SUNBELT 2015: “Teaching Network Analysis with the statnetWeb R-Shiny Application”

ABSTRACT:

This workshop will introduce the new statnetWeb application as a tool for teaching statistical network analysis. The application is an interactive interface that runs in a browser window and provides access to the functionality of the statnet suite of R packages, without requiring that users have experience with R programming.

Topics covered will include: uploading network data, using plots and descriptive statistics to learn about the network, fitting exponential-family random graph models (ERGMs), model diagnostics, goodness of fit, simulations.

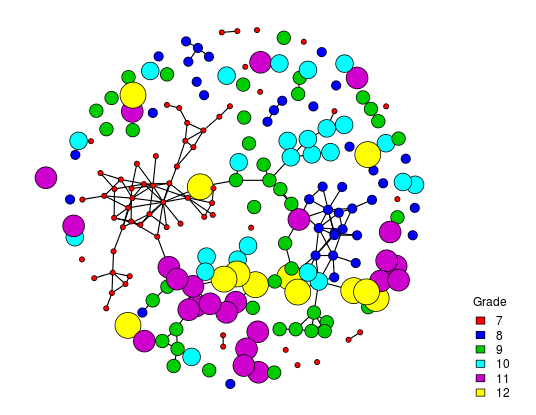
This workshop is intended for people who want to teach introductory network analysis courses or workshops. No coding experience is required, however we will discuss the coding structure and philosophy behind this R-Shiny application. Project code may be found at https://github.com/statnet/ergm-shiny, a prototype of the application may be found here: https://ebey.shinyapps.io/statnetWeb/.

OUTLINE

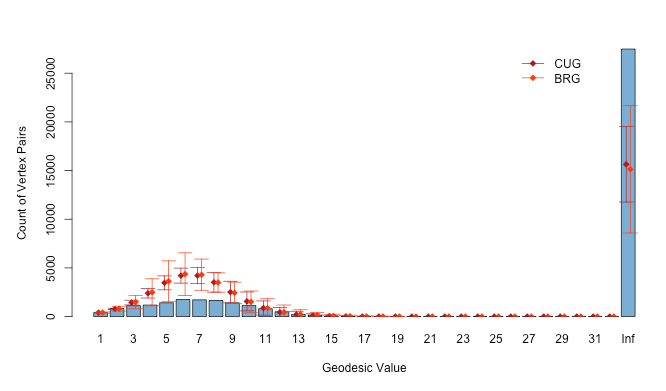
* **Introducing statnetWeb**
  + statnetWeb is a prototype web interactive interface for social network analysis. The functionality is based on the **ergm** package, a member of the **statnet** network analysis software suite, and also incorporates functionality from the associated packages **network** and **sna**.
  + To learn more about statnet, see the [statnet Wiki](https://statnet.csde.washington.edu/trac), where you can find publications, tutorials and recent news.
  + This workshop is meant to serve as a tutorial for the functionality of statnetWeb and as a guide to using the app as a teaching tool for introductory network analysis.
  + Headers that begin with “statnetWeb:” correspond to those pages in the app.
* **Introduction to network analysis**
  + In the network depiction of a community, *nodes* represent individual entities (people, organizations, etc.), which may be connected to each other by *edges*.
    - Nodes can have categorical or quantitative attributes associated to them (age, sex, etc.).
    - Edges can be directed/undirected and valued/binary
    - Types of nodes and edges lead to types of networks (bipartite, valued, directed, etc.)
    - In simple random graphs, each edge between two nodes is equally likely
      * Not the case for most social networks
      * The chance of a tie between nodes i and j may depend on the attributes of i and j
    - By analyzing a social network, we can gain insight to the underlying structure of the network and how that structure influences the individual actors and relationships in the network
    - Then, we can predict the formation and structure of other networks
  + How networks are denoted (adjacency matrices, incidence matrices, edge lists)
* **statnetWeb: Data Upload**
  + Sample datasets are available for immediate use
    - Users can experiment while becoming familiar with statnetWeb
  + Users can upload files created in R, Excel or Pajek
    - R
      * Upload a matrix of relational data or a statnet network object
      * Objects should be exported from R using the command `save(objectname, file=”newfilename”)`

By default the file will be saved into the current working directory, but the full path to a new location can be specified in the `file=` argument, or set `file=file.choose(new=TRUE)` to use a save dialog box.

