StatNet33 Documentation

Course: Technology Innovation Project

By

Group 33

Duc Phat Nguyen - 104203829

Thi Thanh Thuy Tran - 103514782

Hoa Ngoc Vu - 104188285

A Luan Luong - 104479251

Viet Hoang Lai - 104350042

Table of Contents

Disclaimer

While the documentation is structured for sequential reading, it can also serve as a reference guide. It incorporates an overview of the software's features and functionalities, detailed instructions on installation and usage, as well as guidance on models, maintenance, and security considerations. Additionally, it provides troubleshooting tips for common issues that users may encounter.

Subject to Change

This documentation serves as a guide to understanding the features and functionality of our application, but please be aware that the content provided herein is subject to change without notice. As we continue to improve and evolve our application, updates may be made to the software, its features, user interface, and documentation.

Reasons for Changes

The application undergoes continuous development, with enhancements, bug fixes, and new features being implemented to improve user experience and address evolving needs. Changes to the application and its documentation may occur due to:

- **User Feedback**: Feedback from users helps us identify areas for improvement and prioritize feature enhancements.
- **Technological Advancements**: Emerging technologies and industry standards may influence the development and architecture of the application.
- **Market Trends**: Changes in market demands and competitor offerings may necessitate adjustments to our application to remain competitive and meet user expectations.
- **Regulatory Compliance**: Updates may be required to ensure compliance with legal and regulatory requirements.

Impact on Users

While we strive to minimize disruption to users, it's important to note that changes to the application and its documentation may impact user workflows, functionality, and compatibility. We understand that adjustments may be required on the part of users to adapt to these changes, and we appreciate your patience and understanding as we work to deliver the best possible experience.

Contact Us

If you have any questions, concerns, or feedback regarding the application or its documentation, please don't hesitate to contact our support team. We are here to assist you and ensure that your experience with our application is positive and productive. Thank you for choosing our application. We appreciate your support and collaboration as we strive to deliver an exceptional user experience.

Note: This disclaimer is provided for informational purposes and does not constitute a legally binding agreement. The terms and conditions of use for the application are subject to the applicable terms of service and privacy policy.

1. Introduction

1.1. What is Social Network Analysis?

Social Network Analysis (SNA) is an interdisciplinary approach that combines sociology and mathematics to study the structures, patterns, and dynamics of social networks. A social network is a social structure that facilitates communication and interactions between a group of actors, such as individuals or organizations, who are connected by various types of relationships, including common interests, shared values, financial exchanges, friendship, or dislike.

Social networks exist on multiple levels, ranging from personal relationships like friendships and family ties to larger-scale interactions, such as disease spreading, company strategies, social movements, or even international relations. Research in numerous scientific fields has demonstrated the significance of social networks in understanding how problems are solved, diseases are transmitted, organizations are managed, and the extent to which individuals achieve their objectives.

Key aspects of SNA include:

1. **Network Structure**: SNA examines the overall structure of a network, including properties such as density (the proportion of actual connections among actors compared to the total possible connections), centrality (the importance or influence of actors based on their position in the network), and subgroups or communities within the network.
2. **Actor Attributes**: SNA considers the attributes or characteristics of the actors in the network, such as demographics, attitudes, or behaviours, and how these attributes relate to network structures and processes.
3. **Network Dynamics**: SNA explores how networks evolve and change over time, including the formation of new ties, the dissolution of existing ties, and the impact of external factors on network structure and behaviour.
4. **Social Capital**: SNA investigates how the position and connections of actors within a network provide them with resources, opportunities, and advantages, known as social capital.
5. **Diffusion and Influence**: SNA examines how information, ideas, behaviours, or innovations spread through a network and how actors influence each other based on their network positions and relationships.

SNA has applications in various domains, including sociology, mathematics, psychology, organizational studies, and computer science. It is used to study a wide range of social phenomena, such as the spread of information and misinformation, the formation of social movements, the structure of organizational networks, and the diffusion of innovations.

1.2. What is StatNet33?

StatNet33 is a Social Network Analyzer project to build a flexible and user-friendly, cross-platform tool for social network analysis and visualisation. It is developed in Python and R, which can run on both Windows and macOS. The app features an intuitive and user-friendly interface that guides users through the analysis process. It offers interactive controls and tooltips to assist users in navigating functionalities. The web application offers a set of features and capabilities to facilitate social network analysis:

- Users can easily *upload network data* in common formats such as *CSV* or *JSON*. The application provides validation tools to ensure data integrity. See more information in section 5, Data Processing.
- A wide range of network cohesion statistics to summarize and characterize the properties of the uploaded network. These statistics include *measures of centrality*, *connectivity*, *density*, and more. Read more in section 6, Network Descriptive Statistics.
- Networks can be visualised using *interactive and customizable graph layouts* provided by **Dash Cytoscape**. The application supports various styles, colours, and labelling options to enhance the visual representation of the network. Read details in section 7, Network Visualisation.
- The app utilised *Exponential Random Graph Models (ERGM)* to analyse the structural patterns and associated factors in the network. Users can use ERGM models to assess model fit and interpret the results. See more in section 8, Statistical Models.

1.3. Target audience

The web application for social network analysis is designed to cater to both general and professional audiences. This includes:

- Researchers and academics in the fields of social sciences, network science, and data analysis who require a tool to study and analyse social networks.
- Business professionals, such as marketing analysts and human resource managers, who seek to understand the dynamics of social networks within their organizations or target markets.
- Students and educators in relevant disciplines who want to explore and learn about social network analysis concepts and techniques.
- Curious individuals with an interest in understanding the structure and patterns of social networks in various contexts.

The application aims to provide a user-friendly interface and accessible features that allow users with different levels of technical expertise to perform social network analysis effectively.

2. System Architecture

StatNet33 is built mainly using python as front-end and back-end programming language. The system architecture follows a client-server model, where the user interacts with the application through a web browser (client), and the server handles the processing and analysis tasks. The description of each file can be referred to in the Appendix A. It is important to note that these files have been named to convey their functionality. Renaming them is strongly discouraged as it may break these intended functionalities. Subfolders can also be found in the main directory. It is not recommended to alter the structure of the folders and subfolders, as it can lead to disruption of the web application functionality.

2.1 Front-end components

- **Dash**: The user interface is developed using Dash, a Python framework for building interactive web applications. Dash allows for the creation of responsive and appealing layouts, integrating HTML, CSS, and JavaScript. It simplifies the process of creating data-driven applications.
- **Cytoscape**: A powerful network visualization library - Dash Core Component is used to render interactive network graphs within the web application. It provides a wide range of customization

options and supports various layout algorithms. StatNet33 will use Dash Cytoscape which is a package that extends and renders visualization at node and edge - levels.

2.2 Back-end components

- **Python**: The server-side logic of the application is implemented using Python. Python serves as the main programming language, handling data processing, analysis, and communication with the front-end.
- **R**: The application leverages the capabilities of R, a statistical programming language, for advanced network analysis tasks. R is particularly used for running Exponential Random Graph Models (ERGM) through the statnet package.
- **rpy2**: To facilitate the integration between Python and R, the application uses rpy2, a Python library that allows seamless communication and data exchange between the two languages. rpy2 enables the execution of R code within the Python environment.
- **Additional Libraries**: The application utilizes various Python and R libraries for specific functionalities:
    - **NetworkX**: A Python library for studying complex networks and graph structures.
    - **Pandas**: A Python library for data manipulation and analysis.
    - **NumPy**: a Python library for numerical computing, providing support for arrays, matrices, and mathematical functions.
    - **Seaborn**: a Python data visualization library based on matplotlib, providing a high-level interface for creating attractive statistical graphics.
    - **ergm (statnet)**: An R package for fitting and evaluating Exponential Random Graph Models.

3. Installation and Setup

This section provides instructions on how to set up and run StatNet33 for social network analysis on your local machine.

