 59056-450, Brazil
[e] Division of Network AI Statistics, Medical Institute of Bioregulation, Kyushu University, Westwing Building 8F 801, Hospital Campus, Maidashi 3-1-1, Higashi-ku, Fukuoka, 812-8582, Japan

ARTICLE INFO

MSC:
05C80
05C50
62H30
62F10

Keywords:
Graph
Complex network
Parameter estimator
Model selection
Hypothesis test
Correlation
Clustering

ABSTRACT

The analysis of complex networks has traditionally relied on descriptive measures, such as centrality and clustering coefficients, as well as algorithms for detecting partitions and components. Additionally, a range of software packages has been designed for visualization and structural analysis. Although these approaches provide valuable information, they primarily focus on observable network features rather than their underlying generative mechanisms. We introduce **statGraph**, a nonparametric statistical framework for inferring properties of unobserved network generation mechanisms. At its core, **statGraph** leverages graph spectra, which intrinsically capture structural information and provide a robust basis for nonparametric inference. The package implements a range of methods, including graph entropy estimation, random graph parameter estimation, model selection procedures, statistical tests for comparing graphs, correlation analysis between sets of graphs, and graph clustering algorithms. By bridging graph theory and statistics via spectral analysis, **statGraph** provides a comprehensive toolkit for advancing the statistical analysis of complex networks.

Metadata

Code metadata.

Code metadata description	Metadata
Current code version	statGraph version 1.0.6
Permanent link to code/repository used for this code version	https://github.com/fujita-team/statGraph.git
Permanent link to Reproducible Code	https://zenodo.org/records/17592808
Legal Code License	MIT License
Code versioning system used	git
Software code languages, tools, and services used	R
Compilation requirements, operating environments & dependencies	R version ≥ 4.3.0
Support email for questions	andrefujita@usp.br

* Corresponding author at: Division of Network AI Statistics, Medical Institute of Bioregulation, Kyushu University, Westwing Building 8F 801, Hospital Campus, Maidashi 3-1-1, Higashi-ku, Fukuoka, 812-8582, Japan.
Email addresses: grover@usp.br (G.E.C. Guzman), diogocost2@gmail.com (D.R. da Costa), edulira@ime.usp.br (E.S. Lira), suzana.siqueira@ufabc.edu.br (S. de Siqueira Santos), taiane@ic.uff.br (T.C. Ramos), takahashiyd@gmail.com (D.Y. Takahashi), andrefujita@usp.br (A. Fujita).

https://doi.org/10.1016/j.softx.2025.102459
Received 5 October 2025; Received in revised form 19 November 2025; Accepted 22 November 2025
Available online 3 December 2025

Software metadata.

(Executable) software metadata description	Metadata
Current software version	R version 1.0.6
Permanent link to executables of this version	https://CRAN.R-project.org/package=statGraph
Permanent link to Reproducible Code	https://zenodo.org/records/17592808
Legal Code License	MIT License
Computing platforms/Operating Systems	Linux, macOS, Microsoft Windows, Unix-like
Installation requirements & dependencies	R https://github.com/fujita-team/statGraph/blob/master/statGraph/DESCRIPTION
Support email for questions	andrefujita@usp.br

1. Motivation and significance

Traditionally, we have modeled complex systems as random graphs (graphs that are generated by some random process) and analyzed them using approaches based on descriptive measures, such as vertex centrality and clustering coefficient [1], algorithms to find bipartitions, components, and matchings [2,3], and analysis software focused on graph visualization (**igraph** [4] and **Graphviz** [5]), descriptive measures of graph structure (**Tulip** [6], **Pajek** [7], **Cytoscape** [8]), and clustering of vertices (**Gephi** [9]).

However, all analyses focus on network features, rather than on our actual objects of interest. For example, consider two cities. Even if both cities exhibit the same number of roads and comparable measures of traffic flow, this does not imply that the cities are identical. The true criterion for equivalence lies in whether they were constructed according to the same underlying design or planning principles. Analogously, in the study of networks, it is not the observable structural features alone that define similarity, but rather the generative mechanisms from which these networks arise. In network terms, this is analogous to determining if two networks, despite having similar global statistics, originated from the same underlying stochastic process. Currently, few tools address this question. To our knowledge, **statnet** [10] is the only one that makes inferences from the generation mechanisms of networks. **statnet** includes tools for model estimation, evaluation, model-based network simulation, and visualization. However, **statnet** is restricted to parametric exponential-family random graph models. The problem is that the generation mechanism is rarely known. Thus, a nonparametric statistical framework becomes necessary to make inferences from the unobserved generation mechanisms.

