Model Visualization Techniques for a Social Network Model

Samantha Tyner*

Department of Statistics and Statistical Laboratory, Iowa State University

and

Heike Hofmann

Department of Statistics and Statistical Laboratory, Iowa State University

September 21, 2017

Abstract

Social networks have been studied for decades, beginning with a few foundational works, including the 1967 study, "The Small World Problem" by Stanley Milgram. In this paper, we concentrate on one type of model for dynamic social networks: the stochastic actor-oriented models (SAOMs), introduced by Snijders (1996). Unlike other network models, SAOMs are not very well understood. We use model visualization techniques introduced in Wickham et al (2015) in order to make them a little less murky. The SAOMs are a prime example of a set of models that can benefit greatly from application of model visualization. With the help of static and dynamic visualizations, we bring the hidden model fitting processes into the foreground, eventually leading to a better understanding and higher accessibility of stochastic actor-oriented models for social network analysts.

Keywords: social network analysis, model visualization, dynamic networks, network visualization, network mapping, animation

Contents

1 Introduction

Social networks have been studied for decades, beginning with a few foundational works, including the 1967 study, "The Small World Problem" by Stanley Milgram (Goldenberg et al., 2010). Examples of social networks include collaboration networks between academic researchers, friendship networks in a school or university, and trade networks between nations. In recent years, the study of social networks has grown in popularity due to an increase in the availability and access to social network data. There are many kinds of social networks, but there are not as many statistical models for social network data. Some network models that have been applied to social networks include the exponential random graph model and latent space models. These models, however, are only for single instance networks. If we only have one network observation, or only care about one state of a complex network in time, like a snapshot of the World Wide Web, using the well-established models for single network observating is not a problem. If, on the other hand, we have many observations of social network over time, these models may not be appropriate because they do not explicitly allow for the network to change as time passes. When studying a network over time, referred to as a *dynamic network*, we need a model that can take the time aspect of the network into account. Models for dynamic social networks have a great deal of modelling potential because of how realistic their structure can be. A social network does not form spontaneously: it evolves over time. Ties are formed and dissolved, and new actors join the social structure. Modeling the underlying mechanisms that create network changes over time is very complex but also provides potential to uncover hidden truths.

In this paper, we concentrate on one type of model for dynamic social networks: the stochastic actor-oriented models (SAOMs), introduced by Snijders (1996). These models are fundamentally different from other social network models because they allow us to incorporate network *and* actor statistics, where other models only rely on the network statistics, to model the changes in the network. Allowing the actor-level statistics to directly effect the structure of the network leads to a more practical and relevant approach to model change in a social network. In the "real" world we expect people with common interests to be more likely to form relationships, and SAOMs allow us to incorporate this intuition in the modeling process.

Unlike other network models, SAOMs are not very well understood. They are relatively new, especially compared to the classic exponential random graph models, and they are not very tractable analytically. Likelihood functions quickly very complex objects to analyze due to the dependency structure inherent in the data. Therefore, computationally more tractable solutions are used to fit estimators, and in particular, SAOMs are often fit to data using a series of Markov Chain Monte Carlo (MCMC) phases for finding method of moments estimators. In order to estimate the parameters of SAOMs, we use the software SIENA, and its `R` implementation `RSiena`, which was developed by Ripley et al. (2016a). This software marks a huge contribution to the field of social network analysis, but the many moving pieces involved in parameter estimation are largely "behind the scenes" and hidden from the software user. In this paper, in order to better understand the model-fitting process, we attempt to bring SAOM fits and the fitting process out of their black boxes, by combining the principals of network visualization with those of model visualization as discussed in Wickham et al. (2015). By bringing some light to the underlying methods and structures that are behind the scenes when fitting SAOMs to network data, we aim to help researchers working with the models better understand the implications and analyses of these models.

We get into the SAOM black box by using model visualization (see Wickham et al. (2015)) to display the model in the data space, view collections of models instead of single models, and visually explore the process of fitting the SAOMs as opposed to looking at only the final output. Stochastic actor-oriented models are a prime example of a set of models that can benefit greatly from the application of model visualization. For instance, the models themselves include a continuous-time Markov chain (CTMC) that is completely hidden from the analyst in the model fitting process. Bringing the CTMC out of the black box and into the light through model visualization can provide researchers with insights into the underlying features of the model. Furthermore, SAOMs can include a great deal of parameters to be added to the model structure, each of which is attached to a network statistic. These statistics are often somewhat, if not highly, correlated, which causes high correlation between the associated parameters in a SAOM. By visualizing collections of SAOMs, we gain a better understanding of these correlations and find ways

4

to deal with them and rectify their effects in the model. In addition, the estimation of the parameters in a SAOM relies on a Robbins-Monro algorithm, and the convergence checks for these estimates rely on simulation from the fitted model. Again, each of these steps are largely kept in the background of the estimation process. With the help of static and dynamic visualizations we bring the hidden model fitting processes into the foreground, eventually leading to a better understanding and higher accessibility of stochastic actor-oriented models for social network analysts.

In Section 2, we introduce basic concepts of networks and network visualizations. In Section 3, we present the family of stochastic actor-oriented models for social network analysis. In Section 4, we combine concepts from Sections 2 and 3 in an application of the model-vis paradigm, and conclude with a discussion in Section 5.

2 Networks and their Visualizations

2.1 Introduction to Network Structures

Network data is of frequent interest to researchers in a wide array of fields. There are technological networks, like power grids or the internet, information networks, such as citation networks or the World Wide Web, biological networks, like neural networks, and social networks, just to name a few (Newman, 2010). Each of these examples have one thing in common: their data *structure*. There are always units of observation: the power stations, websites, neurons, and people, which we refer to throughout this paper as *nodes* or *actors*. There are also always connections of some kind between those units: the power lines, hyperlinks, electrical signals, and relationships, which we will call *edges* or *ties*. Networks might change over time, like when new websites and hyperlinks are added on the World Wide Web, or when there are new people and relationships in a friendship network. The nodes and edges themselves can also have inherent variables of interest, e.g. the institution of authors in a co-authorship network, or the number of times two authors have collaborated.

The multiple layers of network data structures pose unique problems to network analysts. Some questions that network researchers may aim to answer are: How does the

strength of a tie between two nodes affect the overall structure of the network? Do node-level differences affect the formation or dissolution of edges? Which views of the data are most informative for communicating significant effects and other results of statistical analyses of the network of interest? These are, of course, just a few broad questions, and we focus here on the latter, which we aim to answer through visual exploration of network data and models.

2.2 Visualizing Network Data

Network visualization, also called network mapping, is a prominent subfield of network analysis. Visualizing network data is uniquely difficult because of the structure of the data itself. Most, if not all, data visualizations rely on well-defined axes inherited from the data. If variables are numerical, histograms, scatterplots, or time series plots are straightforward to construct. If the variables are categorical, bar charts and mosaic plots are available to to the researcher. If the data are spatial, there is a well-defined region in which to view information. Network data, however, are much less cut-and-dried.

There are two primary methods used to visualize networks: node-link diagrams and adjacency matrix visualization (Donald E. Knuth and John J. Watkins, 2013; Fekete, 2009). As a toy example, let us assume that we have five nodes, $\{1, 2, 3, 4, 5\}$, connected by five directed edges: $\{2 \rightarrow 4, 3 \rightarrow 4, 1 \rightarrow 5, 3 \rightarrow 5, 5 \rightarrow 4\}$ We use this toy data set to demonstrate the two visualization methods in Figure 1.

The first method, the node-link diagram, represents nodes with points in 2D Euclidean space and then represents edges by connecting the points with lines when there is an edge between the two nodes. These lines can also have arrows on them indicating the direction of the edge for directed networks. But because there is typically no natural placement of the points unless they have important spatial locations, a random placement of the points is used, then adjusted via a layout algorithm, of which there are many (Gibson et al., 2013).

Some commonly used layout algorithms, such as the Kamada-Kawai layout (Kamada and Kawai, 1989) and the Fruchterman-Reingold layout (Fruchterman and Reingold, 1991), are designed to mimic physical systems, drawing the graphs based on the "forces" connecting them. In these algorithms, the edges of the network act as springs pushing and pulling
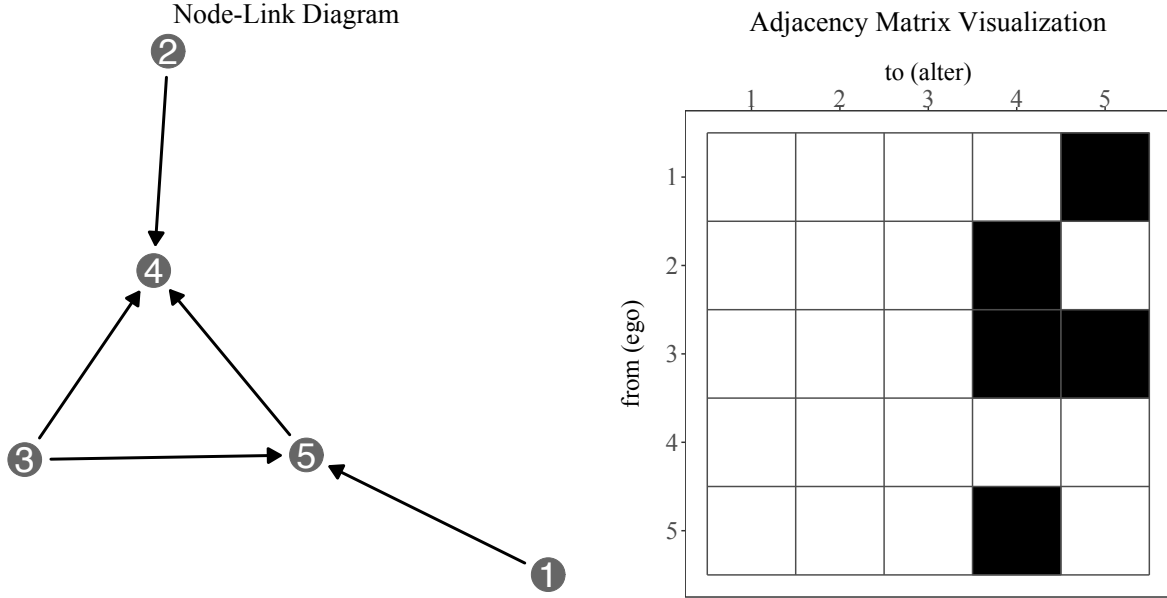
Figure 1: On the left, a node-link diagram of our directed toy network, with nodes placed using the Kamada-Kawai algorithm. On the right, the adjacency matrix visualization for that same network.

the nodes in a low dimensional (usually two-dimensional) space. Another algorithm uses multi-dimensional scaling, relying on distance metric and computing a matrix whose entries represent the "distance" between every pair of nodes. There are also layout algorithms that use properties of the adjacency matrix, like its eigenstructure, to place the nodes in 2D space (Gibson et al., 2013). The node-link diagram using the Kamada-Kawai layout algorithm for our toy network is shown in Figure 1. Unless otherwise stated, all other node-link diagrams in this paper will use the Kamada-Kawai layout.