3.1 Prerequisites

Before installing and running the application, ensure that you have the following prerequisites installed:

- **Python (version >=3.7)**: Download and install Python from the official website (https://www.python.org). Make sure to add Python to your system's PATH environment variable during the installation process.
    - *Required Python Packages*: Install required Python packages listed in the requirements.txt file by running `pip install -r requirements.txt`.
- **R (version >=4.0)**: Download and install R from the official website (https://www.r-project.org). Follow the installation instructions specific to your operating system.
    - *Required R Packages*: Install any required R packages listed in the install_packages.R script by running `source("install_packages.R")` in your R console.
- **For Windows users**: You will need to set the R_HOME and R_LIBS environment variable to your R installation directory. This code snippet is included at the top of the `ergm.py`, located in the *pages* folder. For example:

```
os.environ['R_HOME'] = 'C:\\Program Files\\R\\R-[version]'
os.environ['R_LIBS'] = 'C:\\Program Files\\R\\R-[version]\\library'
```

You can replace the path to your R installation directory on Windows accordingly.

- **For macOS users**: You can set environment variables using the terminal using the `export` command. For example:

  ```
  export R_HOME=/Library/Frameworks/R.framework/Resources
  ```

  ```
  export
  R_LIBS=/Library/Frameworks/R.framework/Versions/[version]/Resources/libra
  ry
  ```

  You can replace the path to your R installation directory on macOS accordingly.

3.2 Installation guide for the web application

After making sure that Python and R have what it needs, StatNet33 can be accessed by opening the preferred web browser (e.g., Google Chrome, Mozilla Firefox, Safari). If users use Windows, users can double-click to run the `run_app.bat` file, which is a script containing commands, written in plain text that can be executed in Windows commands or programs with command-like interfaces.

Else, if users are running macOS, then users can use `run_app.sh`. A .sh file (also known as a shell script) is a text file containing a series of commands written in a scripting language, which can be executed sequentially by the Unix shell. Users can make the file executable with `chmod +x run_app.sh`, and then run it from the terminal using `./run_app.sh`. Alternatively, you can always choose to run our app.py from a terminal, an IDE, or a code editor that support Python environments.

4. User Interface

4.1 Overview of the user interface

The StatNet33 web application features a simple and user-friendly interface designed to guide users through the analysis process. The interface is divided into several main sections:

- **Navigation Bar**: Located at the top of the application, the navigation bar allows users to switch between different pages and functionalities, such as Home, General Metrics, Visualization, ERGM Model, and Model Simulation.

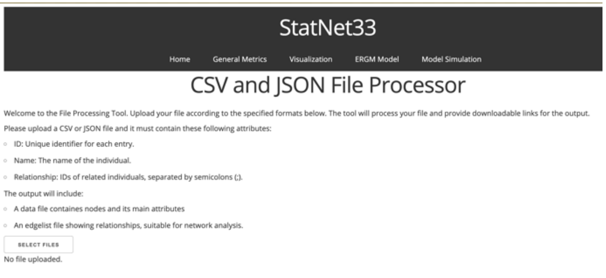
- **Main Content Area**: The central part of the interface displays the content specific to each page or functionality. For example, on the Home page, users can upload their network data files, while on the Visualization page, the network graph is rendered based on the uploaded data.

- **Settings and Options**: On many pages, users can find various settings and options to customize the analysis and visualization of their network data. These options include selecting layouts, styling nodes and edges, or configuring ERGM terms and node attributes.

4.2 Navigation and menu structure

- **Home**: The landing page of the application where users can upload their network data files.
- **General Metrics**: Displays various network metrics and statistics based on the uploaded data.

- **Visualization**: Renders an interactive network graph with customizable layouts and styling options.
- **ERGM Model**: Allows users to specify and estimate Exponential Random Graph Models (ERGMs) based on the uploaded network data.
- **Model Simulation**: Provides functionality to simulate networks based on the specified ERGM and compare the simulated networks with the original network.



CSV and JSON File Processor

Welcome to the File Processing Tool. Upload your file according to the specified formats below. The tool will process your file and provide downloadable links for the output.

Please upload a CSV or JSON file and it must contain these following attributes:

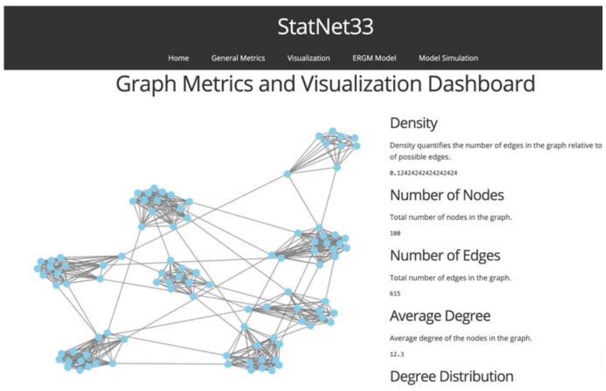
- ID: Unique identifier for each entry.
- Name: The name of the individual.
- Relationship: IDs of related individuals, separated by semicolons (;).

The output will include:

- A data file containes nodes and its main attributes
- An edgelist file showing relationships, suitable for network analysis.

No file uploaded.

Figure 1: Home Page



Graph Metrics and Visualization Dashboard

Density

Density quantifies the number of edges in the graph relative to of possible edges.

0.12424242424242424

Number of Nodes

Total number of nodes in the graph.

100

Number of Edges

Total number of edges in the graph.

615

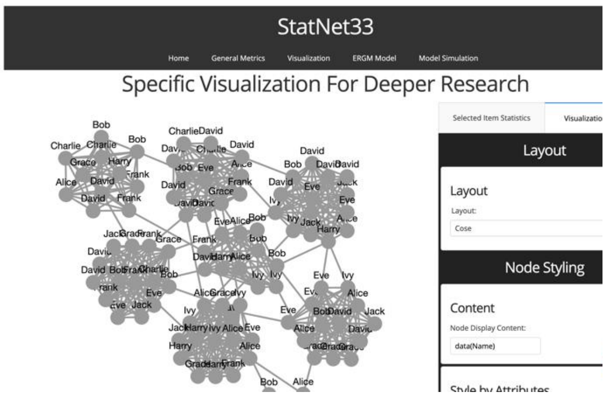
Average Degree

Average degree of the nodes in the graph.

12.3

Degree Distribution

Figure 2: General Metrics Page



Specific Visualization For Deeper Research

Layout

Layout

Layout: Cose

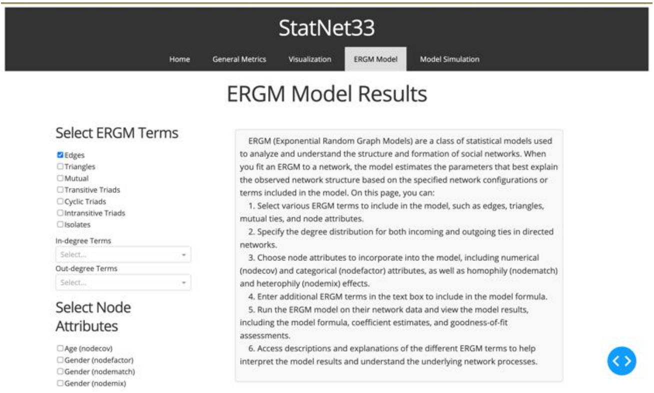
Node Styling

Content

Node Display Content: data(Name)

Style by Attributes

Figure 3: Visualisation Page



ERGM Model Results

Select ERGM Terms

- [✓ Edges]
- Triangles
- Mutual
- Transitive Triads
- Cyclic Triads
- Intransitive Triads
- Isolates

In-degree Terms: Select...

Out-degree Terms: Select...

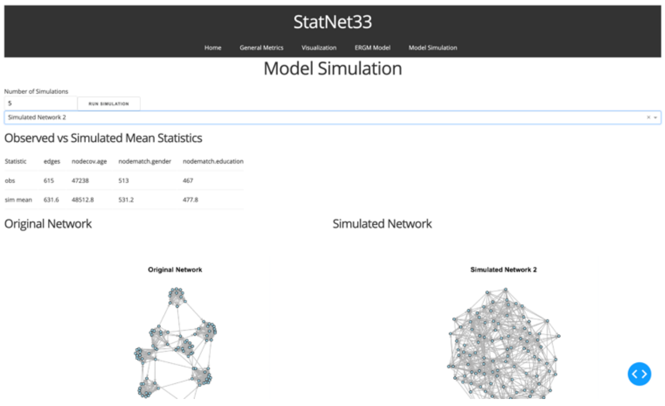
Select Node Attributes

- Age (nodecov)
- Gender (nodefactor)
- Gender (nodematch)
- Gender (nodemix)

ERGM (Exponential Random Graph Models) are a class of statistical models used to analyze and understand the structure and formation of social networks. When you fit an ERGM to a network, the model estimates the parameters that best explain the observed network structure based on the specified network configurations or terms included in the model. On this page, you can:

1. Select various ERGM terms to include in the model, such as edges, triangles, mutual ties, and node attributes.
2. Specify the degree distribution for both incoming and outgoing ties in directed networks.
3. Choose node attributes to incorporate into the model, including numerical (nodecov) and categorical (nodefactor) attributes, as well as homophily (nodematch) and heterophily (nodemix) effects.
4. Enter additional ERGM terms in the text box to include in the model formula.
5. Run the ERGM model on their network data and view the model results, including the model formula, coefficient estimates, and goodness-of-fit assessments.
6. Access descriptions and explanations of the different ERGM terms to help interpret the model results and understand the underlying network processes.