Here, we introduce **statGraph**, a nonparametric statistical framework for making inferences from unobserved generation mechanisms. The core of **statGraph** is the analysis of graph spectra, which are inherent to all graphs, enabling the nonparametric analysis of complex networks. The rationale for this idea is that there is a close relationship between the graph's structure and its spectrum [11]. **statGraph** includes methods for calculating the entropy of the graph, estimating the parameters of a random graph model, a model selection approach, statistical tests to compare two or more (sets of) graphs, a method to identify a correlation between two vectors of graphs, and graph clustering algorithms.

In this work, "nonparametric" means that the data representation and many comparisons do not assume a generative graph model: procedures can operate directly on empirical spectral densities (ESDs) (e.g., distances, clustering, correlations). **statGraph** also provides optional model-based modules that fit or compare parametric random-graph families (e.g., ER, WS, BA) by matching model-implied spectra to the observed ESD. This dual design allows users to remain distribution-agnostic and obtain interpretable model parameters when appropriate.

This article provides an overview of data analysis using the **statGraph** package. This paper is organized as follows. Section 2 describes the notation and background used throughout the paper. Section 3 details the functions. Section 4 illustrates the application of **statGraph** to actual data. Finally, we describe the impact in Section 5 and make some final remarks in Section 6.

2. Software description

At the core of **statGraph** lies the concept of the graph spectrum. For an undirected graph $G = (V, E)$, where V is a set of n vertices and E is a set of edges, its adjacency matrix $\mathbf{A}(G)$ is an $n \times n$ matrix where $\mathbf{A}_{ij} = \mathbf{A}_{ji} = 1$ if vertices i and j are connected, and 0 otherwise. The spectrum of G is defined as the set of n real eigenvalues ($\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$) of its adjacency matrix $\mathbf{A}(G)$. For a random graph g, these eigenvalues are also random vectors, which allow for the computation of their expectation with respect to the graph's probability law [12]. The graph spectrum is a powerful representation for several reasons:

- **Nonparametric analysis:** Graph spectra are intrinsic to all graphs, making them suitable for nonparametric analysis of complex networks.
- **Dimensionality reduction:** The dimension of the spectrum is equivalent to the number of vertices (n), whereas fully describing a graph by its edges can require up to n^2 dimensions.
- **Structural insights [13]:** Analysis of the graph spectrum can reveal several structural properties, such as:
  1. The largest eigenvalue indicates the largest degree.
  2. Bipartiteness can be determined by checking if for every eigenvalue λ, its negative counterpart -λ also exists.
  3. Connectedness is evident if the largest eigenvalue is strictly greater than the second largest.
  4. The diameter of the graph corresponds to the number of distinct eigenvalues.
  5. The size of the largest clique is bounded by the largest eigenvalue.
- **Random graph model representation:** Different spectral distributions imply that the underlying random graph models or their parameters that generated the graphs are different. This makes the spectrum a good representation for random graph models [14–22].
- **Nonparametric likelihood substitute:** The empirical graph spectral density (ESD) serves as a nonparametric alternative to the likelihood function, which is often unknown for many random graph models. This characteristic enables **statGraph** to construct a general statistical framework for graphs, differentiating it from parametric methods tied to specific random graph models.

For a random graph g, the spectral density $\rho_g(\lambda)$ is formally defined as the limit of the expectation of a normalized sum of Dirac delta functions centered on the eigenvalues. This concept is crucial, as the spectral density of graphs generated by the same random process converges to a fixed function as the graph size n approaches infinity, given proper normalization. Notable examples include Wigner's semicircle law for Erdös-Rényi and k-regular graphs [23]. Furthermore, empirical studies have shown that empirical spectral density (ESD) characterizes various graph ensembles, including geometric, Barabási-Albert, and Watts-Strogatz graphs [12,23–26]. While the graph spectrum is invariant under vertex relabeling, it is an incomplete invariant for graph isomorphism (i.e., non-isomorphic graphs can be isospectral). However, in practical applications, the probability of encountering isospectral graphs diminishes

significantly as the number of vertices increases, making the spectrum a highly effective characterization tool for sufficiently large graphs [13].