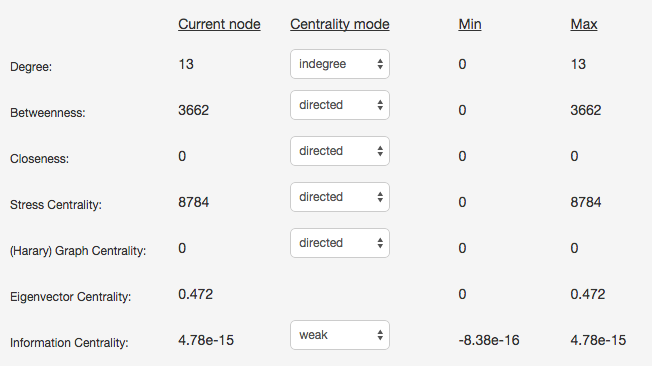
* + - Excel
      * Specify the type of matrix and upload a .csv file of relational data.
      * Adjacency matrices should have vetex labels in the first row or column.
      * Incidence matrices should have edge labels in the first row and vertex labels in the first column.
      * Edge lists should not have row or column labels
    - Pajek
      * Upload network or project files (.net or .paj, respectively).
  + Network summary on right side of page
    - After uploading, verify that network has intended properties
  + Modify network as necessary (symmetrize or add attributes)
* **statnetWeb: Network Descriptives**
  + Network descriptives are valuable for gaining insight into the observed network. Exploring plots and descriptive statistics before fitting a model can lead to better model formulations and improve interpretation of results.
  + Plot: display options can be revealing
    - e.g. easy to see clustering when nodes of faux.mesa.high are color-coded and sized by Grade



* + Degree dist:
    - Do most nodes have similar degree? Are there a few nodes with high degree (“hubs”) and many with low degree?
    - Does the distribution vary by attribute?
  + Geodesic dist: lengths of shortest paths (compared to null models)
    - Conditional uniform graphs: Draws from the distribution of simple random graphs with the same fixed density as the observed network.
    - Bernoulli random graphs: Draws from the distribution of simple random graphs with the same stochastic tie probability as the observed network (exact densities will vary).



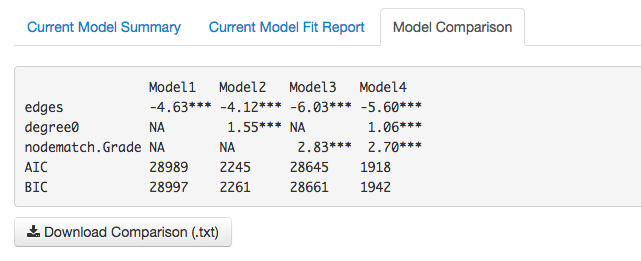
* + - The faux.mesa.high network has fewer paths of length 5-8 than we would expect, and more node pairs with paths of length inf (unreachable)
  + More: node- and graph-level indices



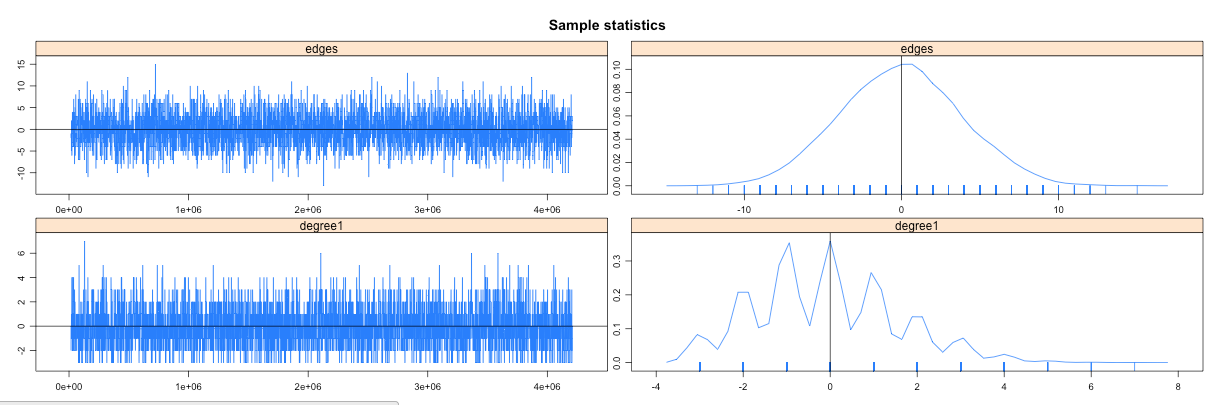
* **statnetWeb: Fit Model**
  + We want to go beyond the descriptive statistics and fit a model to the network
  + Our goal is to use key model statistics to fit an exponential-family random graph model (ERGM) to the observed data. We want this model to capture the underlying structure of the network.
  + What is an ERGM?
    - Exponential-family random graph models: a class of models for specifying the probability distribution for a set of random graphs or networks
      * : random variable for state of network (with realization )
      * : vector of model statistics for network
      * *θ*: vector of coefficients for statistics
      * : numerator summed over all possible networks (with same node set as y)
    - conditional log-odds:
      * is the change statistic, i.e. it records how changes if the tie is toggled on or off
    - (the coefficient for a model statistic) is the log-odds of an individual tie conditional on all others
  + Choosing terms
    - We can choose which model statistics to include in an ergm
      * These become elements of the vector
      * Each model statistic is represented by a term in the model formula
      * We include statistics that we hypothesize differ significantly in the observed network when compared to a simple random graph
    - Dyad dependent vs dyad independent
      * Presence or absence of a tie does/does not depend on the state of other ties
    - Using term documentation window
  + How does statnet find a model that fits?
    - MLE: maximum likelihood estimation
    - MCMC: only for dyadic dependent terms
  + Output from model fitting
    - Iterations – useful to see if model is degenerate
    - Model summary
      * Interpretation of coefficient estimates:
        + Conditional log-odds of two actors having a tie is