The second primary method for network visualization uses the adjacency matrix of the network. The adjaceny matrix of a network, $\mathbf{A}$, describes the edges of a network in matrix form. An entry A_{ij} of $\mathbf{A}$, for two nodes $i \neq j$ in the network is defined as

$$A_{ij} = \begin{cases} 1 & \text{if an edge exists } i \to j \\ 0 & otherwise \end{cases}$$

Note here that our edge variables are binary: we only consider the presence or absence of an edge. If the network has weighted edges, for example an email network where edge weights represent the number of emails sent from one person to another, the entries in

the adjacency matrix are the edge weights instead of zeroes and ones. In an undirected network, $A_{ij} = A_{ji}$, but in a directed graph this is only true if there is an edge from i to j and from j to i. Thus, $\mathbf{A}$ is always symmetric for undirected networks, and is symmetric for directed networks only if every edge between two nodes is reciprocated. An adjacency matrix visualization for our toy example is also shown in Figure 1.

Each type of visualization comes with its own advantages and disadvantages. For example, paths between two nodes in a network are easier to determine with node-link diagrams than with adjacency matrix visualizations (Ghoniem et al., 2005). In node-link diagrams, node-level information can be incorporated into the visualization by coloring or changing the shape of the points representing the nodes, and edge-level information can be incorporated by coloring the lines, or changing their thickness, linetype, or color. Incorporating a node-level variable into an adjacency matrix visualization is not as straightforward or simple, which is more focused on edges. Adjacency matrix visualization has been found to be particularly useful when the network is very complex, dense, or large, and experimental studies have shown adjacency matrix visualization to be superior to node-link diagrames for large networks. For example, for basic perceptual tasks on networks, including node and edge count, adjacency matrix visualizations outperform node-link diagrams as the size and density of the network increases (Ghoniem et al., 2005). One drawback of the adjacency matrix visualization that Ghoniem et al. found was that edges are overrepresented for undirected graphs, due to the symmetry of $\mathbf{A}$: the edge x_{ij} for $i \neq j$ appears in $\mathbf{A}$ twice: in A_{ij} and A_{ji}, and so it also appears twice in the adjacency matrix visualization. This, however, may actually be an advantage for *directed* graphs, where exactly the correct number of edges is represented in a matrix visualization, due to the fact that the edges x_{ij} and x_{ji} are not interchangeable. A node-link diagram, however, may underrepresent the edge count if the edges x_{ij} and x_{ji} both exist and are drawn on top of one another. Ultimately, however there is not one "correct" way to visualize network information, and we will be using both the node-link and adjacency matrix visualization methods throughout this paper to explore social networks and stochastic actor-oriented models.

3 Stochastic Actor-Oriented Models for Longitudinal Social Networks

A Stochastic Actor-Oriented Model (SAOM) is a model that incorporates all three components of dynamic networks: edge, node, and time information. It models the change of a network over time, allowing for changes in network structure due to actor-level covariates. This model was first introduced by Snijders in 1996 (Snijders, 1996). The two titular properties of SAOMs, stochasticity and actor-orientation, are crucial to understanding networks as they exist naturally. Most social networks, even holding constant the set of actors over time, are ever-changing as relationships decay or grow in seemingly random ways, and most actors (or nodes) in social networks have inherent properties that could affect how they change their role within the network, and vice versa.

3.1 Definitions, Terminology, and Notation

In this paper, the term *dynamic network* refers to a network, consisting of a fixed set of n nodes, that is changing over time, and is observed at M discrete time points, $t_1, \ldots, t_M$ with $t_1 \leq t_2 \leq \cdots \leq t_M$. We denote the network observation at timepoint t_k by $x(t_k)$. In the modelling process, we condition on the first observation, $x(t_1)$. The SAOM assumes that this longitudinal network of discrete observations is embedded within a continuous time Markov chain (CTMC), which we will denote $X(T)$. This process is almost entirely unobserved: we assume that the beginning of the process, $X(0)$, is equivalent to the first network observation $x(t_1)$, while the end of the process, $X(\infty)$, is equivalent to the last observation $x(t_M)$. Nearly all other parts of the process are unseen, with the exception of $x(t_2), \ldots, x(t_{M-1})$. Unlike the first and last observations of the network, these "in-between" observations do not have direct correspondence with steps in the continuous time Markov chain. Thus, the "in-between" observations are considered to be "snapshots" of the network at some point between two steps in the CTMC. The whole process $X(T)$ is a series of single tie changes that happen according to some pre-defined rate function, where one actor at a time is given the opportunity to add or remove one outgoing tie, or to not make any changes. Once an actor is chosen at random according to the rate function, it is "given"

the chance to change a tie, and it tries to maximize its utility function based on the current and near future states of the network. We expand on the model description further in the subsequent sections.

3.1.1 The Rate Function

For the network x and each actor i in the network, the number of times that an actor i gets to change its ties, x_{ij}, to other nodes $j \neq i$ in the network is dictated by a *rate function* $\rho(x, \mathbf{z}, \boldsymbol{\alpha})$, where $\boldsymbol{\alpha}$ are the parameters in the function ρ, and x is the current network state, with covariates of interest $\mathbf{z}$. For this paper, we assume a simple rate function, $\rho(x, \mathbf{z}, \boldsymbol{\alpha}) = \alpha_m$ that is constant across all actors between observations at time t_m to t_{m+1}, thus our rate function is just a rate parameter in the overall model. In general, SAOMs can incorporate covariate values and network statistics into the model, so that each node will have a different rate of change. Other modelling scenarios allow this rate to be more flexible, e.g. a function that depends on the time period of observation, some actor-level covariates or some actor-level network statistics. In our simple model with a simple rate parameter instead of a rate function, the rate parameter dictates how quickly an actor i gets an opportunity to change one of its ties to the other nodes in the network, x_{ij}, for $j \in \{1, \ldots, n\}$ in the time period from t_m to t_{m+1}. If $j = i$, no change in the network is made. The model also assumes that the actors i are conditionally independent given their ties, $x_{i1}, \ldots, x_{in}$ at the current network state. Let $\tau(i|x, m)$ be the wait time until actor i makes its next change from its current state in the network x. Note that m indicates the number of the wave that is conditioned on in the SAOM. For any time point, T, where $t_m \leq T < t_{m+1}$, the waiting time to the next change opportunity by actor i is exponentially distributed with expected value α_m^{-1}. The conditional independence of nodes in the network given their current ties is expressed in Equation 1.

$$\tau(i|x, m)|x_{i1}(m), \ldots, x_{in}(m) \overset{\text{iid}}{\sim} Exp(\alpha_m) \tag{1}$$

The waiting time to the next change opportunity by *any* actor in the network is also exponentially distributed with expected value $(n\alpha_m)^{-1}$. The distribution of waiting time for the whole network to change, $\tau(x|m) = \sum_i \tau_i(m)|x_{i1}(m), \ldots, x_{in}(m)$ can then be written

$$\tau(x|m) \sim Exp(n\alpha_m) \tag{2}$$

The parameter for the wait time for the whole network $n\alpha_m$ is the rate at which any tie change occurs. The estimation of this parameter is straightforward: a the method of moments is used to estimate the rate with the statistic

$$C = \sum_i \sum_j |x_{ij}(t_{m+1}) = x_{ij}(t_m)|$$

which is the total number of changes from observation at time t_m to the observation at time t_{m+1}.

3.1.2 The Objective Function

Because of the conditional independence assumptions given in Equation 1, we can consider the objective function for each node separately, as only one tie from one node is allowed to change at a time. The node i, which is the node that is chosen to change at the current time point, is called the *ego* node. It has the potential to interact with all other nodes in the network, $j \neq i$. These nodes j, are referred to as *alter* nodes, or simply *alters*. These nodes are acted upon by the ego node, and they only act when they become the ego node at a subsequent time point in the CTMC. For the ego node, i, in the current network state x, its objective function, which it tries to maximize, is written as

$$f_i(\boldsymbol{\beta}, x) = \sum_k \beta_k s_{ik}(x, \mathbf{Z}), \tag{3}$$

for $x \in \mathcal{X}$, the space of all possible directed networks with the n nodes, and $\mathbf{Z}$, the matrix of covariates. The vector $\boldsymbol{\beta}$ contains the parameters of the model with corresponding network and covariate statistics, $s_{ik}(x, \mathbf{Z})$, for $k = 1, \ldots, K$. Given the ego node, i, there are n possible steps for the actor i to take: either one of all current ties $x_{ij} = 1$ will be destroyed, a new tie will be created thatis currently $x_{ij} = 0$, or no change will occur.