Let $\langle . \rangle$ be the expectation with respect to the probability law of the random graph, and δ be the Dirac delta function. Then, the spectral density (of a random graph) with a normalizing sequence $(d_n)_{n\in\mathbf{N}}$ is defined as follows:

$$\rho_g(\lambda) = \lim_{n\to\infty} \left\langle \frac{1}{n} \sum_{j=1}^{n} \delta\left(\lambda - \frac{\lambda_j}{d_n}\right) \right\rangle, \tag{1}$$

when the limit exists.

3. Software functionalities

statGraph provides a range of functions for the statistical analysis of complex networks, all of which use the graph spectrum.

3.1. Empirical spectral density (ESD)

The ESD is the empirical distribution of the graph's eigenvalues. It serves as a natural estimator for the spectral density of random graphs. **statGraph** approximates the ESD by obtaining the graph's spectrum, applying a Gaussian kernel regression, and then normalizing the density to ensure an integral of one. The function graph.spectral.density is used for this purpose. Users can choose between two methods:

- `diag` (default): Computes all eigenvalues through diagonalization. It allows specifying the support of the density function, the number of discretization points, and the bandwidth calculation criteria (e.g., Silverman, Sturges, and biased/unbiased cross-validation).
- `fast`: Utilizes a degree-based approximation, particularly efficient for large graphs (ten thousand vertices or more).

The function returns a **statGraph** object containing details on the method, the data name, and the vectors for the x-axis (eigenvalues) and y-axis (density) of the spectral distribution.

3.2. Graph spectral entropy

Graph spectral entropy is an analogous concept to Shannon entropy, measuring the uncertainty of a random graph [12]. Defined as the integral of $\rho_g(\lambda)\log\rho_g(\lambda)$, it is based on differential entropy, which means that it can take negative values, indicating that the network is informative. The function graph.entropy estimates this value using a Gaussian kernel with bandwidth options similar to graph.spectral.density. It returns a **statGraph** object including the calculated entropy.

3.3. Parameter estimation

This functionality addresses the problem of estimating the parameter (θ) of a random graph model ($\mathcal{M}$) that generated a given graph (g) [12]. **statGraph** solves this by minimizing the Kullback-Leibler divergence (KLD) between the empirical spectral density of the observed graph ($\hat{\rho}(\lambda)$) and the parametric spectral density of the model (ρ_θ). This approach is similar to the maximum likelihood estimation method. The function graph.param.estimator performs this task. Its output is a **statGraph** object that contains the estimated parameter (`param`) and the minimum divergence (`dist`). For large graphs (e.g., $n > 10\,000$), users can significantly improve computational efficiency by setting `method` = 'fast', `dist` = 'L1', and `search` = 'ternary'. This configuration employs a linear-time algorithm for the spectral density approximation, uses the L_1 distance instead of the KLD, and a ternary search instead of a grid search [27,28], thereby avoiding the $\mathcal{O}(n^3)$ complexity of diagonalization.

3.4. Model selection

When faced with multiple candidate random graph models, selecting the one that best fits the data is crucial [12,29]. **statGraph** achieves this by minimizing the Graph Information Criterion (GIC), an analogous measure to the Akaike Information Criterion (AIC). The GIC is calculated as $2KLD(\hat{\rho}|\rho_{\hat{\theta}_i}) + 2|\hat{\theta}_i|$, where KLD is the Kullback-Leibler divergence between the empirical spectral distribution and the model's spectral distribution, and $|\hat{\theta}_i|$ is the number of parameters. The function graph.model.selection returns a **statGraph** object indicating the selected model(s) (that minimize GIC) and a matrix of parameter estimates and GIC values for all considered models.

