Dissertation Proposal

Samantha C. Tyner

2016-09-18

Contents

1 Outline of the whole shebang

This section will be removed once I'm done writing this thing!

1.1 Lit Review (CH. 0)

1.1.1 ~~High Level Introduction~~

~~Social networks have been studied for decades, beginning with the seminal 1967 study, "The Small World Problem" by Stanley Milgram (Goldenberg et al. 2009). In recent years, the study of social networks has grown in popularity due to an increase in the availability of and easy of access to social network data. ~~

- ~~discuss the data format and how it is different from traditional data formats seen in statistics~~
- ~~bring up the different disciplines that have studied social networks~~
- ~~mention the models that exist for social network analysis~~
- ~~mention that I am focusing on one type of model class, SAOMs.~~

1.1.2 More Meaty Introduction to Network Analysis

In this section I will discuss in greater detail the models that are out there for network analysis. These include:

- ~~Classics:~~
  - ~~ERGMs~~
  - ~~Erdos-Renyi~~ and its variations
  - ~~Random graphs with fixed degree distribution~~
  - ~~Blockmodels and their relatives~~
  - ~~Latent space models~~
- ~~Social Networks:~~
  - ~~"p_1" models~~
  - ~~"p_2" models~~
- Dynamic:
  - ~~Preferential attachment~~
  - ~~Small-world~~
  - ~~Duplication-attachment~~
  - Discrete time MC models
  - Continuous time MC models (SAOMs a kind of CTMC)

1.1.3 Why focus on SAOMs

First, I will layout the field of network analysis formally using traditional statistical paradigms, as discussed in Kolaczyk (2009). Then I will discuss the lack of model checking tools specific to SAOMs due to model intractibility. This will be brief and will lead into the next section, a complete introduction to SAOMs and their structure/properties.

1.1.4 Full introduction and discussion of SAOMs

I will completely flesh out the form and theory of SAOMs. This will come primarily from the detailed document I created last year and anything else from Snijders' papers.

1.1.5 Lead-in to Visual Inference

Take the theory of SAOMs and lead into the discussion of why visual inference could be useful for model checking and other things for SAOMs.

1.1.6 Introduction to Visual Inference

Fully introduce the concept of visual inference as laid out in Buja et al. (2009). Also include discussion of Majumder et al.

1.1.7 Intro to visualizing networks

Discuss the many ways to visualize networks (also called network mapping). Ubiquitous view is 2D represetation of points and lines. Discuss the many layout algorithms. Discuss the many packages in R that can do this. Discuss the difficulty of using these package for people not intimately familiar with them. Lead into next section with discussion of ggplot2 and its importance and popularity in data visualization.

1.1.8 Lead-in to the full thesis.

Briefly outline the 3 chapters and tie them all together.

1.2 CH 1. Removing the Blindfold from SAOMs

- What does the distribution look like?
- What does an "average network" from an SAOM look like?
- Generation / Creation process video

1.3 CH 2. Using visual inference for SAOMs

Explain how I will use / have used visual inference to explore properties of SAOMs. Main research question: are statistically significant model differences also visually significant?

1.4 CH. 3: geomnet: a ggplot2 wrapper for network visualization

Explain the hole filled by the geomnet package.

1.5 CH. 4 Summary, Impact, Future Work

Brief walk-through of all topics already covered. Emphasize the holes filled by the work.

1.6 CH 5. Other Projects

Discuss the Bob Ross and Graphics Awards papers

2 Literature Review

2.1 Seeing the Forest Before Looking at Trees

Social networks have been studied for decades, beginning with a few foundational works, the most well known of which is the 1967 study, "The Small World Problem" by Stanley Milgram (Goldenberg et al. (2010)). But in recent years, the study of social networks has grown wildly in popularity due to an increase in the availability of and easy of access to social network data. The digital revolution has led to the creation of social media, linking people from all over the world in a way we never have been before. Now that platforms like Facebook, Twitter, and LinkedIn permeate our world, just about everyone knows what social networks are. In academic circles, collaboration networks are a type of social network that have been extensively studied and can even be a point of pride, like a mathematician's Erdös number (University (2016)). Social networks are a rich source of knowledge, but the data format does not fit easily within traditional data collection paradigms. Traditionally, data collection involves a set of units of the same, or at least similar, kind, on which observations are made. The storage of traditional data is simple and organized: rows contain variable values collected from units. These units can be people, plants, animals, stocks, objects, fields, and anything else under the sun, but one social network consits of many units, yet on the whole is just one observation. When observing a social network, one observes the possibly very numerous actors (also referred to as vertices or nodes) and the relationships (also referred to as edges or ties) between those actors. One can also collect information on the nodes and the edges separately, such as the age or gender of people and the length of their relationship or how strong it is in a friendship network. Thus, information on the entire network is more difficult to store than traditional data with which statisticians usually work.

This apparent difficulty has not stopped researchers in many different fields from studying social and other types of networks. Sociologists work with human relationship networks of all kinds imaginable, biologists work with protein-protein interaction networks, neurologists use fMRI scans to study biologic neural networks, and the list goes on. These disciplines worked separately for many years, each developing their own measures, softwares, and theories about the fundamental properties of networks. And although statisticians were comparatively late to the party, many statistical models exist for network analysis. Beginning with the classic Erdös-Rényi random graph model and varying in structure, complexity, and application to include longitudinal network data, such as continuous time markov chain models (Goldenberg et al. (2010)). The many varying models that exist just for social network analysis are impressive, but I focus my research on one type of continuous time markov chain (CTMC) models, called Stochastic Actor-Oriented Models (SAOMs). A full introduction to the various models that exist for social network analysis is presented in Section 4.1, and a full introduction to the structure and theory of SAOMs is presented in Section 2.3.

2.2 Statistical Models for Social Networks

The literature on statistical models for networks is extensive. In their thorough "Survey of Statistical Models", Goldenberg et al separate these models in to two primary classes: static and dynamic. I discuss the several types of models in each of these two categories after a brief introductory section on general network terminology and notation.

2.2.1 Basic Network Terminology and Notation

Formally, a network is a collection of nodes and the set of ties between them. Nodes are also referred to as vertices primarily in the graph theory literature or actors in the sociology literature, while ties are also called edges or relationships. In graph theory, a network is defined with respect to its nodes and edges, and a network G is equivalently written as $G(\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the collection of nodes, or nodeset, and $\mathcal{E}$ is the collection of edges, or edgeset. Typically, the nodes are numbered so that $\mathcal{N} = 1, 2, \ldots, n$, where n is the total number of nodes in the network. The edgeset $\mathcal{E}$ is usually described as a set of pairs, written as (i, j) or $i \mapsto j$ or simply ij, where $i \neq j \in \mathcal{N}$. In an undirected network, the ordering of i and j does not matter: there is no parent-child relationship, to use a term from graph theory, just a connection of some kind.

In a directed graph, however, the order does matter: the tie (i, j) is not equivalent to the tie (j, i). In an undirected graph, the number of possible edges is $\binom{n}{2}$, while in directed graphs it is $n(n-1)$, assuming no self-loops (also called self-ties or simply loops) and only allowing for at most one edge between any two nodes.

In statistical network analysis, a network is denoted by x if it is observed or by X if it is being treated as unobserved or as a random variable. Following this convention, edges in the networks x or X are denoted by x_{ij} or X_{ij}, respectively. If the edge $i \mapsto j$ is present, $x_{ij} = 1$, whereas $x_{ij} = 0$ if the edge is not present. If x is undirected, then $x_{ij} = x_{ji} \forall i \neq j \in \mathcal{N}$. If x is directed, then x_{ij} may equal x_{ji}, but this is not required and should not be assumed. Note that the definition of binary edge variables makes the assumption that edges are unweighted and that their cannot be more than one edge between two nodes. There are graphs and networks with weighted edges or with multiple ties between nodes, such as correlation networks used for modelling fMRI data or network-based epidemic modeling with finite, discrete state spaces (Kolaczyk (2009)). The models I discuss here, including the stochastic actor-oriented models that are my primary focus, are all for unweighted networks, though some allow for extension to weighted networks.

A network x can also be expressed as an $n \times n$ matrix of 0s and 1s called the adjacency matrix, denoted $\mathcal{A}(x)$. The ij^{th} entry of this matrix, a_{ij} is 1 if there is an edge between nodes i and j and 0 otherwise. Typically, in statstical network analysis, the diagonal entries of this matrix, a_{ii} are structurally 0, as self-ties or self-loops are not allowed or do not make sense.

2.2.2 Static Network Models

The Erdös-Rényi random graph model is widely regarded as the first random graph model (Goldenberg et al. (2010)). In this model, first introduced in 1959 in Erdős and Rényi (1959), a random, undirected graph or network, G, is described in terms of its nodeset, N, and its edgeset, E, where E is a random subset of the $\binom{|N|}{2}$ possible edges in the nodeset. The parameter in this model is p, the probability of an edge between any two nodes in N. The likelihood is written in terms of $|E|$ and p,

$$f_G(|E||p, N) = p^{|E|}(1-p)^{\binom{|N|}{2}-|E|}.$$

The properties and asymptotic behavior of this network model are well-established (Goldenberg et al. (2010)). Nodes in networks generated using this model will all have about the same degree, or number of incident edges, which is a very unrealistic property for networks to have. As such, many other models have been devised over the years as a way to better capture the the network creation process underlying real-world networks.

A set of models which has also been very extensively studied is the exponential random graph family of models (ERGMs). The first, less general form of these models is the p_1 model for social networks, first introduced in Holland and Leinhardt (1981). The p_1 model was developed for modelling directed networks, also called directed graphs or digraphs. In this model, the edges state, (x_{ij}, x_{ji}) between a pair of nodes, (i, j) for all $i \neq j \in N$, could exist in one of four possible states: $(0, 0)$ (no ties between i, j), $(1, 0)$ (a tie from i to j), $(0, 1)$ (a tie from j to i), or $(1, 1)$ (a tie from i to j and from j to i). Each one of these states has some probability such that the four state probabilities sum to 1 for each pair (i, j). These probabilities are described in terms of five kinds of parameters: θ, a base rate for edge creation; α_i, an effect for an outgoing edge with parent node i; β_j, an effect for an incoming edge with child node j; ρ_{ij}, an effect of reciprocated ties; and λ_{ij}, a normalizing constant to ensure that the four possible edge states of have probabilities summing to 1. Below, let $x_{ij} = 1$ when the edge from i to j exists and $x_{ij} = 0$ otherwise, and let $x_{ji} = 1$ when the edge from j to i exists and $x_{ji} = 0$ otherwise. Then, the edge state, (x_{ij}, x_{ji}) between nodes i andd j is modeled as:

$$\log f_X((x_{ij}, x_{ji})|\lambda_{ij}, \alpha_i, \alpha_j, \beta_i, \beta_j, \theta, \rho_{ij}) = \lambda_{ij} + x_{ij}(\alpha_i + \beta_j + \theta) + x_{ji}(\alpha_j + \beta_i + \theta) + x_{ij}x_{ji}\rho_{ij}$$

If each reciprocation parameter, ρ_{ij}, were unique to each edge, there would be a lack of identifiability in the model, which can be remedied by (i) not having a reciprocation effect ($\rho_{ij} = 0$ for all $i \neq j$), (ii) having a constant effect for reciprocation ($\rho_{ij} = \rho$ for all $i \neq j$), or (iii) having edge-dependent reciprocation ($\rho_{ij} = \rho + \rho_i + \rho_j$). Assuming scenario (ii), the log-likelihood function for a network X can be written in

exponential family form:

$$\log f_X(x|\boldsymbol{\lambda}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \rho, \theta) \propto \theta \sum_{i,j} x_{ij} + \sum_i x_{i+}\alpha_i + \sum_j x_{+j}\beta_j + \rho \sum_{i,j} x_{ij}x_{ji},$$

where the minimally sufficient statistics are x_{i+}, the outdegree of each node i, $\sum_j x_{+j}$, the indegree of each node j, and $\sum_{i,j} x_{ij}x_{ji}$, the number of reciprocal ties in the network. This p_1 set of models is problematic because, as "the number of $\{\alpha_i\}$ amd $\{\beta_j\}$ increase directly with the number of nodes, [so] we have no consistency results for the maximum likelihood estimation" (Goldenberg et al. (2010), p. 28). An extension of the p_1 model is the p_2 model, introduced by Duijn, Snijders, and Zijlstra (2004). This model is essentially a mixed-effects model version of the p_1 model, with node-level fixed effects for the outgoing edge effects $\boldsymbol{\alpha}$ and the incoming edge effects $\boldsymbol{\beta}$, and edge-level fixed effects for the edge rate effects $\boldsymbol{\theta}$ and reciprocity rates $\boldsymbol{\rho}$. The random effects, added to the covariate effects for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, have normal distribution with mean zero and variance parameters σ_α^2 and σ_β^2.

The final form for ERGMs is far more general than the p_1 and p_2, and is for undirected, rather than directed, networks. There are many more possible sufficient statistics other than the outdegree, indegree, and reciprocal ties. Other graph structures that are considered are the number of triangles, $T(X) = \sum_{i\neq j\neq h} x_{ij}x_{ih}x_{jh}$, and the number of k-stars, $S_k(X) = \sum_i \binom{x_{i+}}{k}$, where $k = 2$ is most commonly chosen. The form of the likelihood for X following this general type of ERGM is

$$f(X|\boldsymbol{\theta}, \tau) = \exp\left(\sum_k \theta_k S_k(X) + \tau T(X) + \psi(\theta, \tau)\right),$$

where θ_k and τ are parameters and $\psi(\theta, \tau)$ is the normalizing constant. A problem with this model arises when one considers the nested nature of the sufficient statistics. For example, an edge can be contained in a 2-star, which can be contained in a triangle. So, the sufficent statistics can be dependent. Despite this flaw, this type of ERGM has been studied extensively, and many methods for parameter estimation exist, for example in the `R` packages `statnet` and `sna` (, R Core Team (2016),Handcock et al. (2008), and Butts (2014)).

Another set of models includes extensions of the Erdös-Rényi (ER) random graph model. A natural way to extend the ER model is to vary the expected node degree of the graph. These models include the preferential attachment model or the small world models, which will be discussed further in 2.2.3. Another model type, the exchangeable random graph model of Airoldi (2006), adds weak dependence into the edge sampling procedure.

Another area of study is community detection or blockmodels. The goal of these models is to simplify the network into a small number of "hubs" in which the members of the same hub form ties much more often with each other than with members of different hubs. These "hubs" are usually referred to as blocks or communities. Two nodes are in the same block if they are found to be "structurally equivalent," meaning that they have similar connectivity with other nodes in the network (Goldenberg et al. (2010), p. 34). The model is defined as follows: given n nodes and K blocks, two nodes $i, j \in \mathcal{N}$ belong to the same block h, $h \in \{1, \ldots, K\}$ if their neighborhoods $\mathcal{C}_i \equiv \{k \in \mathcal{N} : x_{ik} = 1\}$ and $\mathcal{C}_j \equiv \{k \in \mathcal{N} : x_{kk} = 1\}$ are approximately equal. Approximately equal is vague because the metric used to compute the difference between two neighborhoods $\mathcal{C}_i, \mathcal{C}_j$ can vary depending on context. This is similar to clustering but is "a more general task than clustering" (Goldenberg et al. (2010)).