Figure 4: ERGM Model Page



Model Simulation

Number of Simulations: 5

Simulated Network 2

Observed vs Simulated Mean Statistics

Statistic	edges	nodecov.age	nodematch.gender	nodematch.education
obs	615	47238	513	467
sim mean	631.6	48512.8	531.2	477.8

Original Network

Simulated Network

Figure 5: Model Simulation Page

4.3 Input data format and upload process

StatNet33 accepts network data in CSV (Comma-Separated Values) or JSON (JavaScript Object Notation) format. The uploaded file must contain the following attributes:

- **ID**: Unique identifier for each entry.
- **Name**: The name of the individual.
- **Relationship**: IDs of related individuals, separated by semicolons (;).

To upload the data file, users can navigate to the Home page of the application. On this page, there is a file upload section where users can click on the "Select Files" button to browse and select the file from their local machine. For examples of detailed file formats and structure, please refer to appendix B.

Once the file is selected, the application will process the data and generate two output files:

- **Data File**: Contains nodes and their main attributes.
- **Edge list File**: Shows relationships between nodes, suitable for network analysis.

The processed files will be saved in the `pages/data` folder, and users will be provided with download links to access these files.

5. Data Processing

5.1 Data validation and error handling

During the data processing stage, StatNet33 performs several validation checks and error handling procedures:

- **File Type Validation**: The application checks the file extension to ensure that only supported file types (CSV and JSON) are processed. If an unsupported file type is uploaded, an error message is displayed to the user.
- **Duplicate ID Handling**: The application checks for duplicate entries based on the 'ID' column. If duplicates are found, they are removed, keeping only the first occurrence of each unique ID.
- **Null Handling**: Rows with null values in the 'ID' column are dropped from the dataset, as the 'ID' is a mandatory attribute. The application also checks for null values in columns other than 'ID' and 'Relationship'. If null values are found, a message is recorded to inform the user about the presence of null values in these columns.

5.2 Data preprocessing and cleaning

After the data validation and error handling steps, StatNet33 performs data preprocessing and cleaning:

- **Duplicate Removal**: Duplicate rows based on the 'ID' column are removed, keeping only the first occurrence of each unique ID.
- **Null ID Removal**: Rows with null values in the 'ID' column are dropped from the dataset.
- **Null Value Tracking**: The application keeps track of the number of rows with null values in columns other than 'ID' and 'Relationship'. This information is included in the messages provided to the user.

5.3 Data transformation and integration

After the data is pre-processed and cleaned, StatNet33 performs data transformation and integration to create the required output files. The application takes the uploaded data and automatically separates it into two distinct files:

- **Data File**: This file contains the main attributes of each node in the network, such as the unique identifier (ID) and name. It provides a comprehensive list of all the nodes in the network.
- **Edge list File**: This file focuses on the relationships between the nodes. It contains two columns, 'Source' and 'Target', which represent the connections or ties between nodes. Each row in the edge list file represents a relationship, with the 'Source' column indicating the originating node and the 'Target' column indicating the node it is connected to.

These two files are essential for network analysis and visualization. The data file provides information about the individual nodes, while the edge list file captures the structure and connections within the network. Once the transformation and integration process are complete, StatNet33 saves these files and provides users with download links. Users can easily access and download the data file and edge list file, which they can then use for further analysis or import into other network analysis tools.

Throughout the data transformation and integration process, StatNet33 keeps users informed by displaying relevant messages. These messages may include information about any modifications made to the data or any issues encountered during the process.

Here is a detailed of what the application does for gurus:

- **Data File Creation**: The application creates a new Data Frame called `'data_df'` by dropping the 'Relationship' column from the original Data Frame. This Data Frame contains the nodes and their main attributes.
- **Edge list File Creation**: The application iterates over each row of the original Data Frame and extracts the source-target relationships from the 'Relationship' column. It creates a new Data Frame called `'edgelist_df'` with 'Source' and 'Target' columns, representing the relationships between nodes.
- **File Saving**: The `'data_df'` and `'edgelist_df'` Data Frames are saved as CSV files in the application's data directory. The file names are generated based on the name of the uploaded file, appending `'_data'` and `'_edgelist'` respectively.
- **Download Links**: The application provides download links for the generated data file and edge list file, allowing users to access and download these files for further analysis.

Throughout the data processing stage, StatNet33 generates informative messages to keep users informed about any data modifications or issues encountered during the process. These messages are displayed to the user along with the download links for the processed files.

6. Network Descriptive Statistics

6.1 Report structure and sections

StatNet33 generates a report that presents network descriptive statistics and visualizations in a structured and user-friendly format. These sections provide an overview of the network's structural properties, connectivity patterns, and key metrics. The report is divided into 2 main sections:

- **Graph Visualization**:
    - Displays a visual representation of the network graph, allowing users to explore the network structure and node relationships.
    - Provides options for zooming, panning, and interacting with the graph.
- **Descriptive Statistics**:

  - Supports the computation and interpretation of various network descriptive statistics such as density, degree distribution, transitivity, ...
  - For details for each of the metrics, refer to section 6.2.

6.2 Supported descriptive statistics, computation, and interpretation.

The web application computes these descriptive statistics using NetworkX, a well-established library for network analysis. The computed metrics are then presented in the report, along with explanations and interpretations to help users understand their meaning and significance. Here is a more detailed about each metrics:

- **Density**:
  - *Computation*: Density is calculated as the ratio of the actual number of edges in the graph to the maximum possible number of edges. The formula is: $density = \frac{2m}{n(n-1)}$, where m is the number of edges and n is the number of nodes in an undirected graph. For directed graphs, the formula is: $density = \frac{m}{n(n-1)}$.
  - *Interpretation*: Density quantifies how close the graph is to a complete graph, where all possible edges are present. A density of 0 means no edges are present, while a density of 1 indicates a complete graph. A higher density suggests a more interconnected network, where nodes have a higher chance of being connected to each other. Conversely, a lower density indicates a sparser network with fewer connections between nodes.
- **Degree Distribution**:
  - *Computation*: The degree distribution is calculated by counting the number of nodes with each degree value. For undirected graphs, the degree of a node is the number of edges connected to it. For directed graphs, we consider both in-degree (number of incoming edges) and out-degree (number of outgoing edges) distributions separately.
  - *Interpretation*: The degree distribution provides information about the connectivity patterns of nodes in the network. It helps identify the presence of hubs (highly connected nodes) and peripheral nodes (nodes with few connections). A skewed degree distribution, such as a power-law distribution, indicates the presence of a few highly connected nodes and many nodes with low degrees. This can have implications for network resilience, information flow, and influence propagation.
- **Network Diameter**:
  - *Computation*: The network diameter is calculated as the maximum shortest path length between any pair of nodes in the graph. The shortest path length is the minimum number of edges that need to be traversed to go from one node to another. If the graph is disconnected, the diameter is often reported as infinity or a special value indicating disconnectedness.
  - *Interpretation*: The network diameter represents the maximum distance (in terms of the number of edges) between any two nodes in the network. It provides a measure of the overall size and span of the network. A smaller diameter indicates a more compact

network where nodes are more closely connected, allowing for faster information propagation or resource flow. On the other hand, a larger diameter suggests a more spread-out network where nodes are farther apart, potentially leading to longer paths and slower communication, but larger reach.