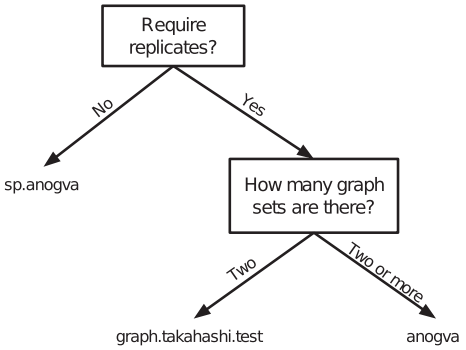

Fig. 1. Decision tree to select the comparative test that best fits the dataset. The order of the methods presented within each leaf is arbitrary.

3.5. Comparison tests

statGraph offers several statistical tests to determine whether two or more graphs or sets of graphs were generated by the same random graph model. A decision tree guides the selection of the appropriate test based on the presence of replicates and the number of graph sets (see Fig. 1).

- **Takahashi et al.'s test (graph.takahashi.test) [12]:** This test compares two graph populations (G_1 and G_2). It uses the Jensen-Shannon divergence (JSD) between their estimated spectral distributions to test the null hypothesis that JSD is zero. Both sets of graphs must have the same number of vertices.
- **Analysis of Graph Variability (ANOGVA) (`anogva`) [30]:** Similar to classical ANOVA, ANOGVA compares the spectral distributions of two or more sets of graphs to a combined average. A large sum of dissimilarities (KLD) indicates that at least one set was generated by a different model or parameters. All graphs in all sets must have the same number of vertices.
- **Semi-parametric ANOGVA (SP-ANOGVA) (`sp.anogva`) [31]:** This ANOVA-like procedure is designed for comparing two or more single graphs (i.e., without replicates). It is "semi-parametric" because it assumes a specific random graph model and tests its parameters. The parameters are estimated using graph.param.estimator; their variances are estimated via a bootstrap method, and then the parameters are compared using an F-statistic. Crucially, `sp.anogva` is suitable for graphs of different sizes, unlike graph.takahashi.test and `anogva`.

3.6. Correlation

statGraph provides a method to identify correlations between vectors of graphs [32]. This involves considering the graph model parameters as random variables and testing their independence. The package utilizes the largest eigenvalue of the graph adjacency matrix as a proxy for association, given its proportionality to the parameters in several random graph models. The function graph.cor.test implements this test. It can handle cases where the graphs in the two vectors (G_1 and G_2) have different numbers of vertices, as long as all graphs within G_1 have the same number of vertices and similarly for G_2.

3.7. Clustering

Instead of clustering vertices of a graph [33,34], **statGraph** provides methods to cluster the whole graphs. The clustering functionalities in the **statGraph** group graphs based on their similarity, measured by the

distances between their spectral densities. These distances can be based on Kullback-Leibler or Jensen-Shannon divergences, or on L_1 and L_2 norms.

- **Distance matrix (graph.dist):** This function calculates the pairwise distance or dissimilarity matrix among a list of graphs. For KLD, the resulting matrix is asymmetric because the j-th graph acts as a reference.
- **Hierarchical clustering (graph.hclust):** Adapts the standard hierarchical clustering algorithm by using the Jensen-Shannon divergence between graph spectra instead of Euclidean distance. It requires that all graphs have the same number of vertices. The function returns cluster labels and the hclust object.
- **K-means (graph.kmeans):** Similarly, graph.kmeans adapts the k-means algorithm using Jensen-Shannon divergence among graph spectra. It also requires graphs to have the same number of vertices. The function returns cluster labels and cluster centroids.
- **Random graph mixture models (graph.cem) [35]:** This advanced clustering method is inspired by Gaussian mixture models (GMMs). Here, each cluster is defined by a specific random graph model and its estimated parameters. It integrates KLD, the model selection approach, and the parameter estimator into its Expectation-Maximization (EM) algorithm. A key advantage is its suitability for graphs of different sizes, distinguishing it from graph.hclust and graph.kmeans. The function returns the cluster labels and the estimated parameters for each group.