The last type static network model I'll present here is the latent space model. These models are called latent space models because they assumes that all nodes in the network can be expressed as a point in some k-dimensional space for "small" values of k (Hoff, Raftery, and Handcock (2002)). These models condition on the unknown locations of the nodes, $\mathbf{Z}_i \in \mathbb{R}^k$. Thus, the conditional probability model for the adjacency matrix Y, where $y_{ij} = 1$ for an edge between i, j and $y_{ij} = 0$ otherwise, can be written in the form

$$Pr(Y|\mathbf{Z}, \mathbf{X}, \boldsymbol{\Theta}) = \prod_{i\neq j} Pr(y_{ij}|\mathbf{Z}_i, \mathbf{Z}_j, \mathbf{X}_{ij}, \boldsymbol{\Theta}),$$

where $\mathbf{X}$ are covariates, $\mathbf{\Theta}$ are parameters, and $\mathbf{Z}$ is the matrix of node positions in $\mathbb{R}^k$. The individual edge probabilities y_{ij} are modeled in terms of $\mathbf{\Theta} \equiv (\alpha, \boldsymbol{\beta})$:

$$\text{logit}(Pr(y_{ij} = 1)) = \alpha + \boldsymbol{\beta}'\mathbf{X}_{ij} - |\mathbf{Z_i} - \mathbf{Z_i}| \equiv \eta_{ij}.$$

This leads to the log likelihood function,

$$\log(Pr(Y|\boldsymbol{\eta}) = \sum_{i \neq j} (\eta_{ij} Y_{ij} - \log(1 + e^{\eta_{ij}}).$$

This model can also be extended to include edge weights.

2.2.3 Dynamic Network Models

Dynamic network models are "the neglected sibling of static network" models (Goldenberg et al. (2010), p. 41). Dynamic networks, however, are extremely important because of how realistic they are. Social networks do not form spontaneously: they evolve over time. Ties can be added and deleted, and new nodes can join the network. Modeling the process of network changes over time is more complex but ultimately more useful if done correctly. Like with static models, the literature on dynamic network models begins with fairly straightforward random graph models that are extensions of the classic Erdös-Rényi model.

2.2.3.1 Quasi-Dynamic Models

A model is quasi-dynamic if it models a static network via an underlying dynamic process. The first is the quasi-dynamic preferential attachment model of Barabási and Albert (1999). Given n_0 nodes to start, at each time point t a new node is added with $n_t \leq n_0$ ties to the nodels already in the network. The n_t new ties are assigned proportionally based on the degree of each existing node. It is quasi-dynamic because it is usually used to model one scale-free network oberervation. The preferential attachment model is also referred to as the "rich-get-richer" model because it results in a network where there are a few nodes with very high degree.

Another quasi-dynamic model is the small-world model of D J Watts and Strogatz (1998). Given n nodes to start, each with k edges that form a ring lattice (nodes layed out in a circle and connected to their k closest neigbors), edges are randomly "rewired" with probability p. This results in networks with the small-world property: let L be the average distance between any two nodes in the graph, and if the graph has the small-world property, $L \propto \log(n)$ as n increases (D J Watts and Strogatz (1998)).

The last quasi-dynamic model is the duplication-attachment model originally studied in computer science theory in order to study the structure of the world wide web (Goldenberg et al. (2010)). This model is for directed, rather than undirected graphs like the previous two models. Generally speaking, this model is constructed as follows: start with a graph G with nodeset $\mathcal{N}$ and edgeset $\mathcal{E}$. At each time point t, one new node is added to G. This node is connected to a "prototype" node, call it m, that was selected at random from $\mathcal{N}$. In addition to this connection to node m (from m to the new node), there are d links added from the new node to other nodes in $\mathcal{N}$. Each new link is added randomly with probability α to one of the nodes in $\mathcal{N}$, selected with uniform probability. With probability $1 - \alpha$, the new link is directed to a node which is linked from m, say ℓ so that a path from m to ℓ through the new node. This model can be expressed in many different ways, but I do not present them here as they are not related to the dynamic model I have chosen to study.

2.2.3.2 Truly Dynamic Models

XXX Discuss discrete-time models (ERGM, latent space, DCFM) XXX

XXX Discuss continuous time markov models (lead in to next section) XXX

2.3 Stochastic Actor-Oriented Models for Longitudinal Social Networks.

The phrase Stochastic Actor-Oriented Model is quite a mouthful, but it contains most of the important information about the model. First, the model is changing in time in order to accomodate for observations from the same network made at different points in time. Second, it allows for changes in network structure due to actor-level covariates. These two properties are crucial to understanding networks as they exist naturally. Most social networks are ever-changing as relationships decay or grow, and most actors (or nodes) in social networks have inherent properties that could affect how they change their place within the network.

2.3.1 Terminology, Notation, and Mathematical Definition of SAOMs

A longitudinal network is a network consisting of the same set of n nodes that is changing over time, and is observed at M discrete time points, $t_1, \ldots, t_M$. Denote these network observations $x(t_1), \ldots, x(t_M)$. The SAOM assumes that this longitudinal network is embedded within a continuous time markov process (CTMP), call it $X(T)$. This process is almost entirely unobserved. The process $X(T)$ theoretically exists outside of the range of observation, but for simplicity of notation, assume that the beginning of the process, $X(0)$ is equivalent to the first observation $x(t_1)$, while the end of the process $X(\infty)$ is equivalent to the last observation $x(t_M)$. The observations $x(t_1), \ldots, x(t_M)$ are observed states of the process, $x(t_1) \equiv X(0), x(t_2) \equiv X(T_{t_2}), \ldots, x(t_{M-1}) \equiv X(T_{t_{M-1}}), x(t_M) \equiv X(\infty)$, but the time points t_m and T_{t_m} for $m = 2, \ldots M-1$ are not equivalent. The process $X(T)$ is a series of single tie changes, in which one actor at a time is given the opportunity to add or remove one outgoing tie. These opportunities for change can arise at a different rate for each actor, and the overall rate of change, the distribution of the waiting times that *any* actor will be given the opportunity to change is a function of all actors' rates. Additionally, once an actor is given the chance to change a tie, it tries to maximize a sort of utility function based on the current and potential future states of the network. These functions are described in detail in subsections 2.3.2 and 2.3.3.

2.3.2 The Rate Function

In the network x and for each actor i in this network, the rate function is most generally denoted $\lambda_i(\alpha, \rho, x, m)$, which dictates how quickly actor i gets opportunities to change one of its ties, x_{ij} in the time period $t_m \leq T < t_{m+1}$. In this function, α and ρ are parameters, x is the current network state at time m. Note that $x \in \mathcal{X}$, where $\mathcal{X}$ is the space of possible networks given the n nodes in the network, and that $|\mathcal{X}| = 2^{n(n-1)}$. We assume that the actors i are conditionally independent given their current ties, $x_{i1}, \ldots, x_{in}$. This assumption gives the rate function for the whole network as $\lambda(\alpha, \rho, x, m) = \sum_i \lambda_i(\alpha, \rho, x, m)$. For any time point, T, where $t_m \leq T < t_{m+1}$, the waiting time to the next change opportunity by actor i has distribution $Exponential(\lambda_i \alpha, \rho, x, m))$ in order to achieve the memorylessness property of a Markov process. Thus, the waiting time to the next change opportunity by *any* actor in the network has distribution $Exponential(\sum_i \lambda_i \alpha, \rho, x, m))$. There are many possibilities for the rate function, λ_i. The simplest is that it is constant over all actors and all unobserved timepoints between observations $x(t_{m-1})$ and $x(t_m)$, $\lambda_i(\alpha, \rho, x, m)) = \alpha_m$. The rate function can also depend on covariate values, call them $\mathbf{z}_i(t_m)$, of the actors or structural network elements such as outdegree, or both. For instance, assume $\lambda_i(\alpha, \rho, x, m)) = \lambda_{i1}\lambda_{i2}\lambda_{i3}$, where λ_{i1} is constant over m, λ_{i2} depends on the actor covariates, and λ_{i3} depends on a structural network property for node i. λ_{i2} might be written as $\lambda_{i2} = \exp\left(\sum_h \rho_h z_{ih}(t_m)\right)$, where there are $h = 1, \ldots, H$ actor covariates of interest, each with their own parameter ρ_h. λ_{i3} can be written as a function of the outdegree of node i, denoted x_{i+} with its own parameter α_{H+1}, so that, for example, $\lambda_{i3} = \frac{x_{i+}}{n-1}\exp(\alpha_{H+1}) + \left(1 - \frac{x_{i+}}{n-1}\right)\exp(-\alpha_{H+1})$. When $H = 0$, this form of λ_{i3} is equivalent to the model proposed by Wasserman (1980), which is one of the first models proposed for modeling dynamic networks as continuous-time Markov processes (Snijders (2001)). Given that a change occurs, the probability that actor i is given the power to change a tie is

$$\frac{\lambda_i(\alpha, \rho, x, m))}{\sum_i \lambda_i(\alpha, \rho, x, m))}$$

.

Table 1: Some of the possible effects to be included in the stochastic actor-oriented models in RSiena. There are many more possible effects, but we only consider a select few here. For a complete list, see the RSiena manual (@RSiena).

Structural Effects	
outdegree	$s_{i1}(x) = \sum_j x_{ij}$
reciprocity	$s_{i2}(x) = \sum_j x_{ij}x_{ji}$
transitive triplets	$s_{i3}(x) = \sum_{j,h} x_{ij}x_{jh}x_{ih}$
Covariate Effects	
covariate-alter	$s_{i4}(x) = \sum_j x_{ij}z_j$
covariate-ego	$s_{i5}(x) = z_i \sum_j x_{ij}$
same covariate	$s_{i6}(x) = \sum_j x_{ij}\mathbb{I}(z_i = z_j)$
jumping transitive triplets	$s_{i7}(x) = \sum_{j \neq h} x_{ij}x_{ih}x_{hj}\mathbb{I}(z_i = z_h \neq z_j)$

2.3.3 The Objective Function

Thanks to the conditional dependence assumptions in the model, we can consider the objective function for each node separately, since only one tie from one node is changing at a time. The objective function is written as $f_i(\boldsymbol{\beta}, x) = \sum_k \beta_k s_{ik}(x, \mathbf{Z}), x \in \mathcal{X}$ and $\mathbf{Z}$ the matrix of covariates. The vector $\boldsymbol{\beta}$ are the parameters of the model and x is any possible state of the network. Given the focal or ego node, i, there are n possible steps for the actor i to take: either one of all current ties x_{ij} will be destroyed, a new tie will be created, or no change will occur.

The parameters, β, are attached to various actor-level network statistics, $s_{ik}(x)$. There are always at least two parameters, β_1 for the outdegree of a node, and β_2 for the number of reciprocal ties held by a node (Snijders (2001) p. 371). There are many possible parameters β to add to the model. They can be split up into two groups: first, the structural effects, which only depend on the structure of the network. The inclusion of these effects has origin in the ERGMs discussed previously for static networks in section 2.2.2. These effects are written in terms of the edge variables x_{ij}, for $i \neq j$. The second set of effects are the actor-level or covariate effects. These effects also depend on the structure of the network. They are written in terms of x_{ij} but also in terms of the covariates, $\mathbf{Z}$. A table of some possible structural and covariate effects is given in **??**.

When node i is given the chance to change a node, we assume that they wish to maximize the value of their objective function $f_i(\boldsymbol{\beta}, x)$ plus a random element, $U_i(x)$, where the $U_i(x)$ are from "the type 1 extreme value distribution (or Gumbel distribution) with mean 0 and scale parameter 1" (Snijders (2001), p. 368). This distribution, which is also known as the log-Weibull distribution, has probability distribution function, using μ for the mean parameter and σ for the scale parameter, of

$$f(u|\mu, \sigma) = \frac{1}{\sigma} \exp\left\{ -\left(\frac{x-\mu}{\sigma} + e^{-\frac{x-\mu}{\sigma}} \right) \right\}.$$

Using this distribution is convenient because it allows the probablity the actor i chooses to change its tie to actor j in terms of the objective function alone. Let $p_{ij}(\boldsymbol{\beta}, x)$ be this probability. Next, write the network x in its potential future state, where the tie x_{ij} has changed to $1 - x_{ij}$, as $x(i \rightsquigarrow j)$. Then, the probility that the tie x_{ij} changes is

$$p_{ij}(\boldsymbol{\beta}, x) = \frac{\exp\{f_i(\boldsymbol{\beta}, x(i \rightsquigarrow j))\}}{\sum_{h \neq i} \exp\{f_i(\boldsymbol{\beta}, x(i \rightsquigarrow h))\}}$$

2.3.4 A SAOM as a CTMP

In order to fit this model definition back into the original context of the CTMP described in section 2.2.3.2, it must be written in terms of its intensity matrix, $\mathbf{Q}$. This matrix describes the rate of change between states

of the process. For networks, there are a very large number of possible states, $2^{n(n-1)}$, so the intensity matrix is a square matrix of that dimension. But, thanks to the property of SAOM that the states are allowed to change only one tie at a time, there are only n possible states given the current state, $n-1$ of which are uniquely determined by the node i that is given the opportunity to change. Thus, the intensity matrix $\mathbf{Q}$ is very sparse, with only $n(n-1)+1$ non-zero entries in each row. Note that $n(n-1)$ of these represent the possible states that are one edge different from a given state, and the additional non-zero entry is for the state to remain the same. All other entries in a row are zero because those column states cannot be reached from the row state by just one change as dictated by the SAOM. The entries of $\mathbf{Q}$ are defined as follows: let $b \neq c \in \{1, 2, \ldots, 2^{n(n-1)}\}$ be indices of two different possible states of the network, $x^b, x^c \in \mathcal{X}$. Then the bc^{th} entry of Q is:

$$q(x^b, x^c) = \begin{cases} \lambda_i(\alpha, \rho, x^b, m) p_{ij}(\boldsymbol{\beta}, x^b) & \text{if } x^c \in \{x^b(i \rightsquigarrow j) | \text{ any } i \neq j \in \mathcal{N}\} \\ 0 & \text{if } x^c \text{ differs from } x^b \text{ by more than 1 tie} \in \mathcal{N} \\ -\sum_{i \neq j} \lambda_i(\alpha, \rho, x^b, m) p_i j(\boldsymbol{\beta}, x^y) & \text{if } x^b = x^c \end{cases}$$

Thus, the rate of change between any two states that differ by only one tie, x_{ij}, is the product of the rate at which actor i gets to change a tie and the probability that the tie that will change is the tie to node j.[1] Furthermore, the theory of continuous time Markov chains gives that the matrix of transition probabilities between observation times t_{m-1} and t_m is dependent only on the difference between timepoints, $t_m - t_{m-1}$ and is equal to $e^{(t_m - t_{m-1})\mathbf{Q}}$, where $\mathbf{Q}$ is the matrix defined above and e^X for a real or complex square matrix X is equal to $\sum_{k=0}^{\infty} \frac{1}{k!} X^k$.