- **Average Path Length**:
  - *Computation*: The average path length is calculated as the average of the shortest path lengths between all pairs of nodes in the graph. It is computed by summing up the shortest path lengths between all node pairs and dividing by the total number of node pairs. If the graph is disconnected, the average path length is often calculated only for the largest connected component or reported as infinity.
  - *Interpretation*: The average path length quantifies the typical distance between nodes in the network. It measures how efficiently information or resources can flow from one node to another, on average. A smaller average path length indicates a more efficient network where nodes can quickly reach each other, facilitating faster communication or spreading of information. Conversely, a larger average path length suggests a less efficient network where nodes are more distant from each other, potentially leading to longer paths for information to travel.
- **Clustering Coefficient**:
  - *Computation*: The clustering coefficient quantifies the tendency of nodes to form clusters or tight-knit groups. It is calculated as the average of the local clustering coefficients of all nodes in the graph. The local clustering coefficient of a node is the fraction of pairs of its neighbours that are also connected to each other. Mathematically, it is defined as: $C_i = \frac{2e_i}{k_i(k_i-1)}$, where C_i is the local clustering coefficient of node i, e_i is the number of edges between the neighbours of node i, and k_i is the degree of node i.
  - *Interpretation*: The clustering coefficient measures the extent to which nodes in the network tend to cluster together. A higher clustering coefficient indicates a more clustered network, where nodes form tightly connected subgroups or communities. In such networks, there is a higher likelihood that two neighbours of a node are also connected to each other, forming triangles. This can be indicative of social networks, where friends of friends tend to be friends themselves, or in biological networks, where functionally related entities tend to interact more closely. On the other hand, a lower clustering coefficient suggests a more loosely connected network with fewer clusters.
- **Transitivity**:
  - *Computation*: is known as the global clustering coefficient. Transitivity is calculated as the ratio of the number of closed triads (triangles) to the total number of connected triples in the graph. A connected triple consists of three nodes where two of them are connected to the third node. Transitivity is defined as: $T = \frac{3 \times \text{number of triangles}}{\text{number of connected triples}}$.
  - *Interpretation*: Transitivity quantifies the overall tendency of the network to form clusters or tightly connected subgroups. It measures the probability that two neighbours of a node are also connected to each other, considering the entire network. A higher transitivity value indicates a more cohesive network with a higher prevalence of

triangles, suggesting that nodes tend to form dense subgroups. This can be observed in social networks, where friends of friends are more likely to be friends themselves, forming closed triads.

- **Assortativity Coefficient**:
    - *Computation*: The assortativity coefficient measures the correlation between the degrees of connected nodes in the network. It is calculated as the Pearson correlation coefficient between the degrees of nodes at either end of an edge. The formula is: $r = \frac{\Sigma\left((x_i-\mu_x)(y_i-\mu_y)\right)}{\sqrt{\Sigma((x_i-\mu_x)^2)}*\sqrt{\Sigma\left((y_i-\mu_y)^2\right)}}$, where x_i and y_i are the degrees of nodes at the ends of the i-th edge, and μ_x and μ_y are the mean degrees of the nodes.
    - *Interpretation*: The assortativity coefficient quantifies the tendency of nodes with similar degrees to connect to each other. It ranges from -1 to 1. A positive assortative coefficient indicates that nodes with high degrees tend to connect to other nodes with high degrees, and nodes with low degrees tend to connect to nodes with low degrees. This is often observed in social networks, where popular individuals tend to associate with other popular individuals. On the other hand, a negative assortativity coefficient suggests that nodes with dissimilar degrees are more likely to be connected, i.e., high-degree nodes tend to connect to low-degree nodes and vice versa. This can be seen in technological networks, such as the Internet, where high-degree nodes (hubs) connect to many low-degree nodes. An assortative coefficient close to zero indicates no pattern of degree correlation.

7. Network Visualisation

7.1 Supported network layouts

StatNet33 provides a range of network layouts and styles to visualize social networks effectively. The supported layouts include:

1. **Random Layout**: Positions nodes randomly within the available space. This layout is often used as a starting point for exploration or as a baseline for comparison with other layouts.
2. **Preset Layout**: Places the nodes at predetermined positions. Those positions are specified in each node's position field. The positions will be computed by Cytoscape.js behind the scenes, based on the given items of the layout dictionary.
3. **Grid Layout**: Arranges nodes in a grid structure, which is useful for visualizing networks with a regular or structured pattern.
4. **Circle Layout**: Arranges nodes in a circular layout, with edges connecting the nodes. This layout is useful for visualizing the overall structure and connectivity of the network.
5. **Concentric Layout**: Organizes nodes in concentric circles based on a specified node attribute or centrality measure. This layout helps in understanding the hierarchical structure of the network.

6. **Breadthfirst Layout**: Arranges nodes in a breadth-first tree layout, starting from a specified root node. This layout is helpful for visualizing the network's hierarchy and the distance of nodes from the root.

7. **Cose Layout**: Employs a force-directed layout algorithm to position nodes based on their connectivity. This layout is effective in revealing clusters and groups within the network.

External layouts can also be used thanks to the `load_extra_layout()` function from `dash_cytoscape`:

8. **Cose Bilkent Layout**: an extension of the Cose layout algorithm, developed by the Bilkent University. It employs a force-directed approach to position nodes based on their connectivity while considering additional factors such as edge crossing reduction and node overlap avoidance.
9. **Fcose Layout**: (Fast Cose) layout is a faster version of the Cose layout. It achieves improved performance by using techniques such as graph coarsening, which involves aggregating nodes and edges into clusters, and multilevel layout, which progressively refines the layout at different levels of detail.
10. **Euler Layout**: positions nodes in a way that reflects their set membership. It aims to create visually distinct clusters or groups of nodes based on their shared attributes or properties.
11. **Spread Layout**: distributes nodes evenly across the available space while minimizing edge crossings. It tries to maximize the distance between nodes and ensures that connected nodes are placed closer to each other.
12. **Dagre Layout**: visualises directed acyclic graphs (DAGs) or hierarchical networks. It positions nodes in a top-to-bottom or left-to-right manner based on their hierarchical relationships.
13. **Klay Layout**: The Klay layout is a versatile algorithm that can handle various types of networks, including hierarchical, layered, and cyclic graphs. It focuses on producing aesthetically pleasing and readable layouts by considering factors such as edge crossing minimization, node placement, and symmetry.

7.2 Node Styling

This section allows the user to customise the nodes dynamically with the according attributes:

- **Content**: Display node labels to provide additional information about each node by changing the node attribute in the parentheses.

- **Style by Attributes**: Display the node attribute through colour (categorical attribute or continuous values) and size (numerical attribute or centrality measure). Users can also define the colour palettes and the max size for them.

- **Shape**: Choose from various node shapes (ellipse, triangle, rectangle, …) to represent different node types or attributes. Users can also define node width and height for aesthetic.

- **Colour**: Display colour as an alternative way with hex code, opacity, and blackening for all the nodes rather than based on attributes.

- **Border**: Customise border for the nodes through border width, style, colour, and opacity.

- **Padding**: Customise the size of all nodes relative to various node measures such as width, height, …

7.3 Edges

This section allows the user to customise the edges dynamically with the according attributes:

- **Edge line**: Display edges through various customisations such as width, colours, and styles. Users have an option to colour the lines by weight and change the curving method of the edges through these options:
    - *Haystack*: Simple and straightforward approach for visualizing edges in a network. It represents edges as straight lines connecting the source and target nodes. The edges are drawn on top of each other, creating a "haystack" effect where multiple edges between the same pair of nodes are stacked together.
    - *Bezier*: Bezier curves are parametric curves defined by control points that determine the shape and curvature of the line. In the context of network visualization, the Bezier method creates smooth and visually appealing curved edges. It can help reduce edge crossings and improve the overall aesthetics of the network representation.
    - *Unbundled Bezier*: An extension of the Bezier method that aims to reduce visual clutter caused by overlapping edges. It separates edges that share the same source or target node by introducing additional control points and creating distinct curved paths for each edge. The unbundling process helps to visually separate the edges and improve the clarity of the network representation.
    - *Segments*: Represents edges as a series of straight-line segments connected by intermediate points. It provides a compromise between the simplicity of straight lines and the smoothness of curved edges. The edges are divided into multiple segments, allowing for more control over their shape and routing.
- **Edge Arrow**: Enables edge arrow and customise multiple arrow positions simultaneously, such as at the target or the source, or the mid-point, using colours, shapes, scales, and fill options.
- **Edge Endpoints**: Enables endpoints customisation for source or target, such as its types, distance, or relative endpoints widths or heights.