3.8. Handling graphs with different numbers of vertices

Many of the statistical procedures implemented in statGraph operate directly on empirical spectral densities (ESDs). Because an ESD is defined on a support tied to the graph size, some methods require all graphs to have the same number of vertices, while others are designed to relax this constraint by working at the level of model parameters or mixture models. This distinction is operationally important when users analyze networks coming from heterogeneous sources (e.g., PPI networks of different species, brain networks with different parcellations) and was already implicit in the decision tree for comparison tests (Fig. 1), but it deserves to be made explicit. Concretely:

1. Methods that compare ESDs graph-by-graph (e.g., graph.takahashi.test, anogva) assume that spectra are comparable on the same dimensional support. For that reason, all input graphs must have the same number of vertices. Violating this assumption can lead to meaningless divergences or inflated dissimilarities.
2. Methods that work through parameter estimation or model-based clustering (e.g., sp.anogva, graph.cem) can accommodate graphs of different sizes because the comparison is carried out in the parameter space (estimated from the spectrum) or at the mixture-model level, not by aligning ESDs directly. These are the recommended options when users have only a single, large graph of varying sizes.
3. For very large graphs ($\approx 10^4$ vertices or more), users should prefer the method = "fast" option in the spectral-density and parameter-estimation functions, which avoids the $\mathcal{O}(n^3)$ diagonalization and makes it feasible to keep the "same-size" requirement without prohibitive running times. This is especially relevant when several large graphs with the same number of vertices must be compared. For illustration, the computational times (in seconds) to estimate the spectral densities of WS graphs with $n = 1\,000$ and $n = 10\,000$ vertices and setting method = 'diag' are (mean and 95 % confidence interval between brackets) 0.327 [0.199, 1.0266] and 50.861 [49.601, 52.742]. In contrast, by setting method = 'fast' we obtain 1.999 [1.931, 2.066] and 2.186 [2.112, 2.299]. Notice that for "small" graphs ($\approx$ 1 000 vertices), the diagonalization method (method = 'diag') is faster than the degree-based method (method = 'fast'). This is because we used the diagonalization method from LAPACK (Linear Algebra PACKage), which is optimized and written in C/C++. The computational advantage of the degree-based method becomes prominent only when n is large. The degree-based method is implemented in R. For both cases, we estimated the spectral density on an Intel Xeon Silver 4116 CPU (2.10 GHz) with 160 GB of memory and 64 cores.
4. Finally, some clustering routines (graph.hclust, graph.kmeans) also inherit the "same number of vertices" restriction because they measure distances between ESDs; when datasets are heterogeneous in size, the user should instead rely on graph.cem.

By making these constraints explicit, users can select a pipeline that matches their input data (homogeneous vs. heterogeneous sizes) and still benefit from the fast spectral approximations available in the package.

4. Illustrative example: enteric pathogen PPI networks

To demonstrate its practical utility, **statGraph** was applied to analyze six protein–protein interaction (PPI) networks of representative enteric pathogens obtained from string-db.org. After preprocessing (filtering edges, unweighting, and selecting the largest connected component), descriptive details of these networks, including the number of vertices and edges, were summarized (see Table 1).

Observing Table 2, we derive the following conclusions:

1. **Graph spectral entropy:** The calculated spectral entropies were negative for all six PPI networks, indicating that these networks are highly informative.
2. **Graph model selection:** Using graph.model.selection, the Barabási-Albert (BA) model consistently showed the lowest GIC value among ER, WS, and BA models for all species. This finding aligns with established literature suggesting that PPI networks often exhibit a scale-free structure [36–41].
3. **Parameter estimation:** Subsequently, graph.param.estimator was used to estimate the parameters for the selected BA model for each pathogen. A higher BA parameter might indicate a more pronounced hub-and-spoke structure [19].
4. **Comparison tests:** Given that the PPI networks were unitary and of different sizes, sp.anogva was employed for pairwise comparisons. After Bonferroni correction for multiple tests, the results indicated that most pairs of PPI networks were statistically different ($p < 0.05$). However, no statistical evidence was found to discriminate the network structures between *Escherichia coli* and *Yersinia enterocolitica* ($p = 1.0$), nor between *Listeria monocytogenes* and *Yersinia enterocolitica* ($p = 1.0$). The high power of this test was attributed to the substantial sizes of the networks.
5. **Clustering:** For clustering graphs of different sizes, graph.cem was chosen. It grouped *Campylobacter jejuni* and *Listeria monocytogenes* into cluster 1, and *Escherichia coli*, *Salmonella enterica* and *Shigella flexneri* into cluster 2. The estimated parameters for these clusters were 1.471335 and 1.231282, respectively.