2.4 Model Fitting and Diagnostics for SAOMs

2.5 What is Visual Inference?

2.6 Network Visualization

2.6.1 Layout Algorithms

2.6.2 R Packages

2.6.3 The Importance of `ggplot2`

2.7 Lead-in to Thesis

3 Stochastic Actor-Oriented Models for Longitudinal Network Data: Removing the Blindfold

As this is the least-developed section of my thesis, this section will describe the work I intend to do in as much detail as possible.

3.1 Introduction

The introduction to this chapter will primarily serve as a brief introduction to the structure of stochastic actor-oriented models (SAOMs) for longitudinal network data. As there is a greater exposition of these models in the literature review, I do not repeat that information here at this time.

[1] Just to be clear, the change is from x_{ij}^b to $x_{ij}^c = 1 - x_{ij}^b$.

3.2 Visualizing the Dynamic Network Process

The basis for SAOMs is a continuous-time Markov chain (CTMC) that remains entirely unobserved with the exception of the finite number of network observations, $T \geq 2$. The first part of this chapter will discuss a to-be-developed procedure to visualize the intermediary steps in the Markov chain that lead to the network transformation from one time point to another. This video (or interactive app ?) will clearly demonstrate the multilevel process the SAOMs are modelling. First, it will show that a node is selected and given the opportunity to change a tie, and then it will show how that node's objective function decides which tie to change, or to not change at all.

3.3 Characterizing a SAOM

My hope is that this demonstration will also raise some important questions in the minds of researchers working with these social network models and others. Primarily, I hope to bring question of the model *distribution* to the forefront. The process being modeled is random, and as such has a theoretical underlying distribution. What does that distribution look like? This is an open question in statistical network analysis. In statistics, the entity being modeled is assumed to have an underlying distribution according to any level complexity of model. Even in situations of analytical intractability, data can be simulated in some way or another in order to give the researcher an idea of what the distribution of the model looks like. These distributions can be viewed with histograms or barcharts, or with contour plots, heat maps, or 3D plots for higher dimensional data. But none of these statistical graphics are appropriate for most network models, including SAOMs. Thus, I will aim to create a way to, given various parameter values, visualize the distribution of a SAOM. One area which I will explore to accomplish this task is principal component analysis for graph data.

Once a distribution is characterized and viewable, the next things people notice are what the average value from this distribution looks like and how spread out the distribution is. Therefore, my next task will be to come up with a way to compute and view the "average" network from a given SAOM, as well as some measure of variability so that researchers can more easily determine whether or not an observed network or set of networks could possibly have come from a given model.

4 Visual Inference for Networks

4.1 Models

We fit three different stochastic actor-oriented models to a subset of the 50 actor dataset from the "Teenage Friends and Lifestyle Study" that is provided on the RSiena webpage "'friendsdata. We chose to subset the data to decrease the cognitive load on our experiment's subjects. The subset contained actors 20 through 35 and the ties between them, as well as the drinking behavior of each actor at each of the three waves. This specific subset was chosen because it showed somewhat higher connectivity than other subsets, as we've emphasized in the visualizations of the three network adjacency matrices below. For model fitting, we condition on wave 1 and estimate the parameters of our models from the second and third waves.

The first wave of the network, which is conditioned on in estimation, is given in Figure 2.

The two models we fit are a "null model" (model M_1) and two "alternative models" (model M_2 and M_3). The null model only includes the default parameters that are included in the RSiena estimation: the reciprocity and outdegree parameters. The alternative model M_2 includes one additional covariate parameter that was determined to be significant by the Wald test in the RSienaTest package, while the alternative model M_3 includes one additional structural parameter whose significance was determined in the same way. Details on these effects are given in Table 2.

After estimating the parameters in each of these models, we simulate from them to obtain realizations of each of the models. The objective function for each actor in each model is given below.

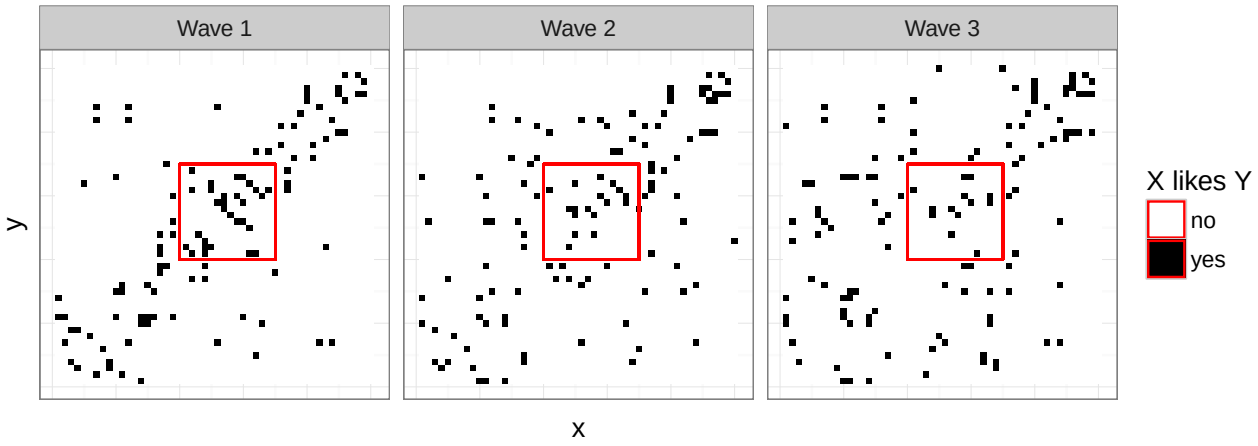


Figure 1: Adjacency matrices describing friendship relations between 50 students in waves 1,2, and 3 of the friendship study @friendsdata. The red squares identify the subset we focus on for our experiment.

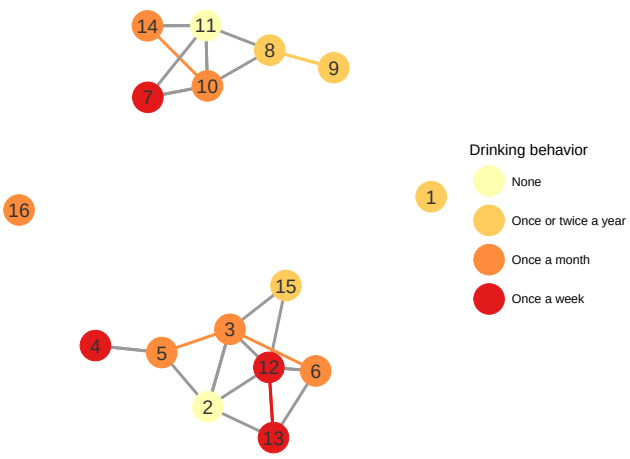


Figure 2: Network of friendships of wave 1 of the subset of students that we will be exploring.

Table 2: Parameters and estimates of models M_1, M_2, and M_3. Estimates are the mean of 1000 iterations of the model estimates. The lineups that follow are simulated from models using these values.

Effect name	Parameter	Corresponding Statistic	M_1	M_2	M_3
Rate 1 (wave 1 $\rightarrow$ 2)	α_1	$\sum_{i,j=1 i\neq j}^{n} (x_{ij}(t_2) - x_{ij}(t_1))^2$	4.66	5.18	4.71
Rate 2 (wave 2 $\rightarrow$ 3)	α_2	$\sum_{i,j=1 i\neq j}^{n} (x_{ij}(t_3) - x_{ij}(t_2))^2$	1.93	2.02	1.98
Outdegree	β_1	$s_{i1}(x) = \sum_{j=1}^{n} x_{ij}$	-3.6	-4.1	-3.59
Reciprocity	β_2	$s_{i2}(x) = \sum_{j=1}^{n} x_{ij}x_{ji}$	4.15	4.28	4.23
Jumping Transitive Triplets	β_3	$s_{i3}(x) = \sum_{\forall j\neq h} x_{ij}x_{ih}x_{hj}\mathbb{I}(v_i = v_h \neq v_j)$	–	3.21	–
# doubly achieved distances	β_4	$s_{i4}(x) = \lvert\{j : x_{ij} = 0, \sum_h x_{ih}x_{hj} \geq 2\}\rvert$	–	–	-7.58

$$
\begin{aligned}
f_i^{M_1}(x) &= \hat{\beta}_1^{M_1} s_{i1}(x) + \hat{\beta}_2^{M_1} s_{i2}(x) \\
f_i^{M_2}(x) &= \hat{\beta}_1^{M_2} s_{i1}(x) + \hat{\beta}_2^{M_2} s_{i2}(x) + \hat{\beta}_3^{M_2} s_{i3}(x) \\
f_i^{M_3}(x) &= \hat{\beta}_1^{M_3} s_{i1}(x) + \hat{\beta}_2^{M_3} s_{i2}(x) + \hat{\beta}_4^{M_3} s_{i4}(x)
\end{aligned}
$$

The rate parameters, α_1 and α_2 represent how many opportunities for change each actor gets when moving from wave 1 to 2 and from wave 2 to 3, respectively. The outdegree parameter, β_1, represents how likely an actor is to change outgoing ties. If the estimate, $\hat{\beta}_1$, is positive, the actor is more likely to create outgoing ties, while a negative estimate leads the actor to deleting outgoing ties. This effect is highly correlated with the reciprocity parameter, β_2. A negative estimate of this parameter implies that the actor is discouraged from reciprocating its incoming ties, while a positive estimate implies that the actor is encouraged to reciprocate all ties. The additional parameter in M_2, β_3, is a covariate parameter. The covariate in this model is the drinking behaviour of the 16 students in our data. The possible values are 1, 2, 3, and 4, which means the student drinks never, once or twice a year, once a month, or once a week. This jumping transitive triplet effect impacts the transitive closure of actors from different groups. Thus, a positive estimate encourages transitive closure when one of three actors is in a different covariate group than the other two, while a negative estimate discourages this behavior. An example of this type of closure is given in Figure 4. With the directed edges we also distinguish between 'i likes j' and 'j likes i'. Finally, the doubly achieved distances effect is defined by the number of actors to whom actor i is not directly tied, and tied through two paths via at least two intermediaries. This is a structural effect, like the density and reciprocity effects. A positive coefficient value encourages indirect ties, while a negative value discourages the formation of indirect ties.

The parameters in the objective function are tested for significance using t-tests. The test statistic is the ratio of the parameter estimate to its standard error. If this value is larger than 2 in absolute value, then the parameter is said to be significant at the $\alpha = 0.05$ level and should be included in the model. This is a fairly simple statistical test, so we wanted to test whether this significance can be detected visually just as simply as the statistical test detects it. If visualizations of simulated networks from two nested models have a much different appearance when placed side-by-side, then the difference in appearance can be attributed to the additional parameter in one. If, however, there is no visually detectable difference, then the additional parameter does not appear to have changed the network structure all that much. Because model selection and diagnostics for network models are less developed areas of the theory, testing network parameters in this visual way could lead to additional methods of model selection for networks.

4.2 Lineup Simulation

Needs more specifics: how many lineups were created for each model?

At the moment we are just considering models M2 and M3 for an inclusion in the lineup study. We are trying to nail down what to include and what not to include in the experiment.

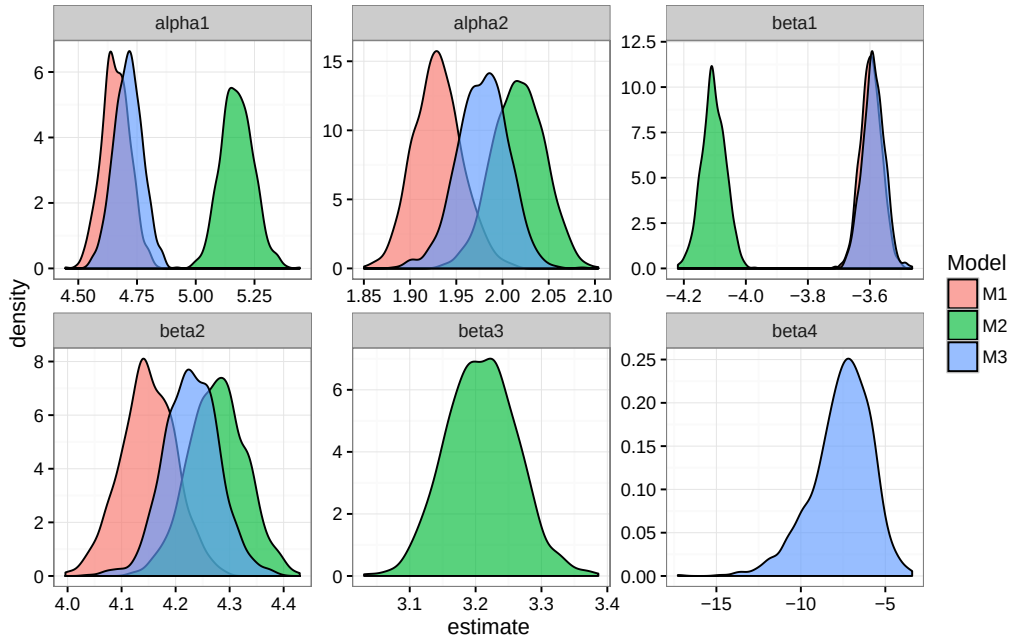


Figure 3: Histogram of the distribution of model parameters based on 1,000 simulation runs. Model parameter β_3 for jumping transitive triplets in model M_2 is significantly different from zero, but its inclusion also leads to significant changes in the other model parameters of model M_1. The parameter β_4 for doubly acheived distances is also significantly different from zero, but has larger variance. The inclusion of β_4 also changes the estimates of the other model parameters, but not as much as the inclusion of β_3.

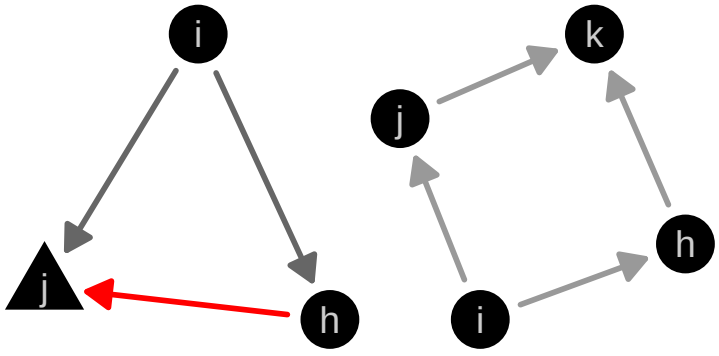


Figure 4: Structural network effects. On the left, a jumping transitive triplet (JTT). On the right, a doubly achieved distance between i and k. At left, a realization of a jumping transitive triplet, where i is the focal actor, j is the target actor, and h is the intermediary. The group of the actors is represented by the shape of the node. At right, doubly achieved distance between actors i and k.