7.4 Labels

This section presents users with advanced settings for customising the label of each node rather than rely on predefined node label given at the top of page.

- **Label Names**: You can choose to further customise the existing label by choosing to include edge labels, edge source labels, or edge target labels.
- **Font Styling**: Select between node or edge to customise its colour, opacity, font family, size, weight, uppercase, or lowercase.
- **Text Wrapping**: Select between node or edge to customise its text to be wrapped or not at a certain width.
- **Label Alignment**: Provides options for users to align the nodes horizontally and vertically. Users can also change the distance between the edge label against its source or target label by pixel.
- **Margins**: Further customise the position of the labels by providing a fixed negative or positive pixel value for nodes and edges.

7.3 Interactive features

Dash Cytoscape provides users with multiple interactive features.

- **Zoom and Pan**:
    - *Zoom In/Out*: Users can zoom in and out of the network visualization using scroll wheel or pinch gestures on touch devices. This allows for a closer examination of specific regions or an overview of the entire network.
    - *Panning*: Users can pan or move around the network by clicking and dragging the background. This enables exploration of different parts of the network that may not be initially visible.
- **Node and Edge Selection**: Users can click on an edge or a node to select it. Selected edges or nodes are highlighted with a red border (node) or a red fill (edge).
- **Drag Interactions**:
    - *Node Dragging*: Users can click and drag nodes to manually reposition them in the network layout. This allows for customizing the node arrangement based on user preferences or to highlight specific patterns.
    - *Edge Manipulation*: Some network visualisations may allow users to manually adjust edge curves or control points to improve the clarity or aesthetics of the network.

8. Statistical Models (ERGM)

8.1 Overview of Exponential Random Graph Models (ERGM)

Exponential Random Graph Models (ERGMs) are a powerful class of statistical models used to analyse and understand the structure and formation of social networks. ERGMs are based on the idea that the observed network structure can be explained by a set of local network configurations or patterns, such as the number of edges, triangles, or other subgraph structures.

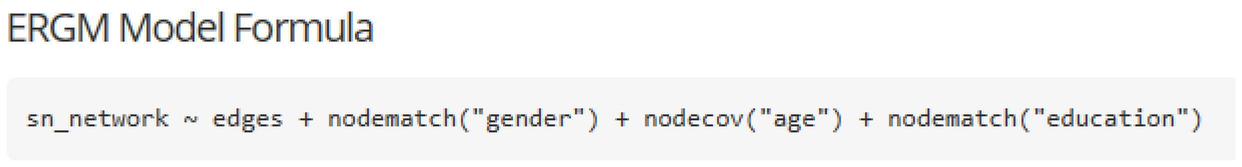
When fitting an ERGM to a network, the model estimates the parameters that best explain the observed network structure. These parameters correspond to the different network configurations included in the model and represent the strength and significance of their influence on the network formation process.

ERGMs allow researchers to test hypotheses about the underlying mechanisms that drive network formation, such as the tendency for nodes to form ties based on similarity (homophily), the presence of certain structural patterns (e.g., triangles or star-shaped configurations), or the influence of node attributes on tie formation.

8.2 Formula and model estimation

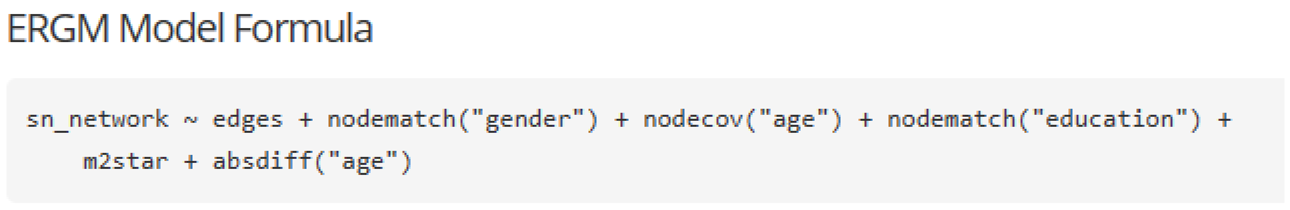
StatNet33 leverages rpy2, a Python library that enables integration between Python and R. With rpy2, we can specify and estimate ERGMs using the statnet package in R while still working within a Python environment.

To specify an ERGM formula and starting with the estimation, users can select various ERGM terms to include in the model. These terms can include basic structural properties like edges, triangles, or mutuals, as well as more advanced configurations such as node attributes, degree distributions, or custom subgraph patterns. StatNet33 supports a number of particular terms, you can see the full list and its concise descriptions in “ERGM Term Descriptions” in the “ergm” page of the application. Alternatively, you can find the detailed version and how to use it in Appendix D. If the users want to include more terms based on preference, the users can input the terms as a list separated by commas in Additional ERGM Terms section. For example, input `m2star, absdiff("age")` would yields a formula `sn_network ~ edges + m2star + absdiff("age")`, provided that edges option are enable.


ERGM Model Formula

```
sn_network ~ edges + nodematch("gender") + nodecov("age") + nodematch("education")
```

Figure 6: Formula with pre-defined options generated by user


ERGM Model Formula

```
sn_network ~ edges + nodematch("gender") + nodecov("age") + nodematch("education") +
    m2star + absdiff("age")
```

Figure 7: Formula with pre-defined options and additional terms generated by user

The selected terms are combined to form the model formula, which represents the hypothesized structure of the network. The formula is then passed to the ergm function from the statnet package in R using rpy2. The ergm function estimates the model parameters using maximum likelihood estimation or other estimation techniques. The results, including the estimated parameters and goodness-of-fit measures, are then retrieved from R and presented to the user.

8.3 Interpretation of model results

Once an ERGM is estimated, it is crucial to assess the model's fit to the observed network data. The web application provides various diagnostic tools and goodness-of-fit measures to evaluate the quality of the model and further understand the factors that influence network formation and the significance of different network configurations.

Goodness-of-fit (GOF) tests compare the observed network statistics with the statistics simulated from the estimated ERGM. These tests help determine whether the model successfully captures the essential structural properties of the observed network. Common GOF measures include degree distribution, edgewise shared partner distribution, geodesic distance distribution, and the overall goodness-of-fit for model statistics. In the application, only the assessment for model statistics is provided, giving a mean to assess whether the ERGM fits the observed network data well for the specified network statistics, helping users identify areas where the model may need improvement.

Note: If the users so choose to assess the other distribution in table format, the users can do so by going to `pages/ergm.py` and un-comment the desired GOF criteria.

8.4 Strategies for Model-Building and Limitations

While ERGMs are powerful tools for analysing social networks, they have some limitations and challenges that users should be aware of:

- **Model Convergence**: Fitting ERGMs to real social networks can be difficult, especially when including complex dependency terms. Compromises may be necessary to achieve model convergence, and close attention to MCMC diagnostics and goodness-of-fit is crucial.
- **Nodal Covariate Inclusion**: ERGMs offer various ways to include nodal covariates (e.g., nodefactor, nodematch, nodemix), each assuming different reasons for the attribute's importance. Users must carefully consider whether a variable increases an individual's overall propensity for ties, is a source of homophily, or both.
- **Boundary Specification**: Like other social network analysis methods, ERGMs require placing artificial boundaries on social connections, even when studying well-defined groups. Ties outside the defined group can still affect the distribution of ties within the group, which should be considered during interpretation.
- **Microlevel Process Limitations**: No ERGM perfectly captures the microlevel processes leading to global network structures. ERGMs are constrained to consider networks with similar density, even though the processes leading to a given network density should ideally be part of what is being explained.

To address these limitations and build effective models, users should:

- Develop strong hypotheses to inform the choice of covariates, including dependency terms, and how to include them in the model.
- Use these hypotheses to evaluate the adequacy of a model in answering the research question.
- Be aware of the limitations of ERGMs and interpret results accordingly.