Table 1
Summary of the number of vertices (proteins) and edges (interactions) before and after filtering edges with weight < 900.

Enteric pathogens (strain)	Original		After filtering (weight ≥ 900)	
	Vertices	Edges	Vertices	Edges
Campylobacter jejuni (NCTC11168)	1 624	299 944	886	11 832
Escherichia coli (O157:H7)	5 313	1 235 364	2 173	19 958
Listeria monocytogenes (EGDe)	2 846	658 052	1 500	17 166
Salmonella enterica (LT2)	4 448	982 270	2 960	28 616
Shigella flexneri (2 a str. 301)	4 164	862 170	1 616	14 432
Yersinia enterocolitica (8081)	5 251	1 079 200	1 487	13 554

Table 2
Graph spectral entropy, graph model selection, and graph parameter estimator (GPE) for six representative enteric pathogens. The graph model with the smallest GIC is the Barabási-Albert (BA) model (in bold). The GPE shows the estimated parameter considering the graph model with the smallest GIC.

	Graph spectral entropy	Graph model selection		GPE parameter
		model	GIC	
C. jejuni	−0.4501	ER	2.4321	1.4646
		WS	2.4649	
		BA	**1.5847**	
E. coli	−0.9332	ER	1.5981	1.3084
		WS	1.6656	
		BA	**0.8806**	
L. monocytogenes	−0.6952	ER	2.0552	1.2181
		WS	2.0816	
		BA	**1.2507**	
S. enterica	−1.0403	ER	1.5667	1.1019
		WS	1.5745	
		BA	**0.8893**	
S. flexneri	−0.8022	ER	1.7762	1.2859
		WS	1.7772	
		BA	**0.9586**	
Y. enterocolitica	−0.7579	ER	1.8706	1.3204
		WS	1.9531	
		BA	**1.1237**	

5. Impact

statGraph represents a significant contribution to the field of complex systems research. By facilitating efficient computation of spectral densities, inferring generative mechanisms, and enabling statistically rigorous comparisons of networks, the package allows researchers to move beyond mere descriptive measures. This shift allows for a deeper, more mechanistic understanding of how networks form, evolve, and which structural patterns predict system behavior. Such insights are pivotal for developing new theoretical frameworks and discovering universal principles that govern complex systems.

Moreover, as an open-source R package, **statGraph** democratizes access to advanced statistical tools, making sophisticated network analyses available to a broader global research community rather than confining them to highly specialized groups. In essence, **statGraph** transforms network science from a predominantly descriptive discipline into one capable of robust mechanistic inference at scale, effectively bridging the gap between theoretical innovation and real-world application across diverse domains such as neuroscience, molecular biology, biochemistry, and the social sciences.

6. Conclusions

In conclusion, the **statGraph** package provides a comprehensive toolkit for empirical network analysis, offering a spectrum-based statistical approach. Its functionalities, including parameter estimation, model selection, various statistical tests, correlation measures, and a suite of clustering algorithms, address critical needs in the study of complex networks, paving the way for novel discoveries and a deeper understanding.

CRediT authorship contribution statement

Grover Enrique Castro Guzman: Writing – original draft, Validation, Software, Methodology, Formal analysis. **Diogo Ricardo da Costa:** Writing – original draft, Software, Methodology, Formal analysis. **Eduardo Silva Lira:** Software. **Suzana de Siqueira Santos:** Software. **Taiane Coelho Ramos:** Software. **Daniel Yasumasa Takahashi:** Writing – review & editing, Methodology, Investigation, Conceptualization. **Andre Fujita:** Writing – review & editing, Writing – original draft, Supervision, Software, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used Grammarly and ChatGPT to improve language and readability. After using these tools/services, the authors reviewed and edited the content as needed and assume full responsibility for the publication's content.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered potential competing interests: Andre Fujita reports that financial support was provided by BlueMeme Inc. Andre Fujita reports a relationship with BlueMeme Inc. that includes funding grants. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has been supported by FAPESP (grants 2013/07699–0, 2020/02415–7, 2023/18337–3, 2024/09195–3, and 2024/03261–4); CNPq (grants 306811/2022–7, and 380313/2024–3); CAPES (finance code 001, 141492/2017–1, and 88882.328173/2019–01); JSPS KAKENHI grant number JP25K15019; BlueMeme Inc.; the MEXT Cooperative Research Project Program, Medical Research Center Initiative for High-Depth Omics and CURE: JPMXP1323015486 for MIB, Kyushu University; the Alexander von Humboldt Foundation; the Academy of Medical Sciences — Newton Fund; Wellcome Leap, and BlueMeme Inc. The infrastructure of Omics Science Center Secure Information Analysis System, Medical Institute of Bioregulation at Kyushu University, provides the (part of) computational resources.