To create the lineups that will be used in our experiment, we used the values given in Table 2 as starting values in simulation. For each lineup simulated, the same default RSiena algorithm was used as was used to generate the fitted models with the exception that the algorithm only simulated from the given set of parameters. Each lineup and plot within lineup was generated idependent of all the others.

4.3 Parameter Estimation from Lineups

After the lineups were created, we re-fit each of our three models to each graph in each lineup.

Questions that should be answered in this section:

1. Why do we want to get the parameters for each model? We fit all three models to every plot in the lineups. Fitting all models to all lineup plots allows us to gauge tha ability of the parameter estimates to provide a measure of lineup identification difficulty. XXX ? not quite sure ? XXX For instance, when comparing models 1 and 2 in a lineup, the estimates from fitting model 2 to plots which were simulated from model 2 should be significantly different from the model 2 estimates from plots simulated from model
2. Convergence issues

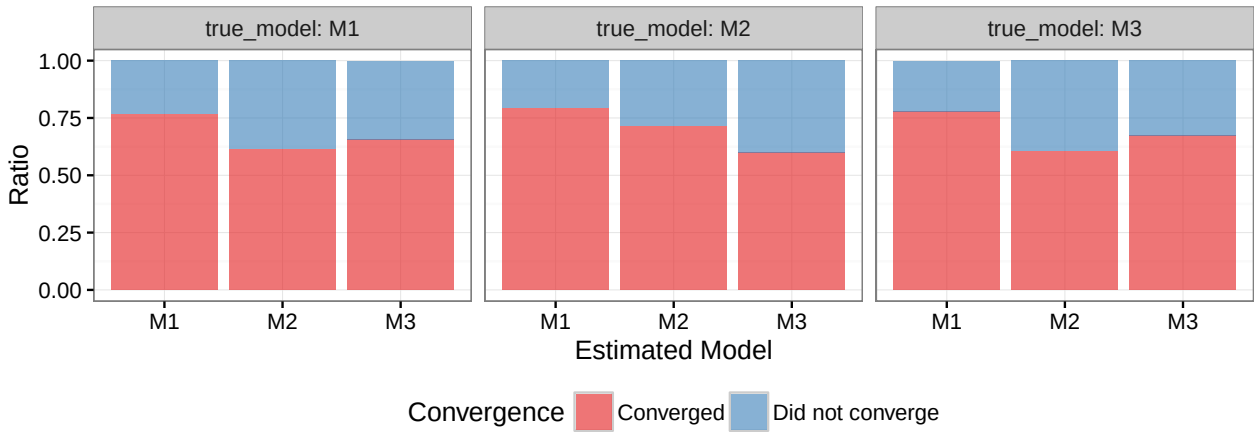


Figure 5: Pattern of convergence. A total of 67.7% of all lineup data converged in 5,000 iterations. The simplest model, M_1 has the highest rate of convergence. For models M_2 and M_3 data generated from the model converged at a higher rate than data generated from the other model.

1. The smaller the difference between these estimates, the harder it should be to identify the different model in the lineup. XXX IDEA: should we consider some sort of distance metric comparing all estimates from the models at once in addition to the differences in the β_3/β_4 values? XXX}
2. How is the fitting done, exactly? XXX get into nitty gritty from RSiena model XXX

Because the likelihood function for these complicated models is intractable, RSiena implements a Monte Carlo simulation to obtain method of moments estimates of the parameter values. This fitting procedure was first introduced in Snijders (1996).

The model fitting in RSiena is done in three phases. Briefly, in the initial phase, sensitivity of the statistics to the parameter values is determined, then in the second phase, parameter values are fit iteratively. Finally, in the third phase, networks are simulated from the fitted models and the model is checked for convergence.

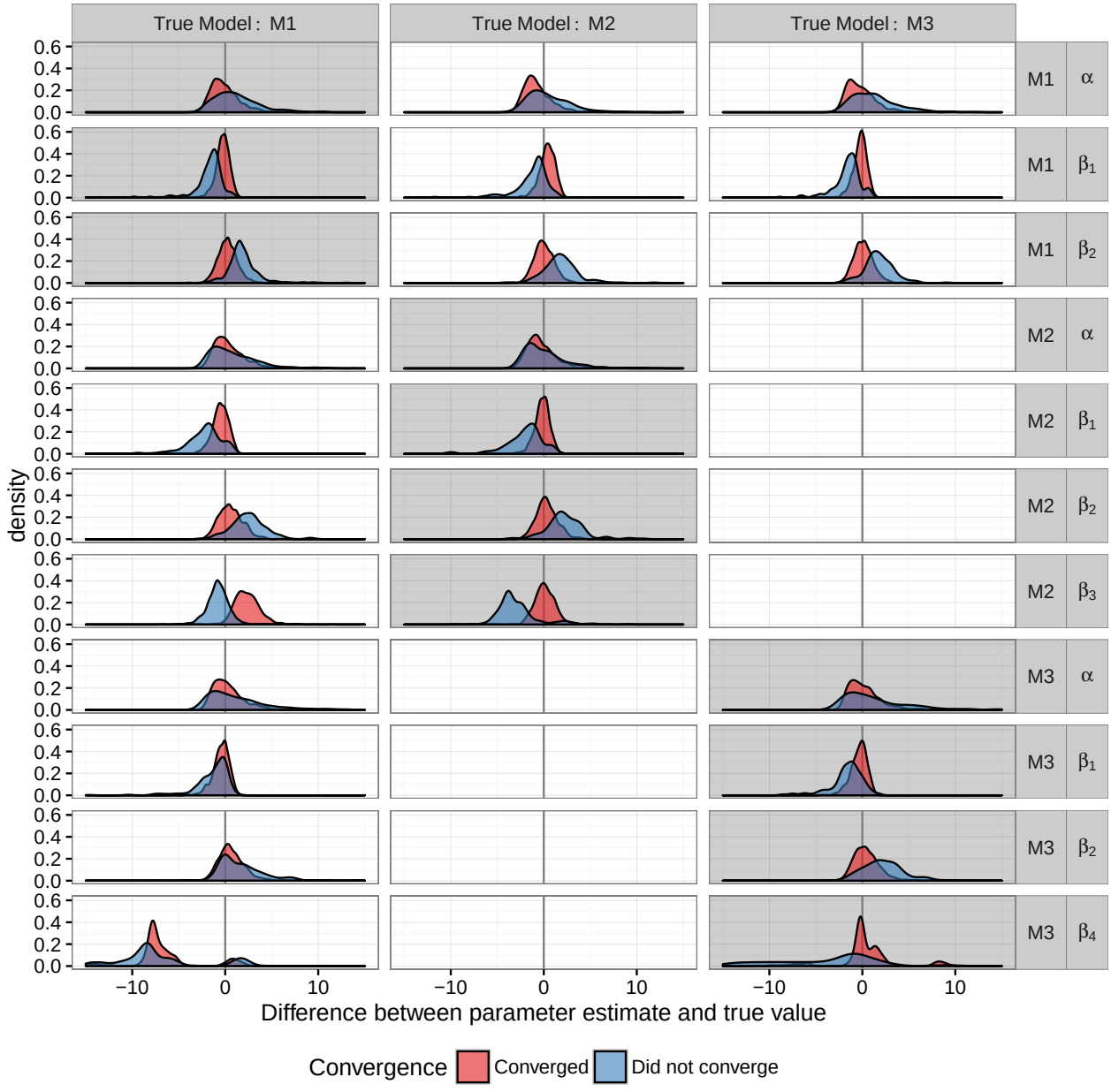


Figure 6: Difference between parameter estimate and true value. Panels with a light grey background show model fits with data sampled from the same model. Densities are drawn for both converged and non-converged data. The red-filled densities should have a mode near zero, idicating that the model converged toward the correct value. For data from model M_1, estimates for parameters β_3 and β_4 converge to a wrong value.

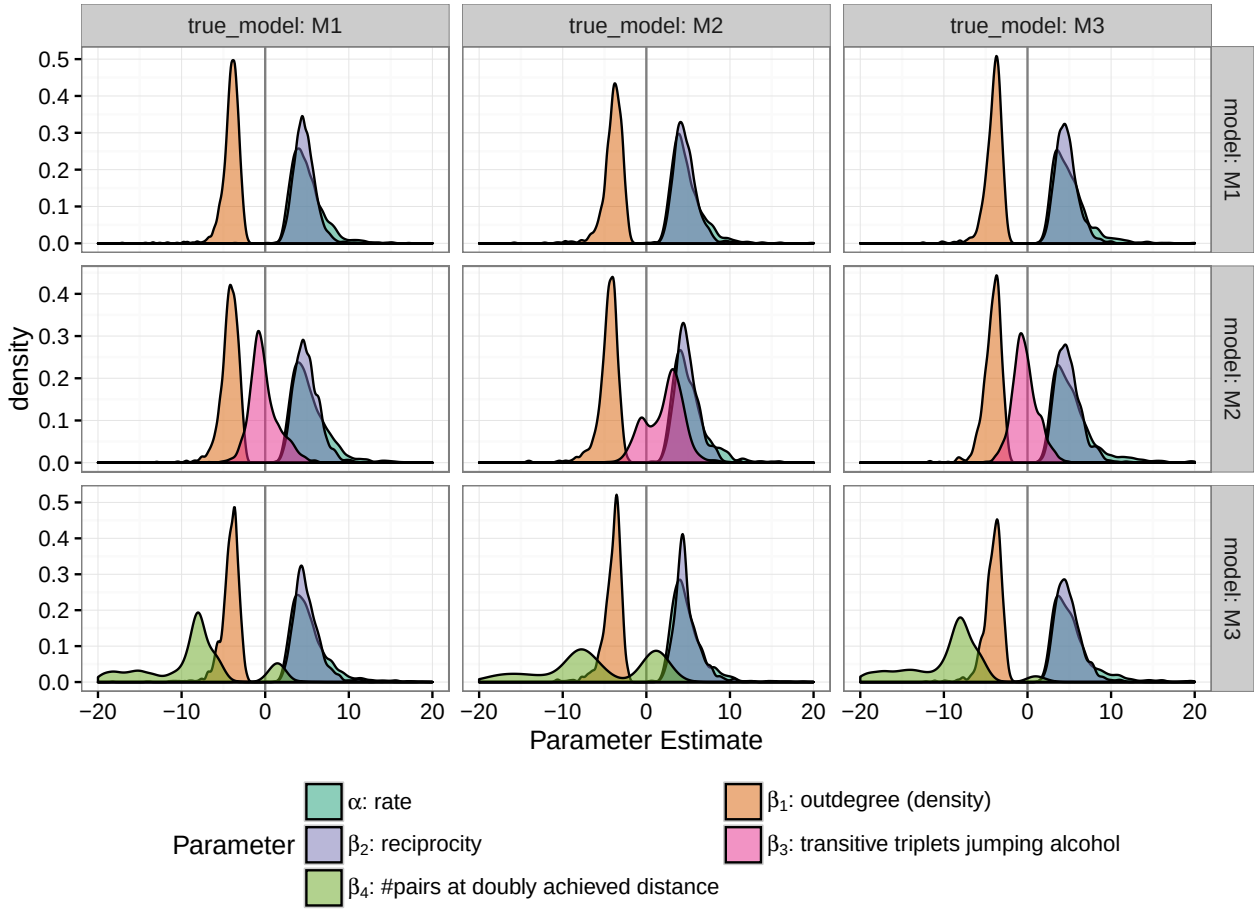


Figure 7: Comparison of model estimates under all three models under investigation.

4. What parameters are in the output? In RSiena, the convergence of a non-rate parameter is determined through the simulated values from Phase 3 of the SienaFit algorithm. XXX more detail will be above later XXX The simulations are compared to the observed values of the statistics observed in the data. The values from the simulations should be fairly close to the observed values, but because the fitting is done stochastically, deviations from statistics will not be exactly zero. Checking for convergence is based on a t-ratio of the average of these deviations to the standard deviation of these deviations. RSiena also performs an overall maximum convergence check by finding the maximum t-ratio value for any linear combination of the observed statistics. According to the RSiena Manual, `convergence is excellent when the overall maximum convergence ratio is less than 0.2", and for the non-rate parameters, the threshold for`reasonable" convergence is set at 0.3 with excellent convergence when the t-ratio is less than or equal to 0.1 in absolute value. (Ripley, Boitmanis, and Snijders (2013)).

5. What are the results?

Figure 7 shows an overview of model estimates for each simulated data set. What we expect to see is that estimates do not change values (much) if they are estimated under a model different from the one they are generated from. This is true for all data sets estimated under model M_1 independently of which model they were generated from (top row of Figure 7. For data fit under model M_2 we see that parameter β_3 (jumping transitive triplets) are estimated to be about zero, if the data is generated from models M_1 and M_3. For model M_2, parameter β_3 is estimated to be significantly different from zero in 53% of cases. This coincides with our expectation.