The parameters, $\boldsymbol{\beta}$, correspond to various actor-level network statistics, $s_{ik}(x)$. According to Snijders (2001, p. 371), there should be at least two parameters included in the model: β_1 for the outdegree of a node, and β_2 for the number of reciprocal ties held

by a node. These effects should seem familiar to readers used to working with the classical exponential random graph model (ERGM) for networks. The outdegree represents the propensity of nodes with a lot of outgoing ties to form more outgoing ties (the "rich get richer" effect), and the reciprocity parameter measures the tendency of outgoing ties to be returned within a network. The statistics corresponding to these effects are written in terms of the edge variables x_{ij}, for $i \neq j$. In the `RSiena` software that we use to fit the SAOMs, there are over 80 possible parameters to add to the model. The formulas for the effects are provided in Ripley et al. (2017). The parameters, β_k, in the model can be split up into two groups: first, the structural effects, whose estimation depends only on the structure of the network, like the outdegree and reciprocity parameters mentioned above. The parameters are included when the researcher hypothesizes that they will model underlying mechanisms of network change. They hope to answer questions such as, "How does the existing network structure influence change in the network?" The second set of effects are referred to as the actor-level or covariate effects. These covariate effects also depend on the structure of the network, with the additional inclusion of node-level covariates of interest. The covariate effects are written in terms of the tie variables x_{ij}, but also in terms of the covariates, $\mathbf{Z}$. A table of some possible structural and covariate effects is given in Table 1. For a complete list of the network and covariate statistics that can currently be included in the objective function, see Ripley et al. (2017).
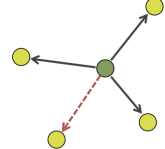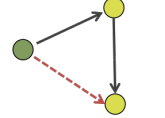
When node i is given the chance to change a tie, it attempts to maximize the value of its objective function $f_i(\boldsymbol{\beta}, x)$ as well as a random element, $U_i(x)$, to account for unknown attraction between nodes. The additional random element is included to account for any random, unexplainable change in the network ties. Snijders (2005) recommends that the $U_i(x)$ be random draws from a type 1 extreme value distribution. This distribution, which is also known as the log-Weibull distribution, has probability distribution function of:

$$g(u|\mu, \sigma) = \frac{1}{\sigma} \exp\left\{ -\left(\frac{u - \mu}{\sigma} + \exp^{-\frac{u-\mu}{\sigma}} \right) \right\}. \tag{4}$$

using μ for the mean parameter and σ for the scale parameter.

The probability density function for the type 1 extreme value distribution is shown in Figure 2. For mean $\mu = 0$ and scale $\sigma = 1$ as Snijders (2005) suggests is convenient because

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	$s_{i6}(x) = \sum_j x_{ij}\mathbb{I}(z_i = z_j)$	

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 (Ripley et al., 2016a).
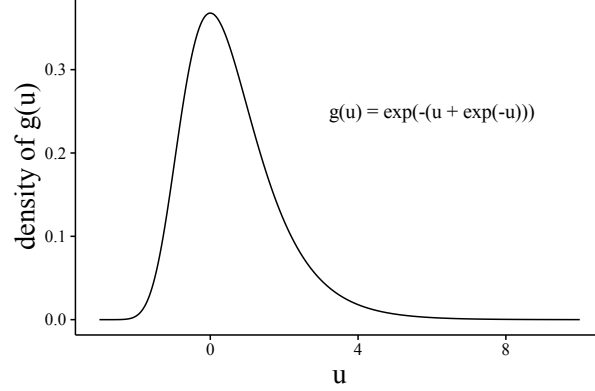
Figure 2: The probability distribution function for the type 1 extreme value distribution, also known as the log-Weibull or Gumbel distribution with location parameter $\mu = 0$ and scale parameter $\sigma = 0$.

it leads to a simple probability formula for the probability that actor i chooses to change its tie to actor j that can be written *only* in terms of the objective function. Let $p_{ij}(\boldsymbol{\beta}, x)$ be the probability that actor i chooses to change its tie to actor j. Next, we write the network x in its potential future state, $x(i \rightsquigarrow j)$, where the tie x_{ij} has changed to $1 - x_{ij}$. Then, the probility that the tie x_{ij} changes is

$$p_{ij}(\boldsymbol{\beta}, x) = \frac{\exp\left\{f_i(\boldsymbol{\beta}, x(i \rightsquigarrow j))\right\}}{\sum_{h \neq i} \exp\left\{f_i(\boldsymbol{\beta}, x(i \rightsquigarrow h))\right\}} \tag{5}$$

When $i = j$ in p_{ij}, the numerator represents the exponential of the value of the objective function when evaluated at the current network state. When the value of the objective function is high at the current state, the probability of not making a change in a microstep is also high. In the CTMC, when actor i may make a change, it chooses which tie $x_{i1,\ldots,x_{in}}$ to change at random according to the probabilities $p_{ij}(\boldsymbol{\beta}, x)$. The objective function and the resulting values of p_{ij} are combined with the rate function to fully describe the CTMC that is used to model network change in a SAOM.

3.1.3 Continuous Time Markov Chain (CTMC)

In the continuous-time Markov chain literature, see for instance Yin and Zhang (2010), chains are characterized by their *generator* or *intensity* matrix $\mathbf{Q}$. This matrix describes the rate of change between two states of the CTMC process, and the rows of this matrix

always add to zero. For directed networks with binary edge variables like the ones we will be working with, there are a very large number of possible states for a directed network with n nodes. We denote the state space as $\mathcal{X}$, a set which contains $2^{n(n-1)}$ states: there are two possible states for an edge, $\{0, 1\}$, and there are $n(n-1)$ edge relationships because the network is directed and we exclude self-ties. The intensity matrix for a CTMC in a SAOM is then a square matrix of dimension $2^{n(n-1)} \times 2^{n(n-1)}$. Only one tie changes at a time in the CTMC, resulting in $n(n-1)$ reachable states from the current network state. 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 entries represent the possible states that are one edge different from a given state, while the additional non-zero entry is for the state to remain unchanged. All other entries in a row are structural zeroes because those network states cannot be reached from the current state in a single change.

The two pieces of a SAOM, the rate function/parameter and the objective function, each contribute to the entries of the intensity matrix to describe the rate of change between two network states. 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:

$$
q_{bc} = \begin{cases} q_{ij} = \alpha_m p_{ij}(\boldsymbol{\beta}, x^b) & \text{if } x^c \in \{x^b(i \rightsquigarrow j) | \text{ any } i \neq j \in \{1, \ldots, n\}\} \\ 0 & \text{if } \sum_i \sum_j |x_{ij}^c - x_{ij}^b| > 1 \\ -\sum_{i \neq j} q_{ij} & \text{if } x^b = x^c \end{cases}
$$

Thus, the rate of change between any two states, x^b and x^c, 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. This matrix $\mathbf{Q}$ is the foundation for estimation of a SAOM.

3.2   Fitting Models to Data

To fit a SAOM to observations of a dynamic network, we use the package `RSiena` (Ripley et al., 2016a). This package uses simulation methods to estimate parameter values using either the method of moments or maximum likelihood estimation. In this paper, use the method of moments estimation because the theory behind it was established in Snijders

(1996), while the maximum likelihood estimation methods were not fully established until Snijders et al. (2010), though `RSiena` contains capabilities to use maximum likelihood estimation. We also use the score function method for estimating the derivatives of the expected values, as opposed to the finite differences method, both of which are outlined in detail in Snijders (2016).

For the score function method, the SIENA software uses a Robbins-Monro algorithm (see Robbins and Monro (1951)) to estimate the solution of the moment equation

$$E_\theta S = s_{obs}$$

where θ is the vector of rate and objective function parameters, and s_{obs} is the observed vector of model statistics, S. The entire algorithm is provided in Snijders (2016).

There are three phases in the the SIENA algorithm, as described in Ripley et al. (2017); Snijders (2016). The first phase performs initial estimation of the score functions for use in the Robbins-Monro procedure for method-of-moments estimation. The second phase carries out the Robbins-Monro algorithm and obtains estimates of the parameter values through iterative updates and simulation from the CTMC at current parameter values. The third phase uses the parameter vector estimated in phase two to estimate the score functions and covariance matrix of the parameter estimate, and also carries out convergence checks. In each of the the first two phases, the estimation procedure also uses "microsteps" that simulate from the model as it exists in its current state in order to update either the score functions or the parameter estimates. These simulated microsteps are observed instances of the continuous-time Markov chain that is the backbone of the stochastic actor-oriented model. In Section 4, we further explore these phases in the SIENA method-of-moments algorithm through visualization, bringing them out of the "black-box" and into the light.

3.3   Model Goodness-of-Fit

The `RSiena` software that fits the models to data also includes a goodnes-of-fit function for examining model fit, `sienaGOF()`. This function "assess[es] the fit of the model with respect to auxiliary statistics of networks" (Ripley et al., 2017, p. 53). Examples of auxiliary statistics include the out- or indegree distribution on the nodes, with the option for users to input their own statistics to examine. The goodness-of-fit is evaluated as follows:

16

1. The auxiliary statistics are computed on the observed data and on N simulated observations from the model. Typically, $N = 1000$.

2. The mean vector and covariance matrix of the statistics on the simulations from the model are computed.

3. The Mahalanobis distance from the observed statistics to the distribution of the simulated statistics is computed using the mean and covariance found in step 2.

4. The Mahalanobis distance from each of the N simulations to the same distribution is computed, and the Mahalanobis distance of the observed data is compared to this distribution of distances.

5. An empirical p-value is found by computing the proportion of simulated distances found in step 4 that are as large or larger than the Mahalanobis distance from the data. A SAOM is thus considered a good fit if p is large.