References

[1] Opsahl T, Agneessens F, Skvoretz J. Node centrality in weighted networks: Generalizing degree and shortest paths. Soc Netw 2010;32(3):245–51. https://doi.org/10.1016/j.socnet.2010.03.006
[2] Gibbons A. Algorithmic graph theory. Cambridge University Press; 1985.
[3] Cameron K. Induced matchings. Discrete Appl Math 1989;24(1):97–102. https://doi.org/10.1016/0166-218X(92)90275-F. https://www.sciencedirect.com/science/article/pii/0166218X9290275F.
[4] Csardi G, Nepusz T. The igraph software package for complex network research. InterJournal 2006;Complex Systems:1695. https://igraph.org.
[5] Ellson J, Gansner ER, Koutsofios E, North SC, Woodhull G. Graphviz and dynagraph—static and dynamic graph drawing tools. 2004. 127–48.
[6] Auber D, Archambault D, Bourqui R, Delest M, Dubois J, Lambert A, Mary P, Mathiaut M, Melançon G, Pinaud B, Renoust B, Vallet J. TULIP 5. 2017. 1–28. https://doi.org/10.1007/978-1-4614-7163-9_315-1 https://hal.archives-ouvertes.fr/hal-01654518.
[7] Batagelj V, Mrvar A. Pajek—analysis and visualization of large networks. 2004) 77–103.
[8] Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. Genome Res 2003;13(11):2498–504. https://doi.org/10.1101/gr.1239303. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC403769/.
[9] Bastian M, Heymann S, Jacomy M. Gephi: An open source software for exploring and manipulating networks; 2009. http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154.
[10] Handcock MS, Hunter DR, Butts CT, Goodreau SM, Krivitsky PN, Morris M. ergm: Fit, Simulate and Diagnose Exponential-Family Models for Networks. 2018, The Statnet Project (http://www.statnet.org.), r package version 3.9.4. https://CRAN.R-project.org/package=ergm.
[11] Von Collatz L, Sinogowitz U. Spektren endlicher grafen: Wilhelm blaschke zum 70. geburtstag gewidmet. In: Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg, vol. 21. Springer; 1957. p. 63–77.
[12] Takahashi DY, Sato JR, Ferreira CE, Fujita A. Discriminating different classes of biological networks by analyzing the graphs spectra distribution. PLoS One 2012;7(12):e49949. https://doi.org/10.1371/journal.pone.0049949
[13] Van Dam ER, Haemers WH. Which graphs are determined by their spectrum? Linear Algebra Appl 2003;373:241–72.
[14] Erdős P, Rényi A. On random graphs i. Publ Math (Debrecen) 1959;6(290–297):18.
[15] Gilbert EN. Random graphs. Ann Math Stat 1959;30(4):1141–4.
[16] Penrose M. Random geometric graphs, vol. 5. OUP Oxford; 2003.