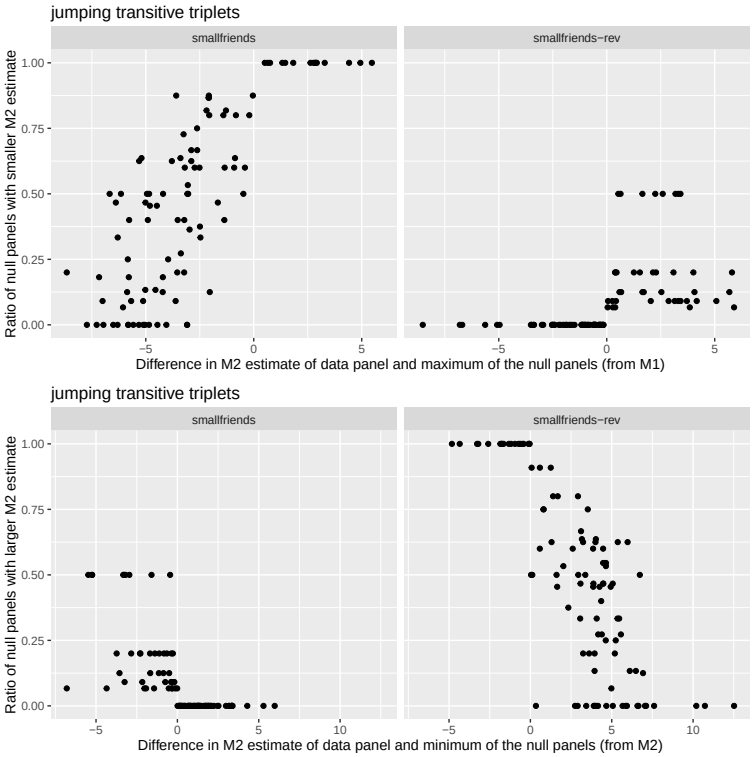
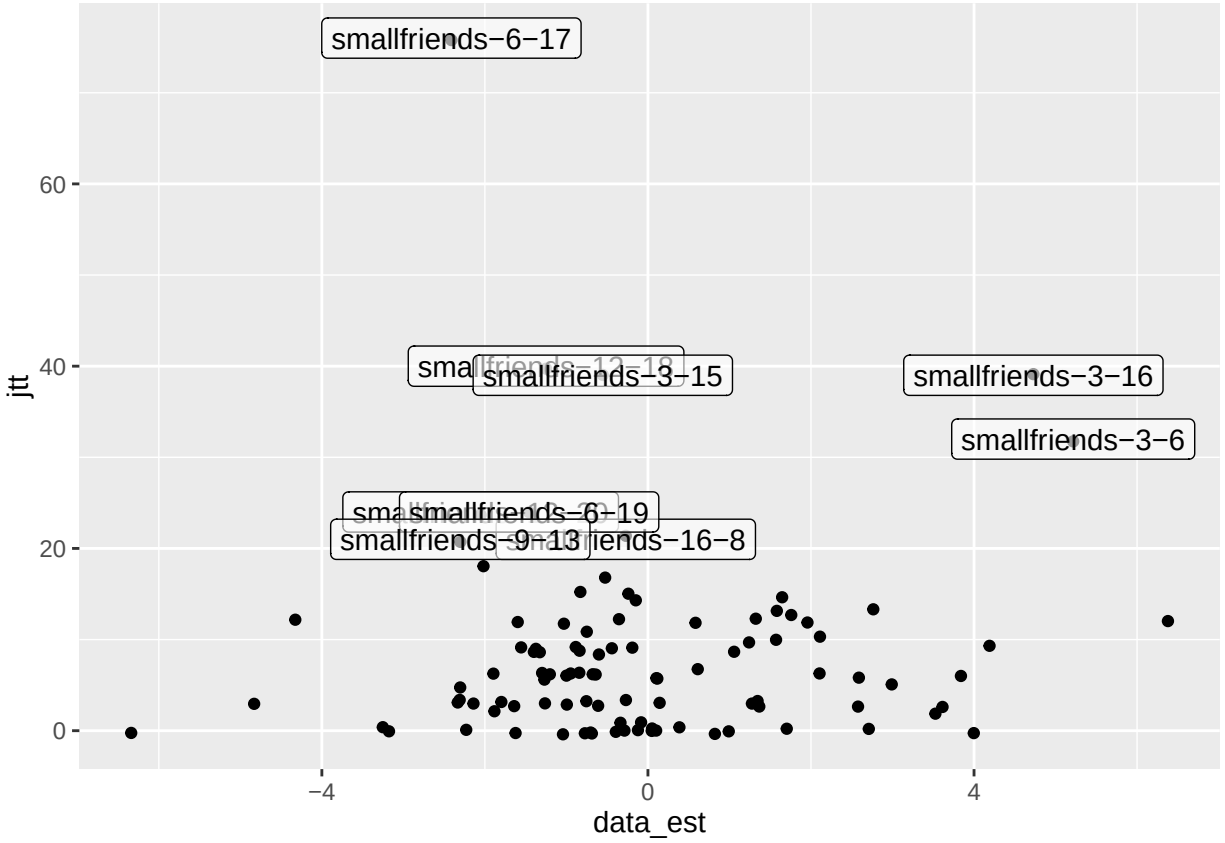
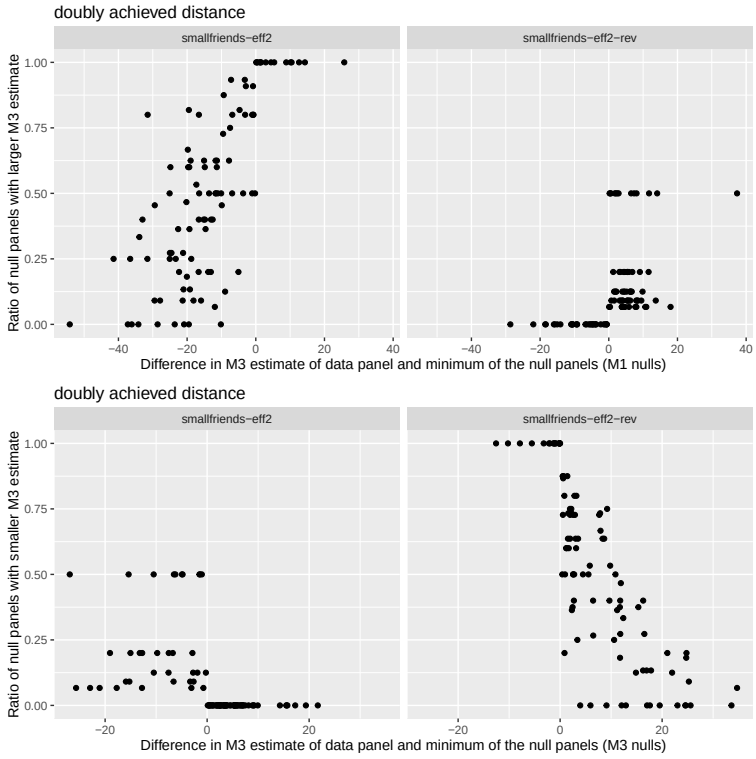


Figure 8: Scatterplots of smallfriends and smallfriends-rev: comparing the ratio of null plots with smaller estimates of β_3 in the nulls than in the data panel (top row) and larger estimates of β_3 in the nulls than in the data panel. In lineups with a ratio of 1 the data should be 'easy' to identify in both scenarios.

However, the bottom row of Figure 7 shows that independently of which model data is generated from, β_4, the number of pairs at doubly achieved distance, is estimated to be significantly different from zero in a large number of cases (about half of the data generated from model M_2 and more than that in model M_1). While β_4 is a highly significant parameter in model M_3, this questions the way that parameters are fitted and tells us that we are not likely to be able to visually distinguish between data generated from model M_3 and data generated from models M_1 and M_2. We might, however, be able to distinguish between data sets for which β_4 is estimated to be significantly different from zero and non-significant ones. XXX can we see differences in the pilot study?





6. How do the results influence our decision for the lineups?

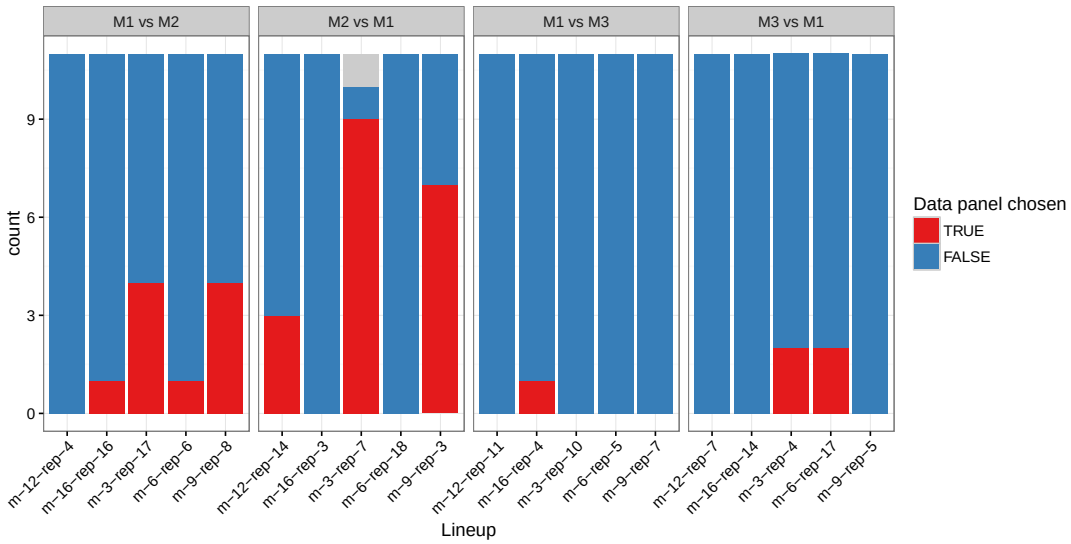


Figure 9: Barchart summarizing the number of responses from the pilot study by model and lineup. Color shows the number of times the data panel was chosen from the lineup. Clearly, the first two sets of lineups (M_1 vs M_2 and M_2 vs M_1) have on average higher number of data identifications.

The strong influence of the inclusion of β_4 in model 3 has made any comparison between model 1 and model 3 impossible. The β_4 estimate is always significantly greater than zero in fitted models that converged. Thus, β_4 should have been included from the beginning. % at end of param est we know that model m3 is useless. m3 messes with structure of m1. should have been in cluded in all of the models. nice & important but doesn't help for lineup study

% results from the the pilot study. how do estimates combine with pilot study ? have 10 non-m3 ones that are good. also want to see that m1 v m3 is way worse than m1 v m2. now have a reason for m1 v m3 is so bad. m1 v m3 ids should be NO GOOD.

4.4 Results from the pilot study

Big goal: Can we derive measures from the model estimates or the visual representation (ie. the network structure) that help us determine which lineups are more difficult than others, i.e. based on these estimates, how relaibly can we predict which panel will be picked from a lineup? XXX For that, we could also calculate the number of JTTs in each network - use `jtt` for that.

Littler goal: evaluate pilot study and see whether the results line up with the conjectures made in the previous section.

The pilot study consisted of an evaluation of a set of 20 lineups by 11 volunteers. For each of the model situations (M_1 vs M_2, M_2 vs M_1, M_1 vs M_3 and M_3 vs M_1) one lineup of size $m = 3, 6, 9, 12$ was used. Overall, the number of data identifications in the lineups was very low (34 out of 220 evaluations). Participants identified the data plot in two to four of the lineups they evaluated. The mode was three data identifications out of twenty per participant.

4.4.0.1 Suitability of M_3 in lineups:

Figure 9 shows barcharts of responses from the pilot study detailing the number of data identifications in each lineup. It is more difficult to identify the data plot in lineups of graphs based on data from models M_1 and M_3 than in lineups of graphs based on data from models M_1 and M_2.

4.5 Amazon Turk Experiment

4.5.1 Methods

In order to test our hypothesis, we set up an Amazon Mechanical Turk experiment (Amazon (2010)) using the lineup protocal of Buja et al. (2009). We presented four types of lineups: M1 v. M2, M2 v. M1, M1 v. M3, and M3 v. M1, where M1, M2, and M3 have the objective functions given in ~2. We created a total of 25 lineups to show to Amazon Turk users taking our experiment: 10 for M1 v. M2 and 5 each for the other three types of lineups. In each lineup, there were 12 plots shown: 11 of the plots were simulated from the first model (the "null" model) and 1 was simulated from the second model (the "alternative" model). We chose to show lineups of size 12 because we felt that showing more than 12 would be too large of a cognitive load, while showing fewer than 12 would leave too much of the experiment to random chance. The order of the plots within the lineups was randomly assigned. We selected five lineups presented from each group based on the following criteria: first, for M1 v M2 and M1 v M3, we selected the lineups where the additional parameter in the objective function, β_3 and β_4, respectively, had the largest estimated value among the 12 networks presented. There were not many of these in the size 12 lineups, so our other selection criteria was that the estimated parameter value was larger in the alternative panel than in at least half of the other lineups and it was close to the estimate from the panel that did have the largest value. For the M2 v were chosen where the smallest parameter estimate belonged to the alternative model. If there more lineups were needed, we next selected those which had estimates smaller than at least half of the other panels and were close to the minimum estimate in the lineup. For the remaining five plots of type M1 v M2, we chose the lineups where the alternative plot had the largest number of jumping transitive triplets appear in the network. We chose this statistic because its value has great effect on both the estimation of the β_3 parameter and on the visual appearance of the plot.

In order to become a subject in our experiment, the Amazon turk user had to first read through some introductory material and prove they could identify the correct plot in two test lineups, one for M1 v. M2 and one for M2 v. M1. These two lineups were constructed to be very simple in order to train the turkers. Once the turker made it into the experiment, they looked at 10 lineups selected at random from a pool of 25. There were five lineups for each of the four types with an additional five lineups of the type M1 v. M2 which were chosen for their high counts of jumping transitive triplets in the alternative model network in an attempt to gauge how important this statistic is to the visualization of the network. If there are many jumping transitive triplets in the alternative model plot and more turkers correctly choose that plot, that would be evidence that the jumping transitive triplets are very noticeable to the user.

4.5.2 Procedure

Each Amazon Turk user was greeted with the following message: "In this survey a series of similar looking charts will be presented. We would like you to respond to the following questions.

1. Pick the plot based on the survey question
2. Provide reasons for choice
3. How certain are you?

Finally we would like to collect some information about you. (age category, education and gender)." Then, they were led through a training page about how to identify the correct plot in a lineup, seen in Figure 10. Then, they saw two trial plots that they had to get correct in order to proceed to the rest of the experiment. They chose the plot that looked the most different from the others, why they chose is (most complex, least complex, or other), and how certain they were that they had chosen correctly (Very Uncertain, Uncertain, Neutral, Certain, Very Certain). These trial plots are shown in Figure 11

Once the turk user chose the correct plot in the two trial plots, the experiment proceeded with the same interface and users selecting which plot they thought was most different, why they thought that, and how certain they were for 10 plots chosen at random.

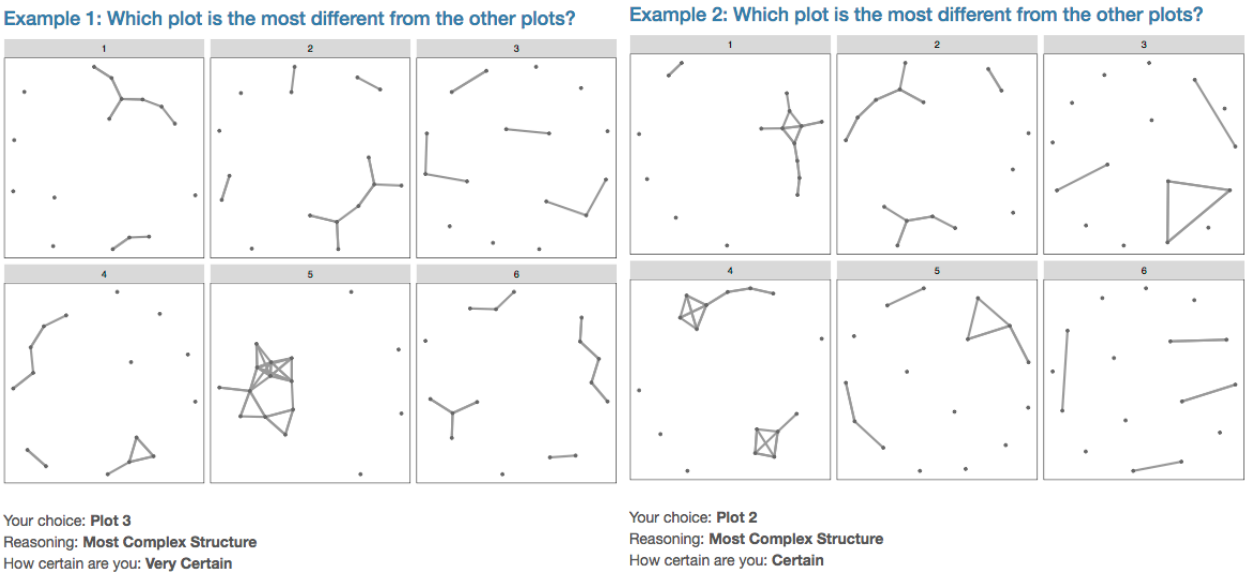


Figure 10: The first training page in the Amazon Turk experiment.