9. Model Simulation

The Model Simulation in the StatNet33 application allows users to simulate networks based on the Exponential Random Graph Model (ERGM) fitted to the observed network data. The purpose of this feature is understanding the underlying mechanisms that drive the formation of networks, allowing researchers to validate their assumptions or test hypotheses about network formation processes. Users can assess the goodness-of-fit of the ERGM and gain understandings into the processes that shape the structure by comparing the simulated networks with the observed network. The primary objectives of the Model Simulation page are:

- Generating multiple simulated networks so that users can explore the variability and uncertainty associated with the network formation process.
- Visually inspecting the similarities and differences between the two networks, allowing users to assess how well the ERGM captures the structural properties of the observed network.
- Summarise statistics and goodness-of-fit measures that compare the observed network with the simulated networks. These metrics help users determine how well the ERGM fits the observed data and whether the model adequately captures the key features and patterns of the network.

This feature enables researchers and analysts to gain a deeper understanding of the network formation process, test the robustness of their models, and make informed decisions based on the simulated results. It enhances the interpretability and reliability of the ERGM analysis, allowing users to explore alternative network scenarios and assess the stability of their findings.

9.1 How It Works

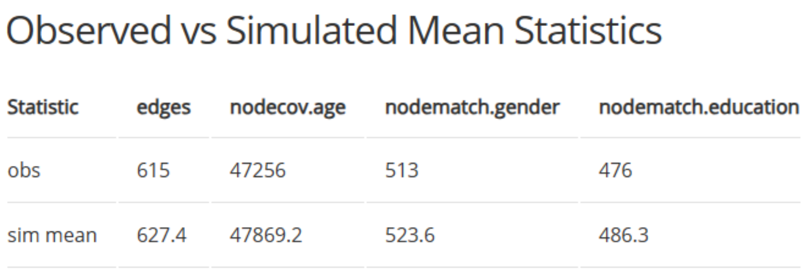
The Model Simulation page leverages R scripts and the statnet package to perform network simulation based on the fitted ERGM. Here's a step-by-step overview of how the simulation process works (assume that users should have already fitted an ERGM to their observed network data using the ERGM Model page):

1. **Specifying Simulation Parameters**: On the Model Simulation page, users can specify the number of simulated networks they want to generate. The application can handle unlimited simulation, though the performance may truly differ at 1000 mark.
2. **Running the Simulation**: When users click the "Run Simulation" button, the application triggers an R script (“`simulate_model.R`”) that performs the network simulation. The script takes the fitted ERGM and the specified number of simulations as inputs.
    a. The R script uses the “`simulate`” function from the statnet package to generate the specified number of simulated networks based on the fitted ERGM. Each simulated network represents a plausible realization of the network given the model parameters.
3. **Comparing Observed and Simulated Networks**: The application displays the observed network alongside a dropdown menu that allows users to select a specific simulated network. Users can visually compare the observed network with the selected simulated network to assess their similarity and differences.
4. **Presenting Summary Statistics**: The page presents a table that compares the observed network statistics with the mean statistics obtained from the simulated networks. These statistics provide a quantitative comparison between the observed and simulated networks, helping users evaluate the goodness-of-fit of the ERGM.

9.2 Results Interpretation

There are three main sections in displaying results:

- **Observed vs Simulated Mean Statistics**: compare the observed statistics of the real network (observed data) with the mean of simulated statistics of the simulated networks. Users can check whether it appears that the simulated sample mean statistics are in fact close to the observed statistics, assess how well the model captures the characteristics of the real network.

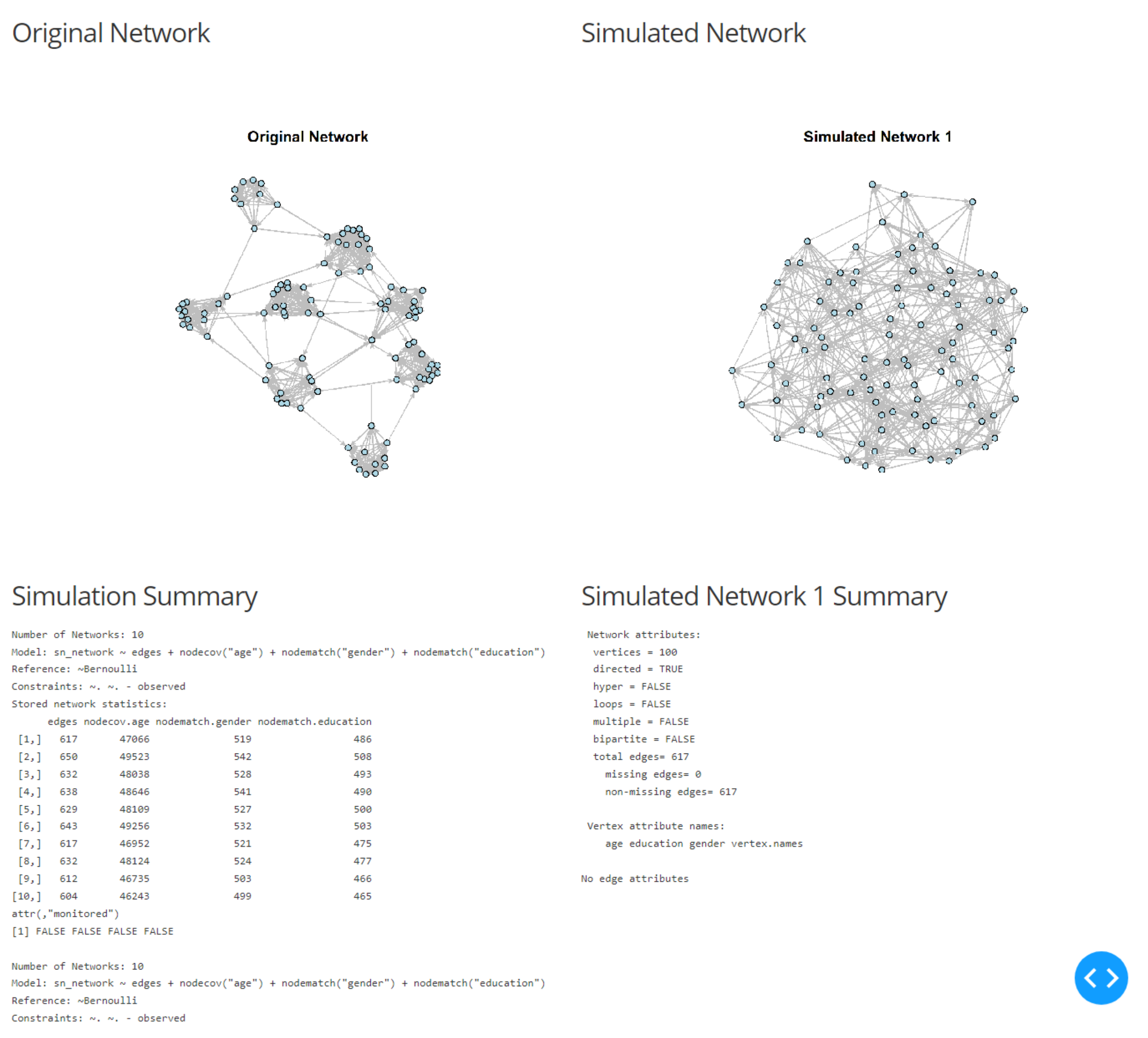


Observed vs Simulated Mean Statistics

Statistic	edges	nodecov.age	nodematch.gender	nodematch.education
obs	615	47256	513	476
sim mean	627.4	47869.2	523.6	486.3

Figure 8: Observed vs Simulated Mean Statistics

- **Original Network**: displays the original network structure by displaying a plot, serve as a starting point to compare with the plots created during the simulation. Under the plot is the summary of the simulation process, providing an overview of how each simulation are created.
- **Simulated Network**: displays the individual simulated network structure by displaying a plot, serve as a comparison to assess the model’s ability to capture the essence of the original one. Under the plot is the summary of the simulated network being plotted.