*θ*1 x change in stat1 + *θ*2 x change in stat2

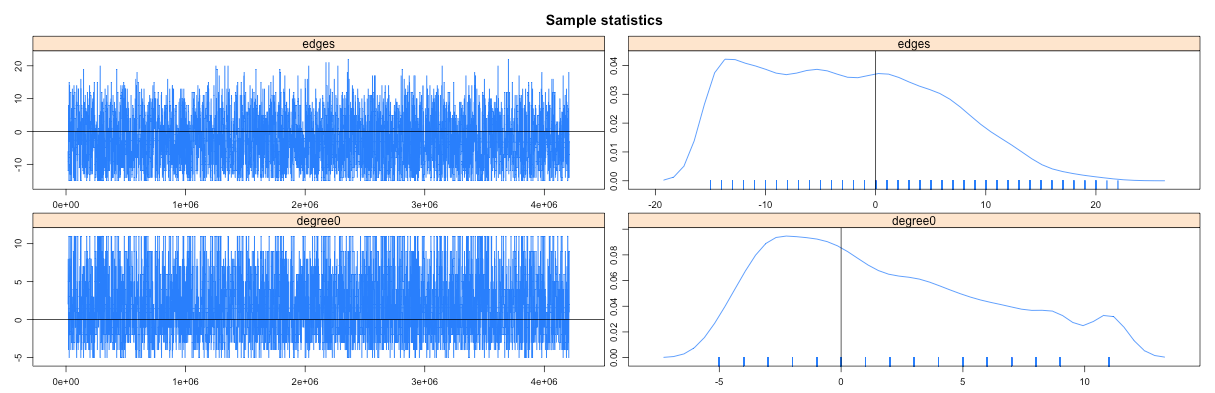
* + - * + Corresponding probability from equation above
      * Example & explanation of –Inf coefficients
        + statnetWeb shows the summary statistics as terms are added to formula
  + Save up to five models for comparison



* + When a model fails
    - Control parameters
      * Burn-in:
      * Interval:
      * Sample size:
      * If the user is familiar with other MCMC control parameters, they can be entered in the text box
* **statnetWeb: MCMC Diagnostics**
  + Only for models where MCMC was run
  + \*no longer used to ensure that mean statistics from model match the observed network stats
  + Want MCMC sample statistics to vary randomly around the observed values and the difference between the observed and simulated values of the sample statistics to have a roughly bell-shaped distribution, centered at 0.
    - e.g. flobusiness ~ edges+degree(1)
    - jagged curve in degree1 stats



* + - Interpreting other MCMC diagnostics
    - flobusiness ~ edges+degree(0)
      * flobusiness has 5 isolates, can’t have any fewer than -5 from target stat



* **statnetWeb: Goodness-of-fit**
  + Test how well your model fits the original data by choosing a network statistic that is not in the model, and comparing the value of this statistic observed in the original network to the distribution of values you get in simulated networks from your model.
  + When multiple models are saved, compare GOF plots in a chart.
* **statnetWeb: Simulate**
  + After creating an ergm and checking the diagnostics, we can simulate from it (take examples of networks drawn from this distribution). If the model is a good fit to the observed data, then networks drawn from this distribution will be more likely to “resemble” the observed data.
  + View network plot of each simulation; edit display options in the same way as the plot of the observed network.
  + View plot of simulation statistics compared to the target statistics

[image of auto-correlated statistics]

* + Edit MCMC controls (e.g. increase the interval if simulation statistics are highly auto-correlated.)

[image after increasing interval]

* + Download simulation statistics
* **Help**
  + GitHub Repository
  + statnet\_help listserve