The `plot.sienaGOF()` function allows us to visualize this fit. This function draws a box plot and a violin plot at each value of the statistic of interest observed in the simulations: on the x-axis, the out(in)degrees observed in the data $(0, 1, 2, 3, \dots)$, and on the y-axis, the cumulative number of times that out(in)degree value appears in the simulations and the data. In order to compare the distribution of the counts of nodes with the specified degree as calculated on the simulated networks to the counts observed in the true data, red points connected by red lines representing the observed values are superimposed on the boxplots. If the red points lie "well within" the simulated values, the model is a good fit to the data. This plot is not shown because it is not intuitive for understanding network data. Boxplots separated by the many outdegree values observed do not communicate *how* the nodes are connected, just that some have more connections than others. In order to understand the fit of the model, we should try to understand how well the model captures the *overall* structure of the network, not just one or two summary measurements of that structure. In Section 4.2, we propose a new way of visualizing goodness-of-fit that uses the traditional node-link diagram to visualize the entire network instead of numerical summaries of the the network.
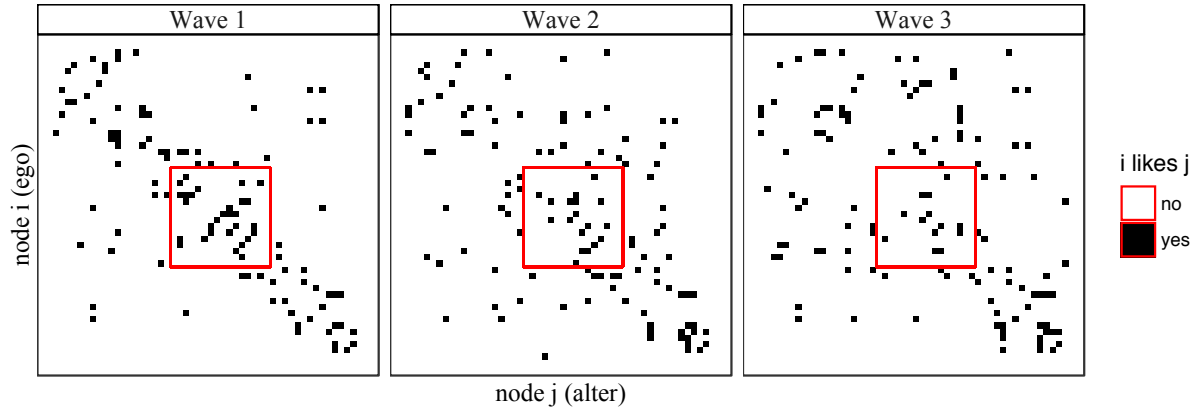
3.4 Example Data

Figure 3: A visualization of the adjacency matrices of the three waves of network observations in the "Teenage Friends and Lifestyle Study" data. The subset we will be using is outlined in red.
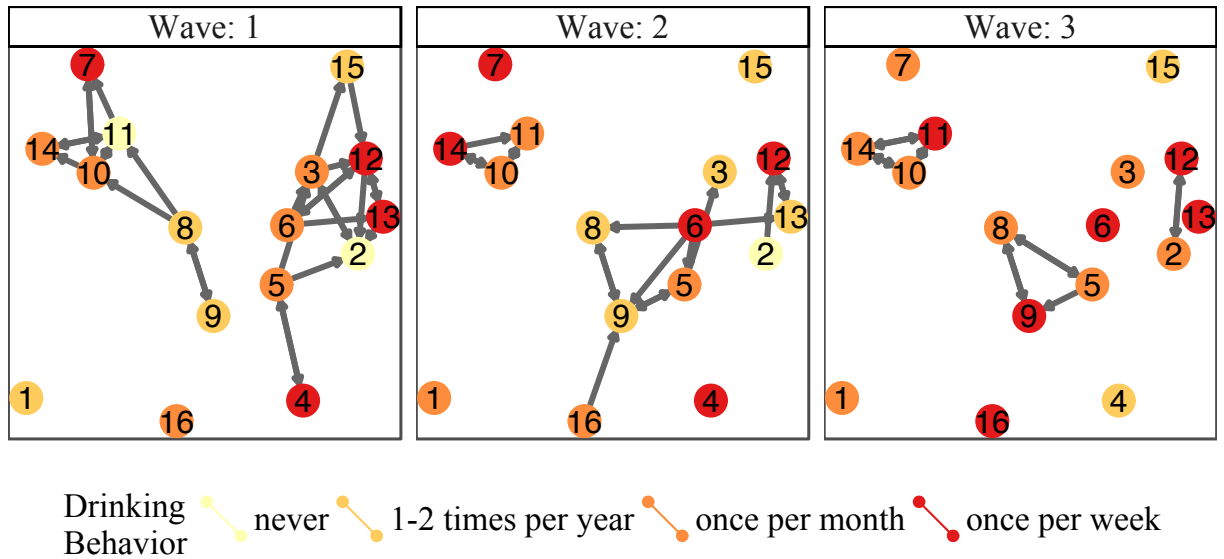

Figure 4: The smaller friendship network data we will be modelling throughout the paper. In the first wave, we can see there are two large, separate friend group. By the second wave, three students with heavier drinking behavior have separated from their group, while the others become members of the other group. By the third wave, the large group has almost completely broken up. We want to capture these changes with our SAOMs.

To guide our visual exploration of stochastic actor-oriented models, we use two data sources. The first is a subset of the 50 actor dataset from the "Teenage Friends and Lifestyle Study" that is provided on the `RSiena` webpage. These data come from Michell and Amos (1997), and we chose to only work with a subset of the data to make network visualizations less busy and to make any changes in the network more noticeable. To determine which subset to select, we visualized all waves of the full network using the adjacency matrix visualization approach, which we show in Figure 3. This adjacency matrix visualization is different from the one in Figure 1 because it does not show the node IDs on the axes. The network in Figure 3 is much larger than our toy data example, so we remove the node labels to remove clutter. In all three adjacency matrices, the ego and the alter nodes are ordered by node ID, 1-50, which were determined arbitrarily by the relationships in wave 1. The subset we selected is outlined in red in the visualization. This 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 observations in Figure 3. For model fitting, we condition on wave 1 and estimate the parameters of the models from the second and third waves. We will also be working with one actor level categorical covariate, drinking behavior. This variable has five values in the original data: (1) does not drink, (2) drinks once or twice a year, (3) drinks once a month, (4) drinks once a week, and (5) drinks more than once a week. The network and the actor covariate values are visualized using a node-link diagram in Figure 4. In the node-link diagram, the nodes are colored according to the drinking behavior of that student. Over time, we can see that the students tend to drink more and become increasingly isolated into smaller groups. An analysis of this type of data with a SAOM should capture these dynamics in a way that allows the researcher to draw conclusions about the nature of the network and behavioral forces at play.

The second data example we use is a collaboration network in the United States Senate during the 111th through 114th Congresses. These sessions of congress correspond to the years of Barack Obama's presidency, from 2009-2016.[1]. In this network, ties are directed

---

[1] Details of how this data can be downloaded are provided by Franois Briatte at `https://github.com/briatte/congress`

from senator i to senator j when senator i signs on as a cosponsor to the bill that senator j authored. There are (somewhat surprisingly) many hundreds of ties between senators when they are connected in this way, so we simplify the network by computing a single value for each senator-senator collaboration called the *weighted propensity to cosponsor* (WPC). This value is defined in Gross et al. (2008) as

$$WPC_{ij} = \frac{\sum_{k=1}^{n_j} \frac{Y_{ij(k)}}{c_{j(k)}}}{\sum_{k=1}^{n_j} \frac{1}{c_{j(k)}}} \tag{6}$$

where n_j is the number of bills in a congressional session authored by senator j, $c_{j(k)}$ is the number of cosponsors on senator j's k^{th} bill, where $k \in \{1, \ldots, n_j\}$, and $Y_{ij(k)}$ is a binary variable that is 1 if senator i cosponsored senator j's k^{th} bill, and is 0 otherwise. This measure ranges in value from 0 to 1, where $WPC_{ij} = 1$ if senator i is a cosponsor on every one of senator j's bills and $WPC_{ij} = 0$ if senator i is never a cosponsor any of senator j's bills.

Because we require binary edges for our models, we focus only on very strong collaborations. For our senate collaboration networks, x, edges are defined as

$$x_{ij} = \begin{cases} 1 & \text{if } WPC_{ij} > 0.25 \\ 0 & \text{if } WPC_{ij} \leq 0.25. \end{cases}$$

The networks we constructed for the four senates during President Obama's administration are shown in Figure 5. In Section 4, we fit several stochastic actor-oriented models to these data sets use those models to guide our further exploration of SAOMs.

4   Model Visualizations

Every good data analysis includes both numerical and visual summaries of the data, so why restrict model description and diagnostics to numerical summaries? The concept of model visualization was developed to complement traditional model diagnostic tools. Typically, numerical summaries such as R^2 are used to assess model fit, and the occasional visualization, like a residual plot, are used to determine how well the model fits the data.
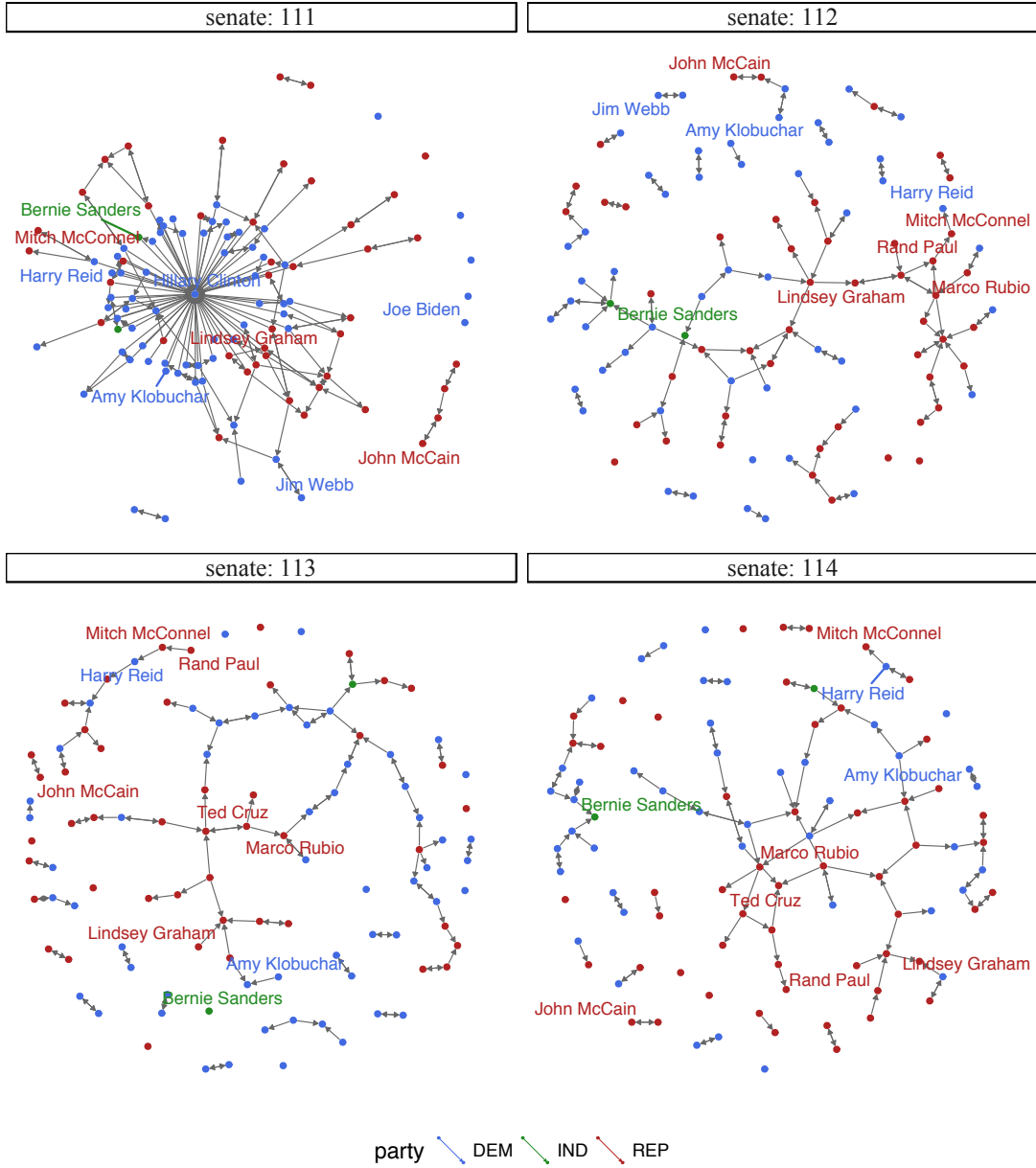
Figure 5: The collaboration network in the four senates during the Obama years, 2009-2016. Edges are shown only if the weighted propensity to cosponsor from one senator to another is greater than 0.25. We use the Fruchterman-Reingold algorithm to layout the node-link diagram.