Simulation Summary

```
Number of Networks: 10
Model: sn_network ~ edges + nodecov("age") + nodematch("gender") + nodematch("education")
Reference: ~Bernoulli
Constraints: ~. ~. - observed
Stored network statistics:
      edges nodecov.age nodematch.gender nodematch.education
 [1,]   617       47066              519                 486
 [2,]   650       49523              542                 508
 [3,]   632       48038              528                 493
 [4,]   638       48646              541                 490
 [5,]   629       48109              527                 500
 [6,]   643       49256              532                 503
 [7,]   617       46952              521                 475
 [8,]   632       48124              524                 477
 [9,]   612       46735              503                 466
[10,]   604       46243              499                 465
attr(,"monitored")
[1] FALSE FALSE FALSE FALSE

Number of Networks: 10
Model: sn_network ~ edges + nodecov("age") + nodematch("gender") + nodematch("education")
Reference: ~Bernoulli
Constraints: ~. ~. - observed
```

Simulated Network 1 Summary

```
 Network attributes:
  vertices = 100
  directed = TRUE
  hyper = FALSE
  loops = FALSE
  multiple = FALSE
  bipartite = FALSE
  total edges= 617
    missing edges= 0
    non-missing edges= 617

 Vertex attribute names:
    age education gender vertex.names

No edge attributes
```

Figure 9: Original vs. Simulated Network

10. Troubleshooting and Support

10.1 Common issues and resolutions

10.1.1 No package named xxx

Issue: When running the web application, you may encounter an error message stating that a required package is not found, such as "No module named 'xxx'". **Resolution**:

- Make sure to install all the package necessary in both python and R, through requirement.txt and r_requirement.txt. The users can run the command: `pip install -r requirements.txt` (in Python console) and `source("install_packages.R")` (in R console).
- If it is any individual package, the users can install the missing package using pip by running the command: `pip install xxx`, where "`xxx`" is the name of the package.

- Verify that the package is installed in the correct Python environment. If you are using a virtual environment, ensure that it is activated before running the application.
- Double-check the spelling of the package name in the error message and ensure that it matches the actual package name.
- If the package is already installed, try upgrading it to the latest version using the command: `pip install --upgrade xxx`.

10.1.2 A formula takes too long to respond.

Issue: When specifying an ERGM formula and running the model estimation, the process may take an exceptionally long time to complete, causing the application to appear unresponsive.

Resolution:

- Simplify the ERGM formula by reducing the number of terms or removing complex terms that may be computationally expensive. Start with a simpler model and gradually add terms to assess their impact on performance.
- Consider using a smaller subset of the network data for initial testing and analysis. Working with a reduced dataset can help identify performance bottlenecks and optimize the formula accordingly.

10.1.3 A formula does not give back any respond.

Issue: After specifying an ERGM formula and running the model estimation, the application does not provide any output or results, leaving the user without any feedback.

Resolution:

- Check the server logs or console output on terminal for any error messages or traceback information. The logs may provide insights into the cause of the issue, such as syntax errors in the formula or missing data.
- Verify that the input network data is properly formatted and loaded into the application. Ensure that the necessary data files (e.g., edge list, node attributes) are present and accessible.
- Review the ERGM formula for any typos, missing terms, or incorrect syntax. Double-check the spelling and structure of the formula components.
- If the issue persists, try running the ERGM estimation with a different formula or a smaller dataset to isolate the problem. Gradually add complexity to the formula to identify the cause of the unresponsiveness.

10.2 Support channels and resources

Understandably, there will be more issues than the abovementioned problems. For any technical support or assistance regarding common issues during the deployment and usage of StatNet33, please feel free to contact the development team at:

- Technical Support Email: thisemailisf@ke.com
- Technical Support Phone: 1-800-DONTCALLUS

Users are encouraged to provide detailed information about the issue for swift resolution to the concerns. This includes error messages and steps to reproduce the problem. In addition to direct support, it is advisable to seek help from online forums and official websites. These platforms offer valuable resources and a community-driven support system for finding solutions to common issues.

11. Frequently Asked Questions

1. Q: What is an ERGM, and how does it differ from other network analysis methods?

A: ERGM stands for Exponential Random Graph Model. It is a statistical modelling approach that allows researchers to understand the formation and structure of social networks by considering various network configurations and node attributes. Unlike descriptive network measures, ERGMs provide a probabilistic framework to assess the significance of different network patterns and test hypotheses about the underlying social processes. For more information, please refer to section 8.1.

2. Q: What types of data can I use with this web application?

A: The web application supports network data in the form of edge lists (a list of edges or ties between nodes) and node attribute data (additional information about each node). The edge list should be provided in a CSV or JSON format, with columns representing the source and target nodes. Node attributes can also be supplied in a separate CSV or JSON file, with each row corresponding to a node and columns representing the attribute values. For detailed representations of data, please refer to Appendix B.

3. Q: How do I specify the ERGM formula in the application?

A: The web application provides an interface for specifying the ERGM formula. You can select the desired terms and configurations from pre-defined options. Additionally, you can refer to the documentation Appendix C or help section within the application for detailed instructions on constructing the ERGM formula.

4. Q: Can I include node attributes in my ERGM analysis?

A: Yes, the web application allows you to incorporate node attributes into your ERGM analysis. You can include terms like nodemix, nodefactor, or nodematch to examine the effect of node attributes on the likelihood of tie formation. The application provides options to select the relevant node attributes and specify how they should be included in the model.

5. Q: How do I interpret the ERGM results provided by the application?

A: The web application presents the ERGM results including the estimated coefficients, standard errors, and p-values for each term in the model, along with its explanation. The application also provides visualizations and summary statistics to aid in the interpretation of the results.

6. Q: Is there a limit to the size of the network that can be analysed with this application?

A: The web application can handle networks of various sizes, but the performance may vary depending on the complexity of the network and the available computational resources. For larger networks, the ERGM estimation process may take longer to complete. It is recommended to start with smaller or subsampled networks to get a sense of the performance and gradually increase the network size as needed.

12. Appendices

A. Code Inventory

- `app.py`: The main application file that sets up the Dash app and defines the layout and callbacks.
- `requirements.txt`: A file listing the Python dependencies required for the application.
- `run_app.bat`: A Windows batch file to run the application.
- `run_app.sh`: A shell script to run the application on Unix-based systems.
- `install_packages.R`: An R script listing the R packages required for the application.

Folders

- **assets**: Contains static assets such as CSS files and images.
    - `base.css`: Base CSS styles for the application.
    - `fonts.css`: CSS styles for fonts used in the application.
    - `welcome.png`: Welcome image displayed in the application.
- **dump**: Contains files for storing intermediate data and results.
    - `ergm_results.rds`: Stores the results of the ERGM analysis in RDS format.
    - `formula.txt`: Stores the ERGM formula used in the analysis.
    - `num_simulations.txt`: Stores the number of simulations for the ERGM analysis.
    - `selected_network.txt`: Stores the selected network for analysis.
    - `simulated_networks.rds`: Stores the simulated networks generated during the ERGM analysis.
- **pages**: Contains the main pages and components of the application.
    - `ergm.py`: Implements the ERGM analysis page and functionality.
    - `ergm_script2.R`: R script for running the ERGM analysis.
    - `generalmetric.py`: Implements the general network metrics page and functionality.
    - `home1.py`: Implements the home page of the application.
    - `plot_network.R`: R script for plotting the network.
    - `readme.txt`: Additional information or instructions for the pages.
    - `simulate5.py`: Implements the network simulation functionality.
    - `simulate_model.R`: R script for simulating networks based on the ERGM model.
    - `viz.py`: Implements the network visualization page and functionality.
- **dash_cytoscape**: Contains the Dash Cytoscape component files.
    - `CyLeaflet.py`: Implements the Cytoscape-Leaflet integration.
    - `Cytoscape.py`: Implements the Cytoscape component.
    - `dash_cytoscape.dev.js`: Development version of the Dash Cytoscape JavaScript file.
    - `dash_cytoscape.min.js`: Minified version of the Dash Cytoscape JavaScript file.
    - `dash_cytoscape.min.js.LICENSE.txt`: License file for the minified Dash Cytoscape JavaScript file.
    - `dash_cytoscape_extra.dev.js`: Development version of the Dash Cytoscape Extra JavaScript file.
    - `dash_cytoscape_extra.min.js`: Minified version of the Dash Cytoscape Extra JavaScript file.
    - `dash_cytoscape_extra.min.js.LICENSE.txt`: License file for the minified Dash Cytoscape Extra JavaScript file.

    - `metadata.json`: Metadata file for the Dash Cytoscape component.
    - `package.json`: Package configuration file for the Dash Cytoscape component.
    - `_imports_.py`: Imports file for the Dash Cytoscape component.
    - `__init__.py`: Initialization file for the Dash Cytoscape component.
- **data**: Contains data files used in the application.
    - `network_data.csv`: Dummy CSV file for testing containing network data.
    - `network_edges.csv`: Dummy CSV file for testing containing network edge data.
- **demos**: Contains demo files and assets.
    - `dash_reusable_components.py`: Reusable Dash components used in the application.
    - `__init__.py`: Initialization file for the demos folder.
- **src**: Contains source files for the application.
    - `cyto_cytoscape.jl`: Julia file related to Cytoscape integration.
    - `DashCytoscape.jl`: Julia file related to Dash Cytoscape integration.