[17] Bollobàs B. Random graphs. 2001 2nd (New York: Academic).
[18] Watts DJ, Strogatz SH. Collective dynamics of 'small-world'networks. nature 1998;393(6684):440–2.
[19] Barabási A-L, Albert R. Emergence of scaling in random networks. science 1999;286(5439):509–12. https://doi.org/10.1126/science.286.5439.509
[20] Papadopoulos F, Kitsak M, Serrano MÁ, Boguná M, Krioukov D. Popularity versus similarity in growing networks. Nature 2012;489(7417):537–40.
[21] Holland PW, Laskey KB, Leinhardt S. Stochastic blockmodels: First steps. Soc Netw 1983;5(2):109–37.
[22] Mata ASD. Complex networks: a mini-review. Braz J Phys 2020;50:658–72.
[23] Wigner EP. On the distribution of the roots of certain symmetric matrices. Ann Math 1958;67(2):325–7.
[24] Arnold L. On the asymptotic distribution of the eigenvalues of random matrices. Mathematics Research Center, United States Army, the University of Wisconsin; 1967.
[25] Farkas IJ, Derényi I, Barabási A-L, Vicsek T. Spectra of "real-world" graphs: Beyond the semicircle law. Phys Rev E 2001;64(2):026704.
[26] Grenander U. Probabilities on algebraic structures. Courier Corporation; 2008.
[27] Newman MEJ, Zhang X, Nadakuditi RR. Spectra of random networks with arbitrary degrees. Phys Rev E 2019;99(4):042309. https://doi.org/10.1103/PhysRevE.99.042309
[28] Guzman GEC, Stadler PF, Fujita A. Efficient laplacian spectral density computations for networks with arbitrary degree distributions. Network Sci 2021;9(3):312–27.
[29] Santos SDS, Fujita A, Matias C. Spectral density of random graphs: convergence properties and application in model fitting. J Complex Netw 2021;9(6):cnab041.
[30] Fujita A, Vidal MC, Takahashi DY. A statistical method to distinguish functional brain networks. Front Neurosci 2017;11:231413.
[31] Fujita A, Silva Lira E, Siqueira Santos SD, Bando SY, Eleuterio Soares G, Takahashi DY. A semi-parametric statistical test to compare complex networks. J Complex Netw 2020. https://doi.org/10.1093/comnet/cnz028. https://academic.oup.com/comnet/advance-article/doi/10.1093/comnet/cnz028/5543003.
[32] Fujita A, Takahashi DY, Balardin JB, Vidal MC, Sato JR. Correlation between graphs with an application to brain network analysis. Comput Stat Data Anal 2017;109:76–92. https://doi.org/10.1016/j.csda.2016.11.016
[33] Daneshfar F, Soleymanbaigi S, Yamini P, Amini MS. A survey on semi-supervised graph clustering. Eng Appl Artif Intell 2024;133:108215.
[34] Daneshfar F, Dolati M, Sulaimany S. Graph clustering techniques for community detection in social networks. In: Community Structure Analysis from Social Networks. Chapman and Hall/CRC; 2025. pp. 81–100.
[35] Ramos TC, Mourão-Miranda J, Fujita A. Spectral density-based clustering algorithms for complex networks. Frontiers in Neuroscience 2023;17:926321.
[36] Uetz P, Giot L, Cagney G, Mansfield TA, Judson RS, Knight JR, Lockshon D, Narayan V, Srinivasan M, Pochart P, et al. A comprehensive analysis of protein–protein interactions in saccharomyces cerevisiae. Nature 2000;403(6770):623–7.
[37] Ito T, Chiba T, Ozawa R, Yoshida M, Hattori M, Sakaki Y. A comprehensive two-hybrid analysis to explore the yeast protein interactome. Proc Natl Acad Sci 2001;98(8):4569–74.
[38] Reboul J, Vaglio P, Rual J-F, Lamesch P, Martinez M, Armstrong CM, Li S, Jacotot L, Bertin N, Janky R, et al. C. elegans orfeome version 1.1: experimental verification of the genome annotation and resource for proteome-scale protein expression. Nat Genet 2003;34(1):35–41.
[39] Giot L, Bader JS, Brouwer C, Chaudhuri A, Kuang B, Li Y, Hao YL, Ooi CE, Godwin B, Vitols E, et al. A protein interaction map of drosophila melanogaster. science 2003;302(5651):1727–36.
[40] Li S, Armstrong CM, Bertin N, Ge H, Milstein S, Boxem M, Vidalain P-O, Han J-DJ, Chesneau A, Hao T, et al. A map of the interactome network of the metazoan c. elegans. Science 2004;303(5657):540–3.
[41] Han J-DJ, Bertin N, Hao T, Goldberg DS, Berriz GF, Zhang LV, Dupuy D, Walhout AJM, Cusick ME, Roth FP, et al. Evidence for dynamically organized modularity in the yeast protein–protein interaction network. Nature 2004;430(6995):88–93.