Wickham et al. outline three separate ideas, each of which are can be referred to simply as the "model": the model family, the model form, and the fitted model. The latter is primarily what one thinks of first when considering a model in a data analysis, where a specified model is fit to data, and parameter estimates and other numerical summaries, such

as R^2 are reported. In the context of SAOMs, the fitted model contains the form of the rate and objective functions, the estimated rate parameters, and the estimated objective function parameters. The model form describes the the model *before* the fitting process, defining which parameters are in the model within the context of the larger model family. In SAOMs, the model form includes description of the rate and objective functions and the variables therein that describe how the network evolves over time. Finally, the model family is the broadest description of the model. This is the type of model that you wish to fit to the data, and is chosen based on the problem, data, and knowledge at hand. For example, we chose to use a SAOM to model network data over an exponential random graph model (ERGM) because we believe that actor-level variables effect network structure and formation, and we wanted to model the network changes over time.

The model family, the model form, and the fitted model can each be visualized according to the three principals of model visualization: we can view the model in the data space, visualize collections of models, and explore the process of fitting the model, not just the end result. Since we have already decided on our model family, SAOMs, we now shift our focus to the fitted model and the model form. Specifically, we want to learn more about how the model form we choose affects the fitted model by using our example data sets and our visualization toolbox. We begin by introducing the five models that we fit to our example data. Next we use the five models to guide our visual explorations of SAOMs. We first use novel tools and ideas to view a SAOM in the data space of a dynamic network. We then explore collections of the same models fit many times to the example data to see how the simulation processes in `RSiena` affect the model fits. Finally, we look behind the scenes and into the individual steps of the continuous time Markov chain to learn more about how this "hidden data" mechanism works and how it results in a fitted model.

4.1 The Models

We first consider the 16 actor subset of the teenage friends and lifestyle data available on the `RSiena` website (Ripley et al., 2016a). To this data, we fit three different SAOMs. Each SAOM used a simple rate function, α_m, and an objective function with two or three parameters. The first model, M1, contains the absolute minimum number of parameters
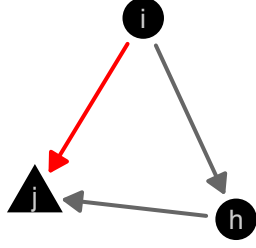
(a) Realization of a jumping transitive triplet, where i is the focal actor, j is the target actor, and h is the intermediary. The group of the actors is represented by the shape of the node.
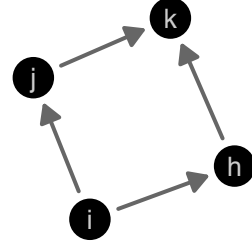
(b) Doubly achieved distance between actors i and k.

Figure 6: The additional network effects included in our models fit to the friends data. On the left, a jumping transitive triplet (JTT). On the right, a doubly achieved distance between i and k.

in the objective function $f_i(x)$:

$$f_i(x)^{M1} = \beta_1 s_{i1} + \beta_2 s_{i2},$$

where s_{i1} is the density network statistic and s_{i2} is the reciprocity network statistic for actor i at the current network state x. The second and third models, M2 and M3, contain one additional parameter each in the objective function which were determined by a Wald-type test provided in the RSiena software to be significant, with p-values less than 0.05 (Ripley et al., 2016b). The M2 model contains an actor-level covariate parameter, and the M3 model contains an additional strutural effect in the objective function.

$$f_i(x)^{M2} = \beta_1 s_{i1} + \beta_2 s_{i2} + \beta_3 s_{i3}$$
$$f_i(x)^{M3} = \beta_1 s_{i1} + \beta_2 s_{i2} + \beta_4 s_{i4},$$

where $s_{i3} = \sum_{j \neq h} x_{ij} x_{ih} x_{hj} \mathbb{I}(z_i = z_h \neq z_j)$, and $s_{i4} = |\{j : x_{ij} = 0, \sum_h x_{ih} x_{hj} \geq 2\}|$. These statistics are known as the number of jumping transitive triplets and the number of doubly achieved distances two effect, respectively. The first statistic emphasizes triad relationships that are formed between actors from different covariate groups, while the other emphasizes indirect ties between actors. The covariate groups are determined by the stu-

dent's drinking behavior, the values of which are numeric and mean-centered. The four values in our data are {never, 1-2 times per year, once per month, once per week}, which after being converted to numeric and mean-centered become $\{-0.8125, -1.8125, 0.1875, 1.1875\}$. The two additional effects are visually represented and further described in Figure 6a and Figure 6b, respectively.

In Section 4.3.3, we also fit models M1, M2, and M3 to the senate collaboration data for comparison. For fitting M2 to the senate data, we use the number of bills authored by each senator as the node covariate for the jumping transitive triplet variable. In terms of the senate data, then, the value of β_3 should dictate how willing a senator is to coauthor a bill on which the author of the bill has a different level of authorship, assuming there is an intermediary between the two who has the same authorship level as the first.

Due to the intractability of SAOMs, it is difficult to know for certain how to interpret a fitted value of a parameter. We can make educated guesses based on the definition of the effect and the sign of the fitted value, but a direct interpretation is not always possible. But, by exploring these models more visually, we aim to understand these effects, their interpretations, and the model fitting process better.

4.2   View the model in the data space

The first way we hope to better understand stochastic actor-oriented models is by viewing the model(s) in the data space. In Wickham et al., they happen to define the *data space* as "the region over which we can reliably make inferences, usually a hypercube containing the data" (Wickham et al., 2015, p. 206). But what does this definition mean for network data? For dynamic social networks, there are a few different data "spaces":

1. the actors and their corresponding covariates,

2. the edges and their correponding variables that describe the relationships between the nodes, and

3. the time, both the continuous unobserved time and the discrete observed time points, over which the network evolves

These three data pieces can be visualized together in various ways. The traditional node-link visualization uses one of many algorithms to layout the actors as points in 2D space, then draws segments connecting the points in 2D if there is an edge between two nodes, and draws nothing otherwise. The time aspect can be visualized by drawing each network observation in time and placing the observed timepoints side-by-side.

Because longitudinal network data consist of three different "spaces" of data, viewing the model in the data space can depend on which aspect of the *model* and the *data* we are interested in viewing. Incorporating data covariates into the network structure allows us to assess whether the ties between nodes are affected by how nodes behave over time. In this instance, we would want to view predictions over time. A SAOM can also model behavior change over time, taking both the node and edge information into account. In this case, a plot of predicted covariate values over time would put the model into the time and node data space. Most likely, however, is that we would want to view all of the data spaces simultaneously.

One tool that can bring the node, edge, and time data spaces together in this way is the R package geomnet (Tyner and Hofmann, 2016). Different visual features in the node-link diagram can be tied to the underlying node or edge data. The color, size, and shape of the points can be used to represent variables in the node data, while the color, linewidth, and linetype of the lines between points can be used to represent the edge variables. In a social network, node data might be age, gender, and occupation of the person in the network, and edge data might be length of connection between two people, how the people first met (school, work, church, etc.), and how often they interact, and we can view the network at different timepoints side-by-side to see its evolution. Pulling all of this information together with geomnet allows the entire data space to be viewed at once.