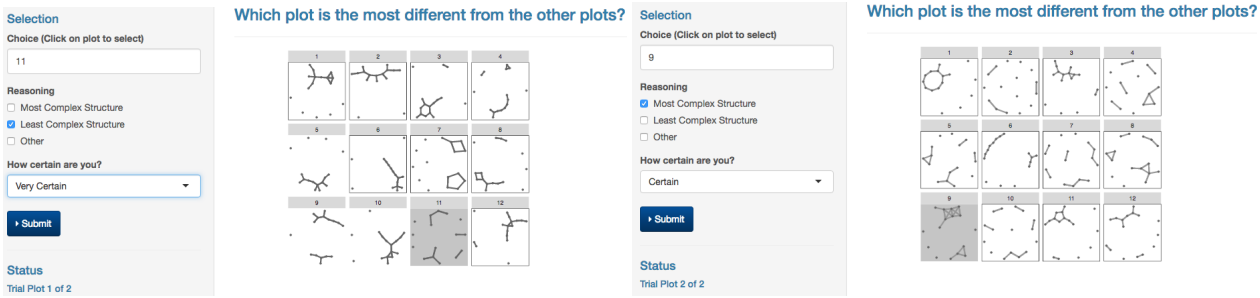


Figure 11: The trial plots that users had to get correct in order to participate in the Amazon Turk experiment.

Table 3: Demographic information collected from experiment participants. An asterisk (*) indicates fewer than 5 participants.

Female	32	18-25	20
I choose not to provide this information	*	26-30	31
Male	69	31-35	24
		36-40	11
		41-45	5
		46-50	2
		51-55	4
		56-60	3
		Over 60	3

Graduate Degree	16
High School or Less	12
I choose not to provide this information	*
Some Graduate Courses	*
Some Undergraduate Courses	30
Undergraduate Degree	40

4.5.3 Results

We collected 10 lineups each from 77 Amazon Turk users. A table of demographic information collected on the participants in the experiment is in Table **??**.

Because of the random assignment of the 25 plots to users, almost every lineup was seen by a different number of people. Repetition 3 of M1 v. M3 was seen by 47 different users while repetition 2 of the same type was seen by only 6 different users. The mean number of times seens was 30.8 and the median was 31. In 15 of the 25 lineups, no user chose the correct plot, and in the remaining 10 lineups, the users correctly identified the alternative plot a minimum of 4.35% of the time and a maximum of 64% of the time. A plot showing the number of different plots chosen, how many times they were chosen, and whether or not it was the correct choice is given in Figure~12.

This plot shows us the abysmal ability of the Turkers to identify the alternative plot correctly. There are also a few other interesting results from this plot. First, there are several lineups where a majority of participants selected the same wrong alternative plot. Additionally, we see a lot of variation in the number of different plots selected at the alternate plot by the Turkers. At most, there were 10 different plots selected, compared to 2 at the opposite end. XXX is it worth exploring these “interesting things" more in-depth? XXX

Next, we investigate the confidence of the experiment's participants in selecting the most different plot from the others. Overall, in 40% of the responses, the respondent was certain they were correct. User confidence in the remaining categories, neutral, uncertain, very certain, and very uncertain, was 26.88%, 16.88%, 9.48%, and 6.76%, respectively. These results are summarized by lineup in Figure~13. We also performed a 5-sample test for equality of proportions in order to test the null hypothesis that the proportion of correct responses is the same in all 5 certainty categories. This test resulted in a p-value of 0.5881, so there is no evidence that the certainty of a user's response affects whether or not they choose the correct plot. This provides further evidence that detecting a network simulated from a different model is an extremely difficult problem.

We next investigate the length of time that users' took to respond to each lineup. The minimum was 3.762 seconds and the maximum was 578.8 seconds (nearly 10 minutes). The median was 13.01 seconds and the mean was 20.32 seconds. First, a 2-sample Kolmogorov-Smirnov test for equality of distribution was done to see if the distribution of times for correct plots chosen is the same as the distribution of times for incorrect plots chosen. This two-sided test resulted in a p-value of 1, so there is no evidence that the distribution of times is different whether or not the response was correct.

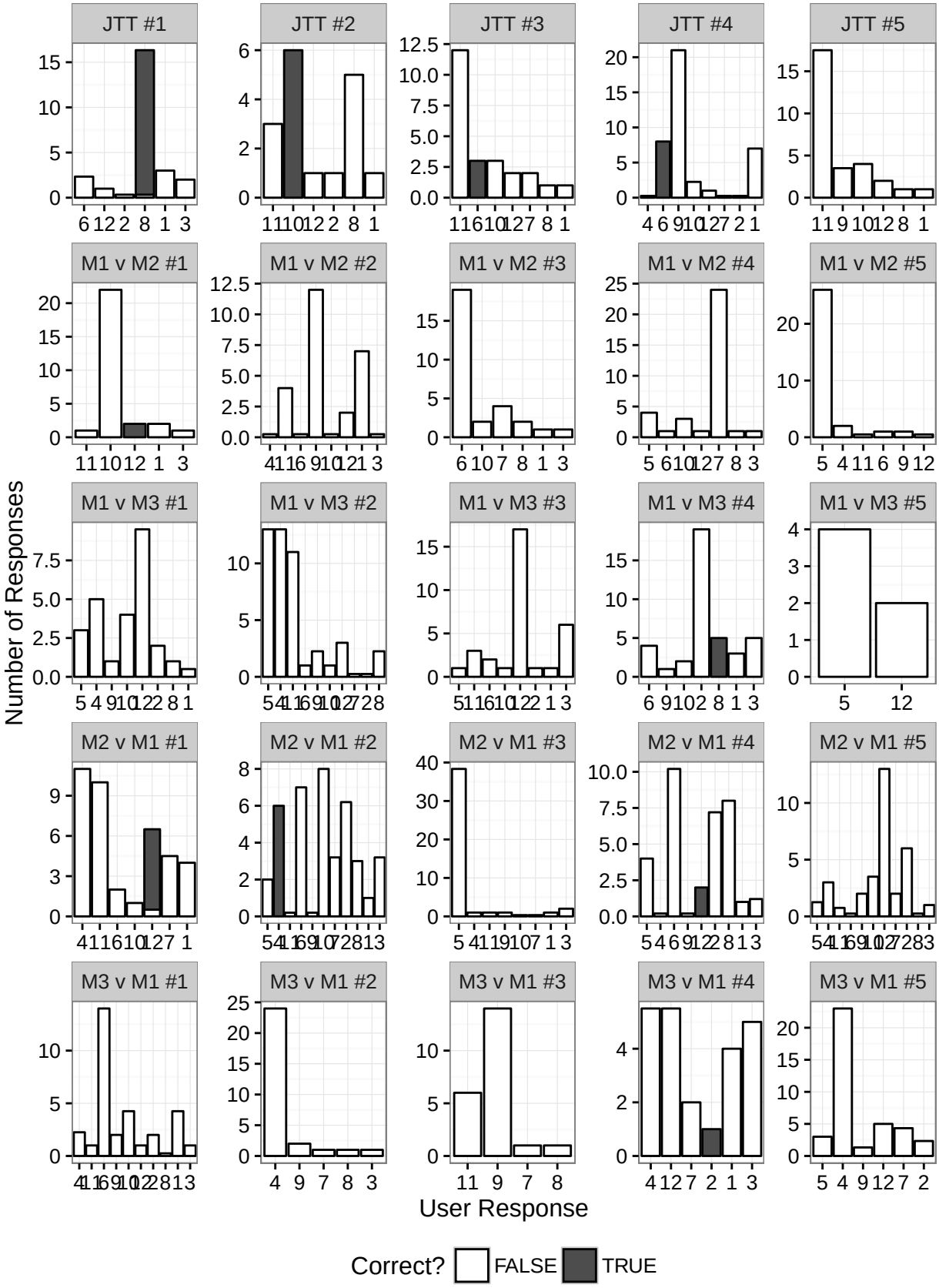


Figure 12: Plots chosen for each of the 25 lineups in the experiment. The x-axis ticks do not have a label because the label is less important than the number of different plots users chose and whether or not they chose the correct plot, shown in the coloring of the bars.