Note: The `__pycache__` folders contain compiled Python bytecode files and can be ignored.

B. Sample input data format (CSV, JSON)

The StatNet33 application accepts network data in two common file formats: CSV (Comma-Separated Values) and JSON (JavaScript Object Notation). This appendix provides examples of how the input data should be structured in each format.

CSV Format

In the CSV format, each row in the CSV file represents a node in the network, with its corresponding attributes and relationships. The network data should have at least 2 columns:

- ID: A unique identifier for each node in the network.
- Relationship: A semicolon-separated list of IDs representing the nodes that the current node is connected to.

Here is an example of a file with 6 columns, ID as the first column and Relationship as the last:

```
ID,Name,Age,Gender,Education,Relationship

0,Grace,22,Female,Bachelor's Degree,10;12;20;38;61;71;73;85;87;90;91;99

1,Bob,53,Female,High School,13;29;30;42;45;47;77;89;49

2,Frank,50,Male,High School,11;18;19;24;44;54;58;63;76;95

3,Frank,41,Male,Bachelor's Degree,15;27;37;48;51;60;65;66;82;88;44

4,Bob,57,Male,Master's Degree,5;9;34;39;43;52;67;69;70;72;78;83;84
```

JSON Format

In the JSON format, the network data should be structured as an array of objects, where each object represents a node in the network. Like CSV, it should have at least 2 properties, ID, and Relationship. The JSON file should be a valid JSON array containing objects with the specified properties:

```
[
  {
    "ID": 0,
    "Name": "Grace",
    "Age": 22,
    "Gender": "Female",
    "Education": "Bachelor's Degree",
    "Relationship": "10;12;20"
  },
  {
    "ID": 1,
    "Name": "Bob",
    "Age": 53,
    "Gender": "Female",
    "Education": "High School",
    "Relationship": "13;29"
  },
```

Note: The Relationship property in both CSV and JSON formats represents the connections or ties between nodes. The IDs listed in the Relationship column/property should correspond to the ID values of other nodes in the network.

When preparing your network data for input into the StatNet33 application, ensure that it follows one of these formats (CSV or JSON) and includes the required columns/properties. The application will parse the input file and create the necessary data structures for analysis and visualization based on the provided data.

C. ERGM model specifications and parameter descriptions

The list below provides detailed descriptions of the ERGM terms, and their corresponding parameters used in the web application for social network analysis:

TERMS	DESCRIPTION	NOTES
EDGES	The edges term represents the baseline propensity for ties (edges) to form between nodes in the network.	The edges parameter estimates the overall density of the network, i.e., the likelihood of an edge existing between any two nodes, irrespective of other network configurations.
TRIANGLES	The triangles term captures the tendency for triangles (three nodes connected to each other) to form in the network.	The triangles parameter estimates the propensity for closed triads, i.e., the likelihood of an edge forming between two nodes that share a common neighbour, beyond what is expected by chance.

MUTUAL	The mutual term represents the tendency for reciprocated ties (edges that go both ways) to occur in the network.	The mutual parameter estimates the likelihood of a reciprocated tie forming between two nodes, given that a tie already exists in one direction.
TRANSITIVE TRIADS	The transitive triads term captures the tendency for transitive triads (if A is connected to B, and B is connected to C, then A is also connected to C) to form in the network.	The transitive triads parameter estimates the propensity for transitive closure, i.e., the likelihood of an edge forming between two nodes that share a common neighbour in a specific direction.
CYCLIC TRIADS	The cyclic triads term represents the tendency for cyclic triads (A is connected to B, B is connected to C, and C is connected to A) to occur in the network.	The cyclic triads parameter estimates the likelihood of cyclic closure, i.e., the propensity for edges to form in a circular pattern among three nodes.
INTRANSITIVE TRIADS	The intransitive triads term captures the tendency for intransitive triads (A is connected to B, and B is connected to C, but A is not connected to C) to form in the network.	The intransitive triads parameter estimates the propensity for open triads, i.e., the likelihood of an edge not forming between two nodes that share a common neighbour.
ISOLATES	The isolates term represents the tendency for nodes to have no ties (i.e., be isolated) in the network.	The isolates parameter estimates the likelihood of nodes having no connections, beyond what is expected by chance.
IN-DEGREE	The in-degree term captures the distribution of incoming ties (edges directed towards a node) in the network.	The in-degree parameter estimates the propensity for nodes to have a specific number of incoming ties. It can be specified as in-degree(k), where k represents the number of incoming ties.
OUT-DEGREE	The out-degree term captures the distribution of outgoing ties (edges directed away from a node) in the network.	The out-degree parameter estimates the propensity for nodes to have a specific number of outgoing ties. It can be specified as out-degree(k), where k represents the number of outgoing ties.
NODECOV	The nodecov term represents the effect of a numerical node attribute on the likelihood of tie formation.	The nodecov parameter estimates the influence of a specific node attribute on the propensity for nodes to form ties. It is specified as nodecov("attribute_name").
NODEMATCH	The nodematch term captures the tendency for nodes with the same attribute value to form ties (homophily effect).	The nodematch parameter estimates the likelihood of tie formation between nodes that share the same attribute value. It is specified as nodematch("attribute_name").

NODEFACTOR	The nodefactor term represents the effect of a categorical node attribute on the likelihood of tie formation.	The nodefactor parameter estimates the influence of a specific categorical attribute on the propensity for nodes to form ties. It is specified as nodefactor("attribute_name").
NODEMIX	The nodemix term captures the tendency for nodes with different attribute values to form ties (heterophily effect).	The nodemix parameter estimates the likelihood of tie formation between nodes with different attribute values. It is specified as nodemix("attribute_name").

D. Glossary of technical terms

Assortativity Coefficient	A measure of the correlation between the degrees of connected nodes in a network. It quantifies the tendency of nodes with similar degrees to connect to each other.
Centrality	A measure of the importance or influence of a node in a network based on its position and connectivity. Common centrality measures include degree centrality, betweenness centrality, and closeness centrality.
Clustering Coefficient	A measure of the tendency of nodes in a network to form clusters or tightly connected groups. It quantifies the proportion of a node's neighbours that are also connected to each other.
Density	The ratio of the actual number of edges in a network to the maximum possible number of edges. It measures how close the network is to being a complete graph.
Directed Graph	A graph in which edges have a direction, indicating a one-way relationship or flow between nodes.
ERGM (Exponential Random Graph Model)	A statistical model used to analyse and understand the structure and formation of networks by considering various network configurations and node attributes.
Goodness-of-Fit (GOF)	A set of tests and measures used to assess how well an ERGM fits the observed network data. It compares the observed network statistics with the statistics simulated from the estimated ERGM.
Graph	A mathematical structure consisting of a set of nodes (vertices) and a set of edges (links) that connect pairs of nodes.
Homophily	The tendency of nodes with similar attributes or characteristics to form connections or ties in a network.
Network Centralization	A measure of how centralized a network is around a few highly connected nodes. It quantifies the extent to which a network is dominated by a small number of central nodes.
Network Diameter	The maximum shortest path length between any pair of nodes in a network. It represents the longest distance (in terms of the number of edges) between any two nodes.
Network Visualization	The process of creating visual representations of networks using graphs, where nodes are represented as points or shapes, and edges are represented as lines connecting the nodes.
Node Attribute	Additional information or characteristics associated with each node in a network, such as demographic data, categories, or numerical values.

Shortest Path	The path between two nodes in a network that minimizes the number of edges traversed. It represents the shortest distance or the most efficient route between two nodes.
Social Network	A structure consisting of individuals, organizations, or entities (represented as nodes) and the relationships or interactions between them (represented as edges).
Social Network Analysis (SNA)	The study of social networks, focusing on the analysis of the structure, patterns, and dynamics of relationships among actors.
Transitivity	A measure of the tendency for triangles to form in a network. It quantifies the proportion of connected triples (two edges with a shared node) that are closed by forming a triangle.
Undirected Graph	A graph in which edges do not have a direction, indicating a mutual or symmetric relationship between nodes.