To demonstrate, we use geomnet to visualize the connections in the 111th United States Senate at two different points: when Hillary Clinton was in the senate, and after she left to become Secretary of State. Clinton was only in the 111th senate for 17 days, from January 3, 2009, to January 20, 2009, when she was in the middle of her second term as senator from New York. In that time, she authored two bills and was a cosponsor on 17 other bills. With Clinton included in the node-link diagram, the senate looks much more highly
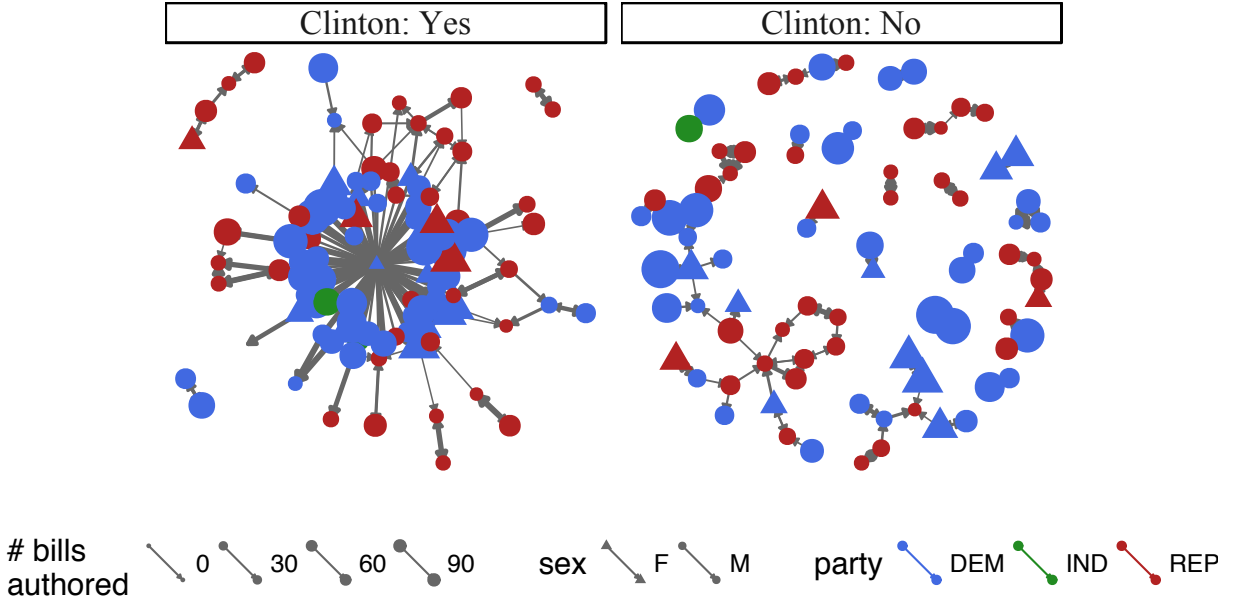
Figure 7: The 111th Senate at two discrete time points: while Clinton was in the senate in the first few weeks of 2009 (on the left) and after she left the senate to become Secretary of State (on the right). We put some potential model covariates, sex and party of the senator and the number of bills they authored in the data space through the shape, color, and size of the nodes, respectively. We also map the strength of the tie, the WPC, to the width of the edge between two nodes. Thicker lines implies higher propensity to collaborate. In addition, senators with no ties higher than $WPC = 0.25$ are not shown.

collaborative than it does without her in the diagram. We can compare the number of bills authored throughout the senate by mapping the size of the node to the numer of bills authored by that senator. We also map shape to sex, and keep color mapped to party of the senator. In addition, we can see the strength of the tie by mapping the linewidth of the edge to the WPC value between the two senators. In this single visualization, we have viewed node information (number of bills authored by a senator and the sex and party of that senator), edge information (the direction and strength of ties between two senators), and time (before and after Clinton left the senate).

Another way to view the model in the data space is through simulation from the model. No single network alone simulated from a SAOM is going to look like the data or represent the model, just as no single value simulated from the standard normal distribution will look like a bell-shaped curve. Statisticians would prefer to look at a sample, or at
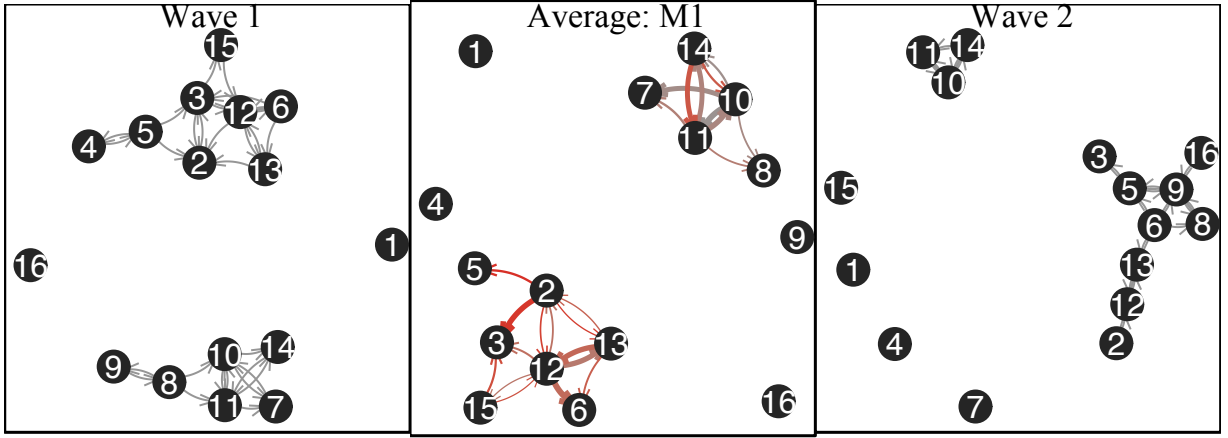
Figure 8: On the left, the first wave of observed data that is conditioned on in the model. On the right, the second wave of observed data. In the middle, a summary network from the first model fit to the data. This summary network represents 1,000 simulations of wave 2 using the values from the simple fitted model M1.

least a summary statistic or two, so it would therefore better to visualize many simulations together. From a statistician's point of view, a stumbling block with statistical network models generally is the lack of an "average network" or "expected network" value. Statisticians frequently rely on averages and expected values in data analyses, but statistical network models, especially those as complex as SAOMs, lack a single, intuitive expected value measure. We could talk about expected values of parameters, but the parameters can be hard to interpret. Expected values of parameters are important, but if they cannot directly tell us anything about our dependent variable (networks in this case), they lose some value to us. Furthermore, there is no way to talk about the expected value of an observation simulated from a statistical network model. How then, can we arrive at an "average" network? We answer this question through visualization. For network data, one way we view an "average" network is through a summary network drawn using the traditional node-link diagram. In Figure 8, we show an *average network* created with 1,000 simulations of the second wave of the network from Model 1. To make this average network, we first simulated 1,000 wave 2 and wave 3 observations of our small friendship example data from model M1, for which parameters had previously been estimated. We
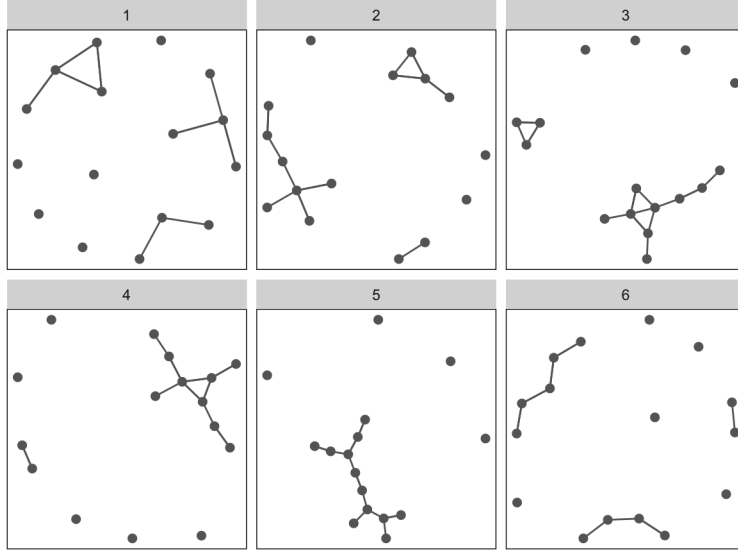
Figure 9: A small lineup of node-link diagrams showing the second wave of our small friendship network among five networks simulated from model M1.

then combine the 1,000 instances of wave 2, and count up the number of times each edge appears in a simulation. Then, we combine these 1,000 networks into a single network with edgeweight equal to the proportion of time that edge appears in our 1,000 simulations. This weighted edgelist is the network we draw using the node-link diagram. An edge is only drawn in the average network if it appears in more than 5% of the simulations (in at least 51 of the 1000 simulations), with edges that appear more frequently emphasized by thicker linewidths and a darker color, representing the proportion of times appeared in the model simulations. On either side of the average network in Figure 8, we show the actual data, wave 1 on the left, and wave 2 on the right. We can see that the structure of the average network is much more similar to the first wave than to the second wave. However, the simulations are supposed to represent the second wave of data, which is shown on the right in Figure 8. This is an indication that the simple model, M1, is doing a very poor job of capturing the change mechanism from the first to the second wave of observation. The average network can thus be used to help determine model goodness-of-fit. Because the the average network looks more like the first wave than the second wave, we can use the visualization in Figure 8 as evidence of poor model fit.

Another potential goodness-of-fit visualization that places the model in the node and edge data space is a lineup like those proposed in Buja et al. (2009). A *lineup* "asks the

witness to identify the plot of the real data from among a set of decoys, the null plots, under the veil of ignorance" (Buja et al., 2009, p. 4369). It can be thought of like a police lineup, where the "suspect" is in a lineup among several innocent lookalike fillers, and a witness picking the suspect out of the lineup is considered evidence against the suspect. In data and model visualization, the "suspect" is a plot of the true data, while the "filler" is composed of several plots of mock data, simulated from a hypothesized model. If the true data stands out among the simulated data, that is taken as evidence of poor model fit, whereas if the true data is difficult to identify among the simulated data, that is taked as evidence of good model fit. An application of the lineup protocol can be found in Hofmann et al. (2012), where the authors examine, for instance, the differences between polar and cartesian coordinates for plotting categorical data, and density plots and box plots for determining distributional differences. They pose questions to experiment participants such as, "Which plot is most different from the others?" for the first example and, "In which plot is the blue group furthest to the right?" for the distributional differences. The data visualizations examined in Hofmann et al. (2012) are less complex than a node-link network diagram. What questions should we ask for network data visualizations? Asking participants to identify the most different plot may be difficult. In the network lineup shown in Figure 9, the second wave of the small friendship network is shown among five simulated networks from model M1 using parameter values estimated from the data. What makes these plots "different"? It seems possible to argue for any one of the six plots in Figure 9 as most different. So, we guide participants to look at the overall structure of the graphs to determine which has the most and least complex structure. The least complex plot, number six, has no triplets, while the most complex plot, number three, has three triplets, and in fact, plot three is the data. We have found in an experiment where all the node-link diagrams shown are based on simulated data, one observation from an "alternative" model and the rest from a "null" model, the triangular shape of the triplets stands out the most to participants. We can use the lineup protocol to help better understand SAOMs because it allows us to view the model in the data space by placing observations the model side-by-side with the data and examining the differences. This can also help us determine the significant structural effects, if any, of the parameters in the model on observations simulated from the

model. The triangular shape mentioned above becomes more prevalent when a transitive triplet parameter in included in a model, but it does not always cause triplets to form in the simulated data. This requires more investigation with larger data sets and more complex models, but is promising for the development of additional goodness-of-fit measures for network models.