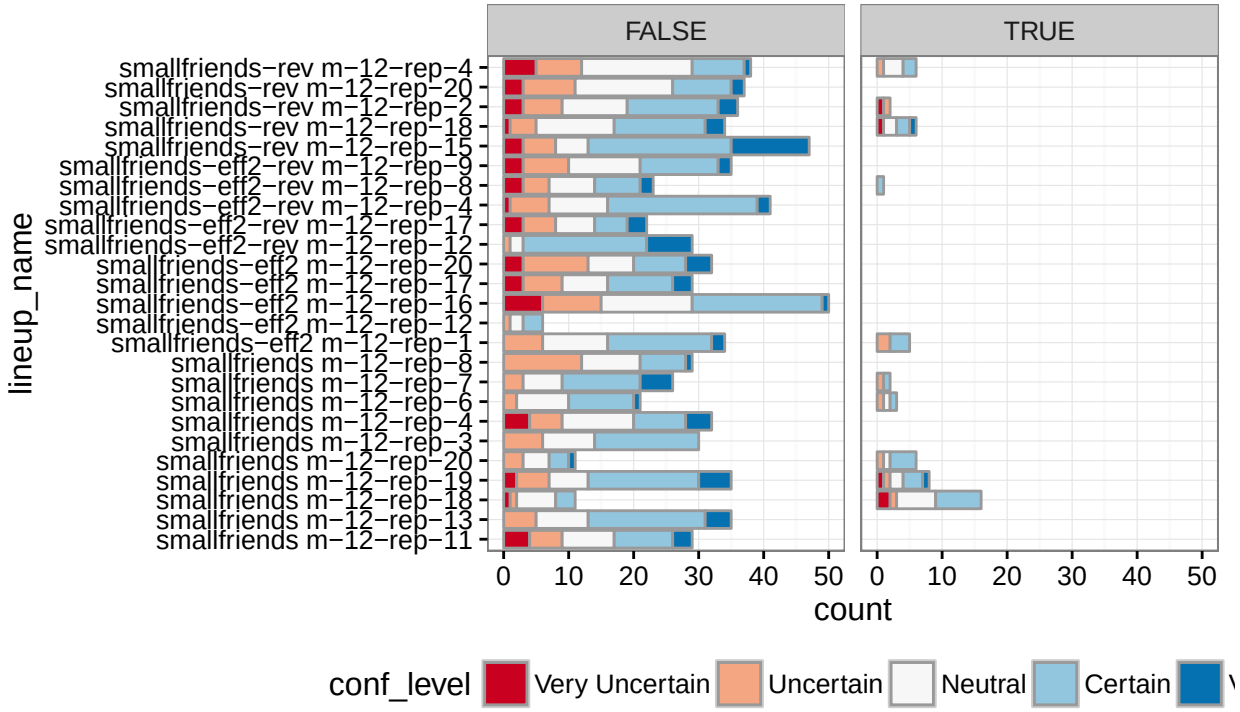
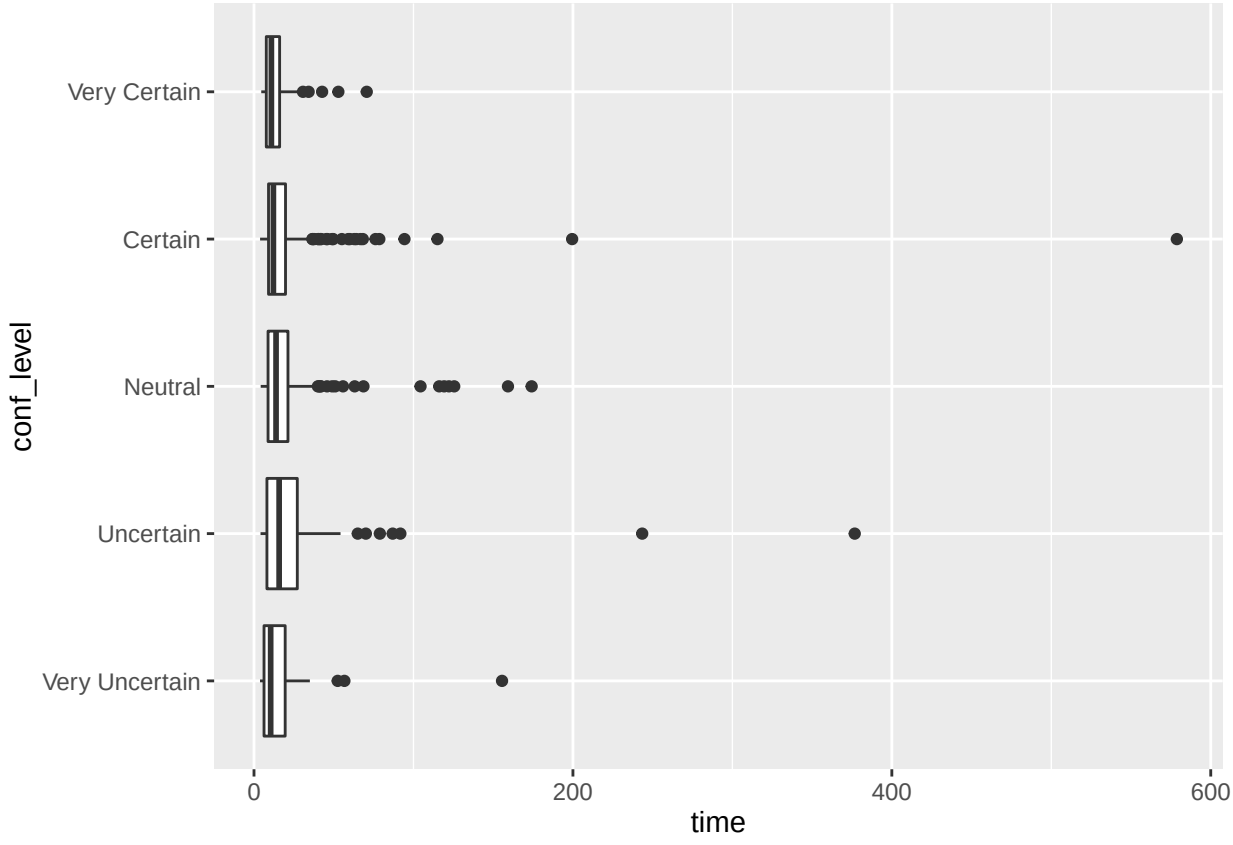
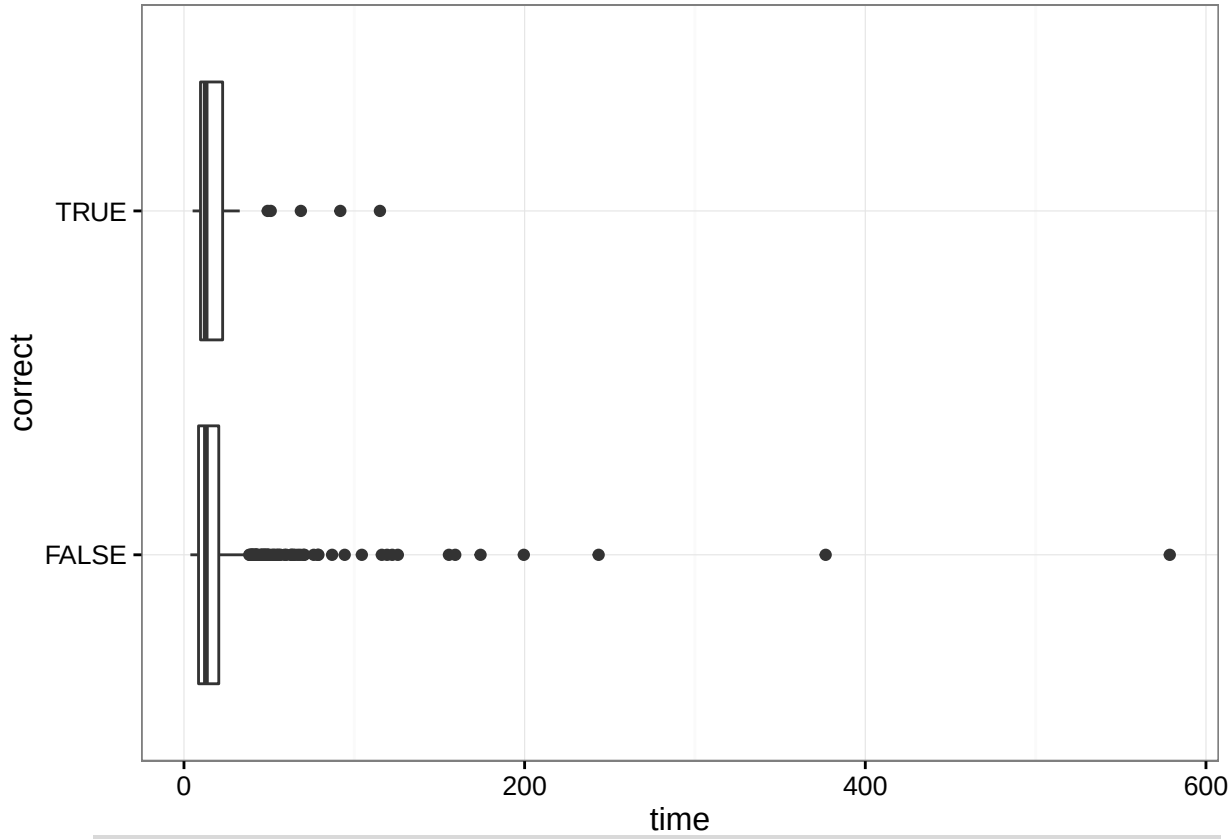
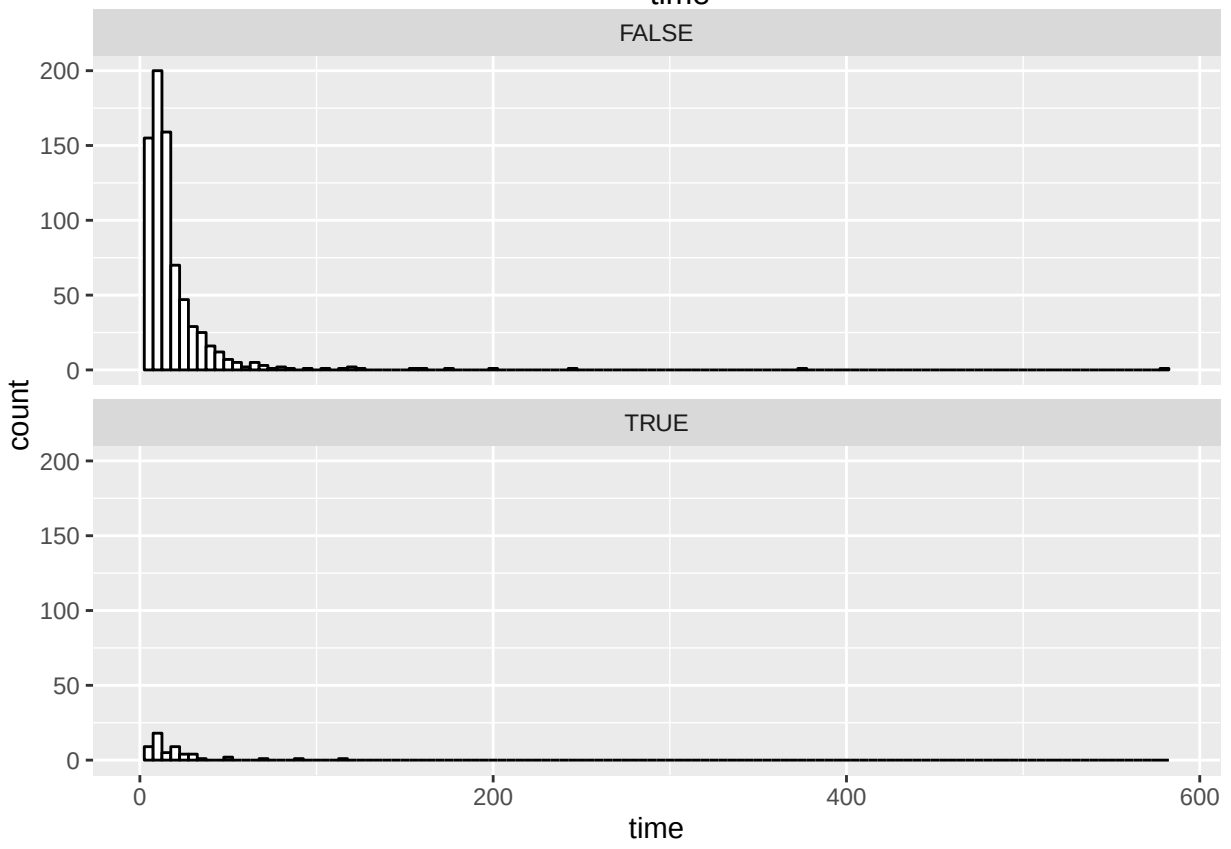
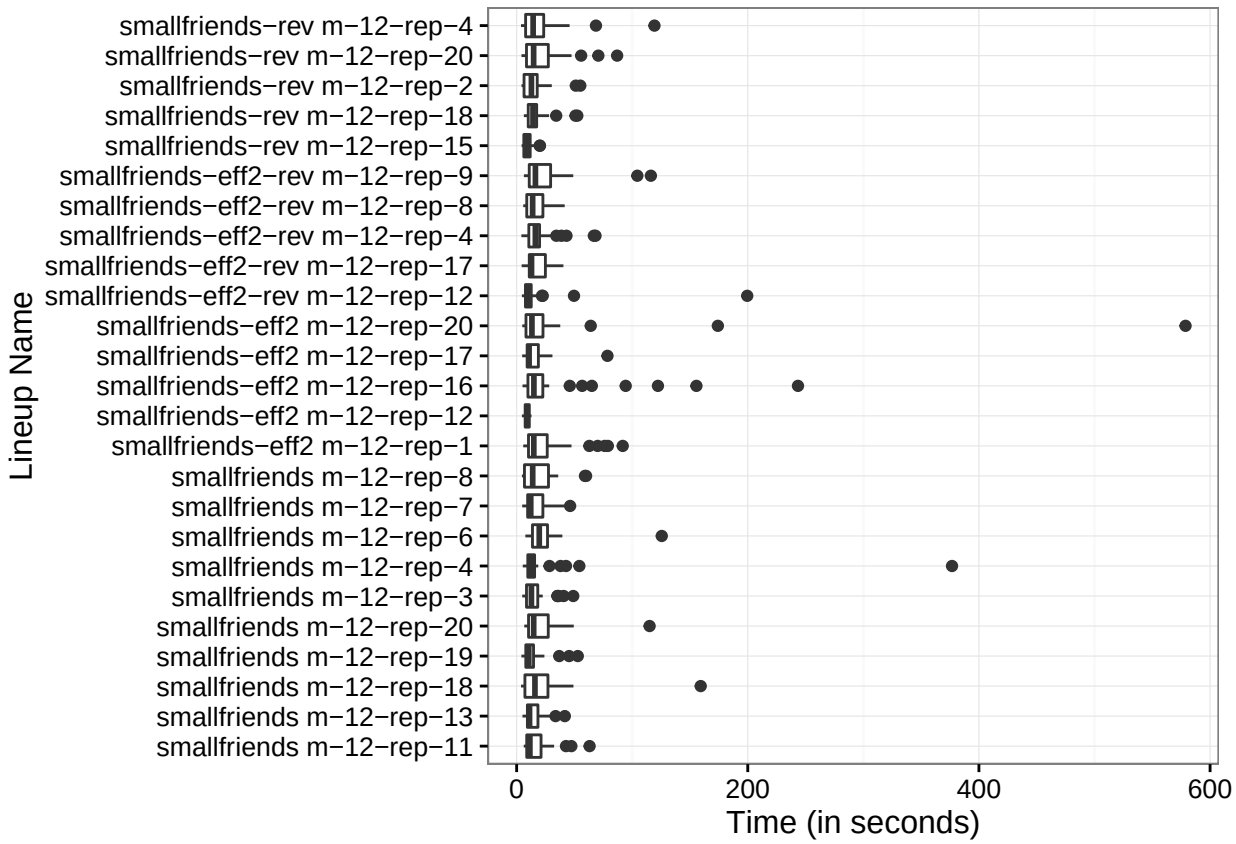


Figure 13: All responses for all lineups, separated by whether the plot selected was the true alternative plot (TRUE) or not (FALSE) and colored by the respondent's uncertainty levels. We see here that most respondents were certain in their answer whether or not they were correct.




TRUE
FALSE
correct
0
200
400
600
time


FALSE
TRUE
count
200
150
100
50
0
0
200
400
600
time



4.5.4 Discussion/Future Work

This small experiment suggests that detecting a network model difference from just one simulation from one model and 11 from the other model is just about impossible. This result is fairly unsurprising: drawing a single random point from, say, a χ^2_1 distribution and placing it in a lineup with 11 separate draws from a standard normal distribution would likely have similar results. Every once in a while, a chi-square value would appear that would be too large to belong with draws from a standard normal distribution, but values near 1, the mean, would probably not appear that different from some random standard normal draws. This means that we need to develop a different way to compare two network models. We need a way to visualize many samples from a network model in one panel.

5 Drawing Networks with the R package `ggplot2`

This next section is a paper I authored with Heike Hofmann (Iowa State University) and François Briatte (European School of Political and Social Sciences) that will be published in the R Journal.

5.1 Introduction

There are many kinds of networks, and networks are extensively studied across many disciplines (Watts (2004)). For instance, social network analysis is a longstanding and prominent sub-field of sociology, and the study of biological networks, such as protein-protein interaction networks or metabolic networks, is a notable sub-field of biology (Prell (2011), Junker and Schreiber (2008)). In addition, the ubiquity of social media platforms, like Facebook, Twitter, and LinkedIn, has brought the concepts of networks out of academia and

into the mainstream. Though these disciplines and the many others that study networks are themselves very different and specialized, they can all benefit from good network visualization tools.

Many R packages already exist to manipulate network objects, such as `statnet` by Handcock et al. (2008), `igraph` by Csardi and Nepusz (2006), `sna` by Butts (2014), and `network` by Butts, Handcock, and Hunter (2014) (see also Butts (2008)). Each one of these packages were developed with a focus of analyzing network data and not necessarily for rendering visualizations of networks. Though these packages do have network visualization capabilities, visualization was not intended as their primary purpose. This is by no means a critique or an inherently negative aspect of these packages: they are all hugely important tools for network analysis that we have relied heavily on in our own work. We have found, however, that visualizing network data in these packages requires a lot of extra work if one is accustomed to working with more "traditional" data structures such as vectors, data frames, or arrays. The visualization tools in these packages require detailed knowledge of each one of them and their syntax in order to build meaningful network visualizations with them. This is obviously not a problem if the user is very familiar with network structures and has already spent time working with network data. If, however, the user is new to network data or is more comfortable working with the traditional data structures, they could find the learning curve for these packages burdensome.

The packages we have written have one primary purpose: to create beautiful network visualizations by providing a wrapper to the popular `ggplot` package Wickham (2009). And so, our focus here is not on adding to the analysis of network data or to the field of graph drawing, Tamassia (2013) but rather it is on implementing existing graph drawing capabilities in the `ggplot` framework, using the more traditional data frame structure.

The `ggplot` package was designed as an implementation of the "grammar of graphics" proposed by Wilkinson (1999), and it has become extremely popular among R users.[2]

Because the syntax implemented in the `ggplot2` package is extendable to different kinds of visualizations, many packages have built additional functionality on top of the `ggplot2` framework. Examples include the `ggmap` package by Kahle and Wickham (2013) for spatial visualization, the `ggfortify` package by Horikoshi and Tang (2015) for visualizing statistical models, the package `ggally` by Schloerke et al. (2016), which encompasses various complementary visualization techniques to `ggplot2`, and the `ggbio` and `ggtree` Bioconductor packages by Yin, Cook, and Lawrence (2012) and Yu et al. (submitted), which both provide visualizations for biological data.

These packages have expanded the utility of `ggplot2`, likely resulting in an increase of its user base. We hope to appeal to this user base and potentially add to it by applying the benefits of the grammar of graphics implemented in `ggplot2` to network visualization. Our efforts rely upon the most recent changes to `ggplot2`, which allow users to more easily extend the package through additional geometries or `geoms`.[3]

In the remainder of this paper, we present three different approaches to network visualization through `ggplot2` wrappers. The section **??** introduces the basic terminology of networks and illustrates their ubiquity in natural and social life. The next section 5.3 then discusses the logic behind each of the three approaches that we offer. The section **??** extends that discussion through several examples ranging from simple to complex networks, for which we provide the code corresponding to each approach alongside its graphical result. We follow with some considerations of runtime behavior in plotting networks in the section **??** before closing with a discussion.

[2]In order to give an indication of how large the user base of `ggplot2` is, we looked at its usage statistics from January 1, 2015 to December 31, 2015 (see link). Over this period, the `ggplot2` package was downloaded over 1.75 million times from CRAN, which amounts to over 6,200 downloads per day. Over 700 R packages list `ggplot2` as a dependency or as a suggestion.

[3]Version 2.1.0, released 1 March 2016. See for the full list of changes in `ggplot2` 2.1.0, as well as the new package vignette, "Extending ggplot2"", which explains how the internal `ggproto` system of object-oriented programming can be used to create new geoms.