4.3   Visualizing collections of models

There are many possible ways to collect models together. We could look at the same models fit to different data, different models fit to the same data, or because of the nature of SAOMs, we could fit the same models to the same data many times to see how the simulations change. For the SAOMs, we decided that there were four collections that were most important:

1. the collections resulting from exploring the space of all possible models;

2. the collections we get when varying model settings;

3. the results from fitting the same model form to different data;

4. the results from fitting the same model to the same data many times.

We chose these four collections because they each explore something different about SAOMs. The first takes the many dozens of parameters available to include in a SAOM into account, which easily translates into the second by showing how those many parameters affect the fitted models. Then, we look at how the same model looks when fit to different sets of dynamic network data. Finally, we look at the results from fitting the same model to the same data because the MCMCs and CTMCs that make up a SAOM lead to different parameter estimates every time, so it important to see how the results can vary.

4.3.1   Exploring the space of all possible models

The `RSiena` manual contains over eighty possible effects to include in the model. In order to select parameters to include in the models for our example data, we searched through the possible effects available to model given the data structure to find *significant* effects.
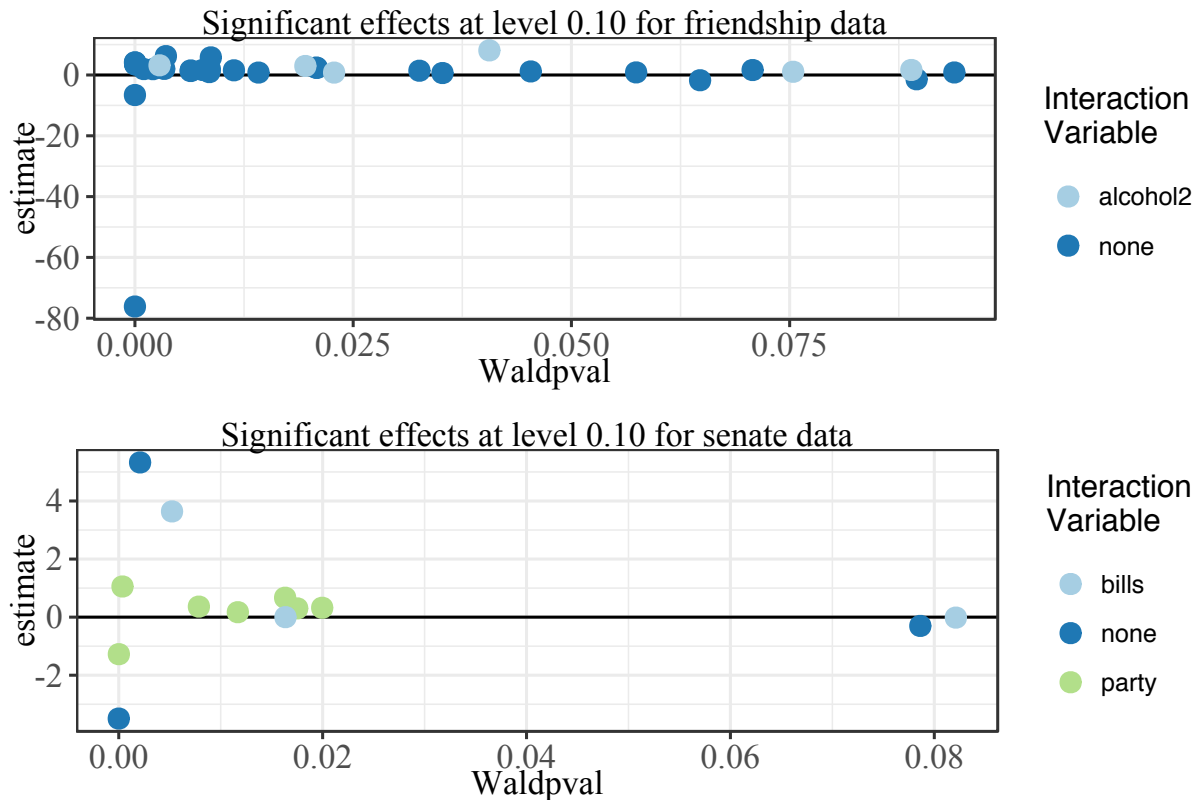
Figure 10: Significant effects for the two data sets, at a significance level of 0.10 or lower as calculated by the Wald-type test available in the SIENA software.

We tested for significance using the Wald-type tests built into `RSiena` for one-at-a-time effects testing. We start with the outdegree and reciprocity measures as the foundation of the models we fit, then add one evaluation effect, fit the model, test the additional effect for significance, and repeat for all possible parameters to add to the model. We performed this procedure for both the small friendship and the senate collaboration data. The results for significant effects are shown in Figure 10. We see in both the friendship data and the senate data results that most of the significant effects have absolute value less than ten. In addition, the p-values for the effects from the friendship data are more spread out than the p-values for the senate data, which are concentrated at about 0.02 or less. This may suggest that larger data sets tend to result generally in smaller p-values, just like a larger sample size results in smaller p-values in a t-test.

4.3.2 Varying model settings

We have varied model settings already by choosing models M1, M2, M3 to fit to our small friendship network data set. In Section 4.3.4, we fit these models to the data 1,000 times, and in this section, we explore simulations from these three models given the mean values of from the 1,000 fitted parameter values as the parameters in our models. From each of these three models using the means of parameter estimates as our fixed parameter values, we simulated 1,000 observations from each of the three models. In this process, we condition on the first friendship network observation, and the second and third observations are simulated from the SAOM models with the given parameter values. From these simulations, we first create a visualization that represents an average network.

To create the average network visualization shown in Figure 11, we follow the same procedure as in Section 4.2, counting occurrences of each possible edge in the simulations, resulting in a summary network with weighted edges representing the number of times an edge appeared in the simulated wave 2 when simulating from the SAOM 1,000 times. As in Figure 8, edges only appear in the average networks if they appear more than 5% of the time in the simulations. In Figure 11, we show the "average" network from the three models we fit and the first and second waves of data. Comparing the three averages to waves 1 and 2, we see that they have very similar structure to wave 1. Model 2, which included the transitive triplet parameter, seems to have created a larger connected component overall than models 1 and 3. In particular, if we look at the group of nodes $\{10, 11, 14\}$, we see they are very strongly connected within the three average networks, and they are completely separate from the other nodes in the true wave 2. None of the three average networks show node 16 gaining ties as it does in wave two, nor do they show nodes 4 and 7 becoming isolated. In Model 2, however, the ties to node 7 appear much weaker than in Model 1 or Model 2, suggesting that of the three, Model 2 may be the best fit for our data.

4.3.3 Fitting the same model to different data

As mentioned in Section 4.1, we fit the models M1, M2, and M3 to both the small friendship network and the senate collaboration network. We fit each model to the friendship data 1,000 times, and to the senate data 100 times. The means and standard deviations of the
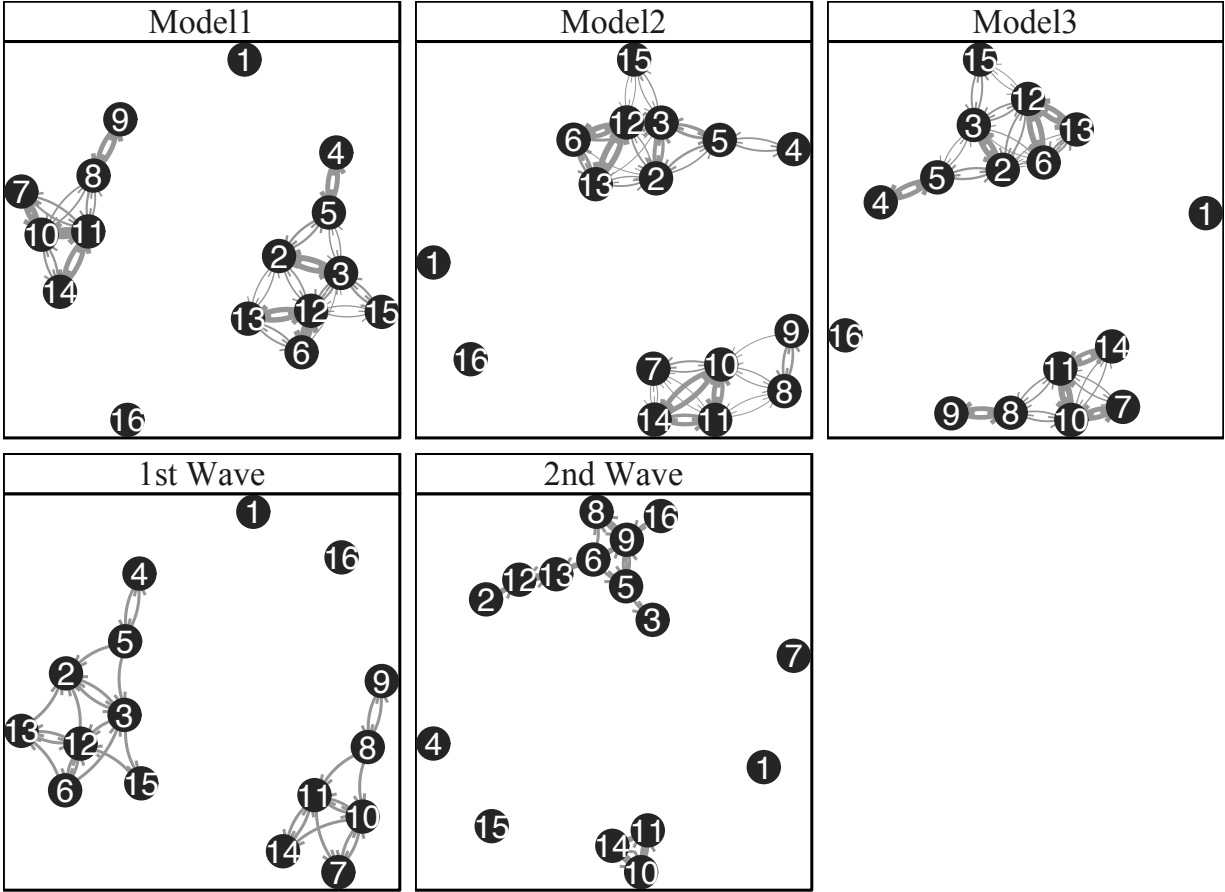
Figure 11: The node-link diagrams from the three "average" networks that we calculated are in the top row, and the true wave 1 and wave 2 data are shown in the bottom row above. There is some difference between the three models, but overall, these three models cannot capture the structure in the true second wave of data.