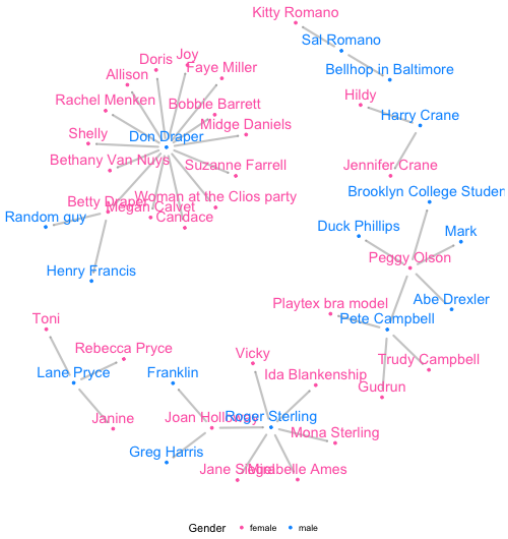


Figure 14: Graph of the characters in the show Mad Men who are linked by a romantic relationship.

5.2 Brief introduction to networks

In its essence, a network is simply a set of vertices connected in pairs by a set of edges (Newman (2010)). Throughout this paper, we also use the terms "nodes" and "actors" to refer to vertices, as well as the terms "ties" and "relationships" to refer to edges. The two sets of graphical objects that make up a network visualization or mapping, points and the segments between them, have been used to examine a huge variety and quantity of information across many different fields of study. For instance, networks of scientific collaboration, a food web of marine animals, and American college football games are all covered in a paper on community detection in networks by Girvan and Newman (2002). Additionally, Buldyrev et al. (2010) study node failure in interdependent networks like power grids. Social networks such as links between actors found on and neural networks, like the completely mapped neural network of the *C. elegans* worm are also extensively studied (D. Watts and Strogatz (1998)).

These examples show that networks can vary widely in scope and complexity: the smallest network with connections is just one edge between two vertices, while one of the most commonly used and most complex networks, the world wide web, has billions of vertices (Web pages) and billions of edges (hyperlinks) connecting them. Additionally, the edges in a network can be directed or undirected: directed edges represent an ordering of vertices, like a relationship extending from one vertex to another, where switching the direction would change the structure of the network. The World Wide Web is an example of a directed network because hyperlinks connect one Web page to another, but not necessarily the other way around. Undirected edges are simply connections between vertices where order does not matter. Co-authorship networks are examples of undirected networks, where nodes are authors and they are connected by an edge if they have written an academic publication together.

As a reference example, we turn to a specific instance of a social network. A social network is a network that everyone is a part of in one way or another, whether through friends, family, or other human interactions. We do not necessarily refer here to social media like Facebook or LinkedIn, but rather to the connections we form with other people. To demonstrate the functionality of our tools for plotting networks, we have chosen an example of a social network from the popular television show *Mad Men*. This network, which was compiled by Chang (2013) and made available in `gcookbook` (Chang (2012)), is made up of 52 vertices and 87 edges. Each vertex represents a character on the show, and there is an edge between every two characters who have had a romantic relationship.

Figure **??** is a visualization of this network. In the plot, we can see one central character who has many more relationships than any other character. This vertex represents the main character of the show, Don Draper,

who is quite the "ladies' man." Networks like this one, no matter how simple or complex, are everywhere, and we hope to provide the curious reader with a straightforward way to visualize any network they choose.

Coloring the vertices or edges in a graph is a quick way to visualize grouping and can help with pattern or cluster detection. The vertices in a network and the edges between them compose the structure of a network, and being able to visually discover patterns among them is a key part of network analysis. Viewing multiple layouts of the same network can also help reveal patterns or clusters that would not be discovered when only viewing one layout or analyzing only its underlying adjacency matrix.

5.3 Three implementations of network visualizations

We present two basic approaches to using the `ggplot2` framework for network visualization. First, we implement network visualizations by providing a wrapper function, `ggnet2` for the user to visualize a network using `ggplot2` elements. Second, we implement network visualizations using layering in `ggplot2`. For the second approach, we have two ways of creating a network visualization. The first, `geomnet`, wraps all network structures, including vertices, edges, and vertex labels into a single `geom`. The second, `ggnetwork`, implements each structural component in an independent `geom` and layers them to create the visualization. In each package, our goal is to provide users with a way to map network properties to aesthetic properties of graphs that is familiar to them and straightforward to implement. Each package has a slightly different approach to accomplish this goal, and we will discuss all of these approaches in this section.

5.3.1 The `ggnet2` function

The `ggnet2` function is an improved version of the `ggnet` function. Both functions are part of the `GGally` package (Schloerke et al. (2016)). A detailed description of the `ggnet2` function is available from within the package as a vignette.[4] The `ggnet2` function offers a large range of network visualization functionality in a single function call. Although its result is a `ggplot2` object that can be further styled with `ggplot2` scales and themes, the syntax of the `ggnet2` function is designed to be easily understood by the user. The aesthetics relating to the nodes are controlled by arguments such as `node.alpha` or `node.color`, while those relating to the edges are controlled by similarly named arguments starting with `edge`. As a consequence, while `ggnet2` applies the grammar of graphics to network objects, the function itself still works very much like the plotting functions of the `igraph` and `network` packages: a long series of arguments is used to control every possible aspect of how the network should be visualized.

The `ggnet2` function takes a single network object as input. This initial object might be an object of class `network` (with the exception of hypergraphs or multiplex graphs), or any data structure that can be coerced to an object of that class, such as an incidence matrix, an adjacency matrix, or an edge list. Additionally, if the `intergraph` package (Bojanowski (2015)) is installed, the function also accepts a network object of class `"igraph"`. Internally, the function converts the network object to two data frames: one for edges and another one for nodes. It then passes them to `ggplot2`. Each of the two data frames contain the information required by `ggplot2` to plot segments and points respectively, such as a shape for the points (nodes) and a line type for the segments (edges). The final result returned to the user is a plot with a minimum of two layers, or more if there are edge and/or node labels.%

The `mode` argument of `ggnet2` controls how the nodes of the network are to be positioned in the plot returned by the function. This argument can take any of the layout values supported by the `gplot.layout` function of the `sna` package, and defaults to `fruchtermanreingold`, which places the nodes through the force-directed layout algorithm by Fruchterman and Reingold (1991). Many other possible layouts and their parameters can also be passed to `ggnet2` through the `layout.par` argument. For a list of possible layouts and their arguments, see `?sna::gplot.layout`.

Other arguments passed to the `ggnet2` function offer extensive control over the aesthetics of the plot that it returns, including %through the addition of edge and/or node labels and their respective aesthetics.

[4]The vignette can be viewed online at

Arguments such as `node.shape` or `edge.lty`, which control the shape of the nodes and the line type of the edges, respectively, can take a global value, such as `15` or `dashed`, a vector of global values, or the name of an edge or vertex attribute, in which case the values of that attribute are used as the mapping aesthetic.

This last functionality builds on one of the strengths of the `network` class, which can store information on network edges and nodes as attributes that are then accessible to the user through the `\%e\%` and `\%v\%` operators respectively.[5]

If the `ggnet2` function is given the `node.alpha = "importance"` argument, it will interpret it as an attempt to map the vertex attribute called `importance` to the transparency level of the nodes. This works exactly like the command `net \%v\% "importance"`, which returns the vertex attribute `importance` of the `"network"` object `net`. This functionality allows the `ggnet2` function to work in a similar fashion to `ggplot2` mappings of aesthetics within the `aes` operator.

The `ggnet2` function also provides a few network-specific options, such as sizing the nodes as a function of their unweighted degree, or using the primary and secondary modes of a bipartite network as an aesthetic mapping for the nodes.

All in all, the `ggnet2` function combines two different kinds of processes: it translates a network object into a data frame suitable for plotting with `ggplot2`, and it applies network-related aesthetic operations to that data frame, such as coloring the edges in function of the color of the nodes that they connect.

5.3.2 The `geomnet` Package

6 Summary, Impact, and Future Work

7 Other Projects

References

Airoldi, Edoardo M. 2006. "Bayesian Mixed Membership Models of Complex and Evolving Networks." PhD thesis, School of Computer Science, Carnegie Mellon University.

Amazon. 2010. Https://www.mturk.com/mturk/welcome.

Barabási, Albert-László, and Réka Albert. 1999. "Emergence of Scaling in Random Networks." *Science* 286 (5439). American Association for the Advancement of Science: 509–12. doi:10.1126/science.286.5439.509.

Bojanowski, Michal. 2015. *Intergraph: Coercion Routines for Network Data Objects*. http://mbojan.github.io/intergraph.

Buja, Andreas, Dianne Cook, Heike Hofmann, Michael Lawrence, Eun-Kyung Lee, Deborah F. Swayne, and Hadley Wickham. 2009. "Statistical Inference for Exploratory Data Analysis and Model Diagnostics." *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 367 (1906). The Royal Society: 4361–83. doi:10.1098/rsta.2009.0120.

Buldyrev, Sergey V., Roni Parshani, Gerald Paul, H. Eugene Stanley, and Shlomo Havlin. 2010. "Catastrophic Cascade of Failures in Interdependent Networks." *Nature* 464 (7291): 1025–8.

Butts, Carter T. 2008. "network: a Package for Managing Relational Data in R." *Journal of Statistical Software* 24 (2).

———. 2014. *Sna: Tools for Social Network Analysis*. https://CRAN.R-project.org/package=sna.

Butts, Carter T., Mark S. Handcock, and David R. Hunter. 2014. *Network: Classes for Relational Data*.

[5] See@network, p. 22-24. The equivalent operators in the `igraph` package are called `E` and `V`.

Irvine, CA. http://statnet.org/.

Chang, Winston. 2012. *Gcookbook: Data for "R Graphics Cookbook"*. https://CRAN.R-project.org/package=gcookbook.

———. 2013. *R Graphics Cookbook*. Sebastopol, CA: O'Reilly.

Csardi, Gabor, and Tamas Nepusz. 2006. "The Igraph Software Package for Complex Network Research." *InterJournal* Complex Systems: 1695. http://igraph.org.

Duijn, Marijtje A. J. van, Tom A. B. Snijders, and Bonne J. H. Zijlstra. 2004. "P2: A Random Effects Model with Covariates for Directed Graphs." *Statistica Neerlandica* 58 (2). Blackwell Publishing: 234–54. doi:10.1046/j.0039-0402.2003.00258.x.

Erdős, Paul, and Alfréd Rényi. 1959. "On Random Graphs I." *Publicationes Mathematicae* 6: 290–97.

Fruchterman, Thomas M.J., and Edward M. Reingold. 1991. "Graph Drawing by Force-Directed Placement." *Software: Practice and Experience* 21 (11): 1129–64.

Girvan, M., and M. E. J. Newman. 2002. "Community Structure in Social and Biological Networks." *Proc. Natl. Acad. Sci. USA* 99 (12): 7821–6.

Goldenberg, Anna, Alice X. Zheng, Stephen E. Fienberg, and Edoardo M. Airoldi. 2010. "A Survey of Statistical Network Models." *Foundations and Trends in Machine Learning* 2 (2): 129–233. doi:10.1561/2200000005.

Handcock, Mark S., David R. Hunter, Carter T. Butts, Steven M. Goodreau, and Martina Morris. 2008. "Statnet: Software Tools for the Representation, Visualization, Analysis and Simulation of Network Data." *Journal of Statistical Software* 24 (1): 1–11. http://www.jstatsoft.org/v24/i01.

Hoff, Peter D, Adrian E Raftery, and Mark S Handcock. 2002. "Latent Space Approaches to Social Network Analysis." *Journal of the American Statistical Association* 97 (460): 1090–8.

Holland, Paul W., and Samuel Leinhardt. 1981. "An Exponential Family of Probability Distributions for Directed Graphs." *Journal of the American Statistical Association* 76 (373). [American Statistical Association, Taylor & Francis, Ltd.]: 33–50. http://www.jstor.org/stable/2287037.

Horikoshi, Masaaki, and Yuan Tang. 2015. *Data Visualization Tools for Statistical Analysis Results*. http://CRAN.R-project.org/package=ggfortify.

Junker, B.H., and F. Schreiber. 2008. *Analysis of Biological Networks*. Wiley Series in Bioinformatics. Wiley. https://books.google.com/books?id=2DloLXaXSNgC.

Kahle, David, and Hadley Wickham. 2013. "Ggmap: Spatial Visualization with Ggplot2." *The R Journal* 5 (1): 144–61. http://journal.r-project.org/archive/2013-1/kahle-wickham.pdf.

Kolaczyk, Eric D. 2009. *Statistical Analysis of Network Data: Methods and Models*. Springer.

Newman, M. E. J. 2010. *Networks : An Introduction*. Oxford New York: Oxford University Press.

Prell, C. 2011. *Social Network Analysis: History, Theory and Methodology*. SAGE Publications. https://books.google.com/books?id=wZYQAgAAQBAJ.

R Core Team. 2016. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Ripley, Ruth, Krists Boitmanis, and Tom A.B. Snijders. 2013. *RSiena: Siena - Simulation Investigation for Empirical Network Analysis*. https://CRAN.R-project.org/package=RSiena.

Schloerke, Barret, Jason Crowley, Di Cook, François Briatte, Moritz Marbach, Edwin Thoen, Amos Elberg, and Joseph Larmarange. 2016. *GGally: Extension to Ggplot2*. https://CRAN.R-project.org/package=GGally.

Snijders, T A B. 1996. "Stochastic actor-oriented models for network change." *Journal of*

Mathematical Sociology 21 (1-2): 149–72. http://www.scopus.com/inward/record.url?eid=2-s2.0-0000413317{\&}partnerID=40{\&}md5=d981a59ed505ebc7fe083b42a8b9a179.

Snijders, Tom A. B. 2001. "The Statistical Evaluation of Social Network Dynamics." *Sociological Methodology* 31 (1). Blackwell Publishers, Inc.: 361–95. doi:10.1111/0081-1750.00099.

Tamassia, Roberto, ed. 2013. *Handbook of Graph Drawing and Visualization*. CRC Press.

University, Oakland. 2016. "The Erdös Number Project." *Http://wwwp.oakland.edu/enp/*.

Wasserman, Stanley S. 1980. "A Stochastic Model for Directed Graphs with Transition Rates Determined by Reciprocity." *Sociological Methodology* 11. [American Sociological Association, Wiley, Sage Publications, Inc.]: 392–412. http://www.jstor.org/stable/270870.

Watts, D J, and S H Strogatz. 1998. "Collective dynamics of 'small-world' networks." *Nature* 393 (6684): 440–2. doi:10.1038/30918.

Watts, DJ, and SH Strogatz. 1998. "Collective Dynamics of 'Small-World' Networks." *Nature* 393 (6684): 440–42.

Watts, Duncan J. 2004. "The 'New' Science of Networks." *Annual Review of Sociology* 30: 243–70.

Wickham, Hadley. 2009. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

Wilkinson, Leland. 1999. *The Grammar of Graphics*. New York: NY: Springer.

Yin, Tengfei, Dianne Cook, and Michael Lawrence. 2012. "Ggbio: An R Package for Extending the Grammar of Graphics for Genomic Data." *Genome Biology* 13 (8). BioMed Central Ltd: R77.

Yu, Guangchuang, David Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. submitted. "Ggtree: An R Package for Visualization and Annotation of Phylogenetic Tree with Different Types of Meta-Data." *Methods in Ecology and Evolution*.