parameter estimates for each combination of model and data are given in Table 2

Looking at Figure 12 and Table 2, we see a few patterns in the estimates from both models. First, we can see in both the table and the density of the estimates that the same relationship between the outdegree parameter, β_1, and the reciprocity parameter, β_2. In both data sets and across all three models, the estimates of β_1 are all negative and hover between -5 and -3, while the estimates of β_2 are all positive and hover between four and five. This suggests that in both data sets, nodes are *discouraged* from forming ties that

	Friendship Data			Senate Data		
	M1	M2	M3	M1	M2	M3
α_1	4.660 (0.059)	5.176 (0.068)	4.712 (0.060)	3.344 (0.016)	3.349 (0.016)	3.340 (0.016)
α_2	1.930 (0.026)	2.017 (0.028)	1.979 (0.027)	2.480 (0.017)	2.487 (0.015)	2.483 (0.014)
α_3	–	–	–	2.221 (0.017)	2.227 (0.017)	2.224 (0.016)
β_1	-3.597 (0.033)	-4.104 (0.038)	-3.589 (0.035)	-4.979 (0.027)	-4.993 (0.025)	-4.987 (0.021)
β_2	4.149 (0.050)	4.277 (0.052)	4.230 (0.050)	4.954 (0.046)	4.974 (0.040)	4.970 (0.035)
β_3	–	3.209 (0.053)	–	–	-1.175 (0.789)	–
β_4	–	–	-7.582 (1.746)	–	–	-1.048 (0.486)

Table 2: The means (standard deviations) of parameter values estimated from repeated fittings of $M1, M2, M3$ to the small friendship network and the senate collaboration network. Each model was fit 1,000 times to the friend data, while each model was fit 100 times to the senate data.

are outgoing without being reciprocated, while also being *encouraged* to form outgoing ties to nodes that tie to them. In both data sets, people seem to want to have reciprocated relationships: teenage girls want be friends with other girls that reciprocate their friendship, and senators want to coauthor bills with senators who have also been coauthors on their bills. We explore the relationship between β_1 and β_2 further in Section 4.3.4.

The inclusion of β_3, the jumping transitive triplet parameter, for the friendship data had a noticeable effect on the other parameters in the model. The same cannot be said for the inclusion of β_3 for the senate data. The covariate used in the senate data for the jumping transitive triplet calculation was the number of bills authored by the ego node in the given year. The number of bills authored by a senator varies wildly, from no bills to 114 bills authored in two years, so this effect could simply be nonsensical for the senate data, since senators are less likely to have the same number of bills authors than teenage girls are to have the same drinking behavior. Looking at the estimates of β_4, we see that the estimates for the senate data are near zero, suggesting this effect, which considers indirect ties, is not important for the senate data: indirect relationships do not describe the senate collaboration structure as much as they do the teenage friendship structure.
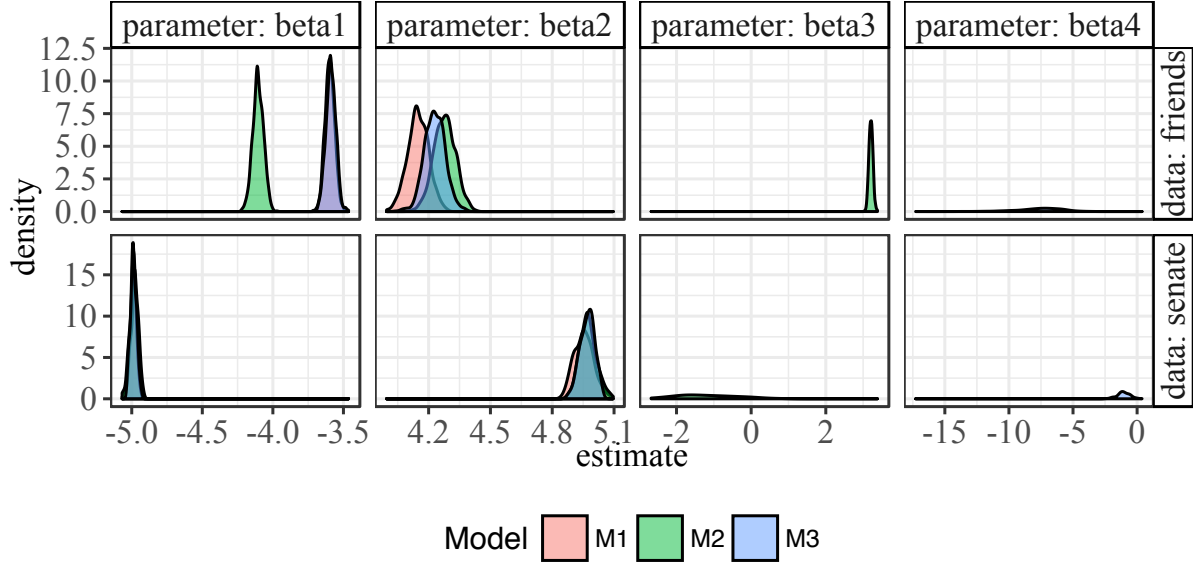
4.3.4   Fitting the same model to the same data

Figure 12: Density plots of the repeated estimates from fitting models M1, M2, and M3 to the friendship and senate example data. Note that the rate parameters are excluded since the two data sets have different numbers of waves. The first two parameters, outdegree and reciprocity, have the same relationship for both data sets: the outdegree parameter estimate is strongly negative, while the reciprocity estimate is strongly postitive. For the friendship data, the inclusion of the transitive triplet parameter has strong effect on the estimates of the other two parameters, while it does not affect the estimates of the other two parameters for the senate data.

To our small friendship network, we fit three models, M1, M2, and M3, using `RSiena` 1,000 times each. We then looked at the distribution of the fitted values, which are shown in Figure 13. We can see from these distributions that the inclusion of the jumping transitive triplet parameter, β_3 is obviously affecting the distributions of the other four parameters included in all models, α_1, α_2, β_1, and β_2. When β_3 is included, its estimate is postitive, meaning that friendships between two girls with different drinking behaviors tend to form when there is an intermediary who is already friends with the two girls. The inclusion of this parameter leads to increases in the rate parameters' estimates, suggesting that encouraging the transitive triplet behavior means that the girls would also change friends more frequently. The outegree parameter, β_1 decreases when β_3 is included, while the reciprocity parameter, β_2 increases. This implies the girls in the data prefer to form closer friend groups, as indicated by reciprocated ties and jumping transitive triplet formation,
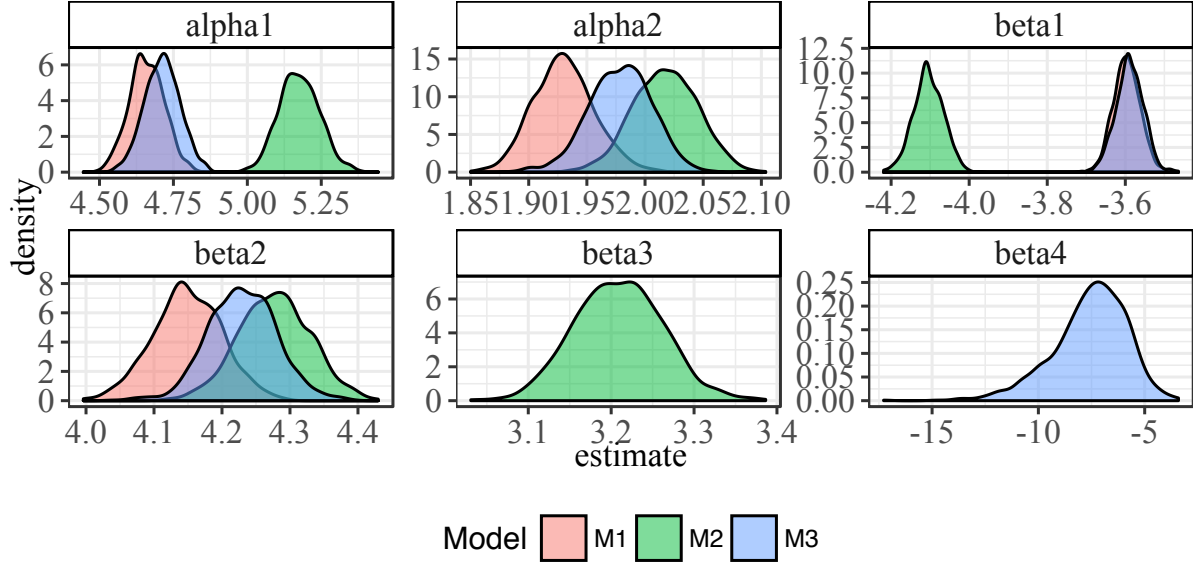
Figure 13: Distribution of fitted parameter values for our three SAOMs. The inclusion of β_3 or β_4 clearly has an effect on the distribution of the rate parameters, α_1 and α_2.

as opposed to being popular and having many friends. Having many friends who do not reciprocate is discouraged by M2. In comparison with models M1 and M3, model M2 typically has higher estimates of the rate parameter, meaning that the inclusion of the covariate statistic in the model leads to higher estimates of the number of times, on average, a node gets to change its ties. It is not clear, however, that the addition of a parameter to the objective function *should* effect the estimation of the rate parameters, so we continue to explore the collection of parameter estimates.

To further investigate the odd relationship between the parameter values, we look at correlations between each of the parameter estimates in each model. In Figure 14, we examine correlations between each of pair of parameters within each model and overall. The strongest correlation within each model is between β_1 and β_2, with absolute value of correlation between those two parameter values greater than 0.90 in all three models. The β_1 parameter is also highly correlated with the β_3 parameter within model M2, but it is not as highly correlated with the β_4 parameter in model M3. It might therefore be advisable to consider only models that either allow β_1, or β_2. Looking at the high correlation with α, we might swiths to a model without β_1.
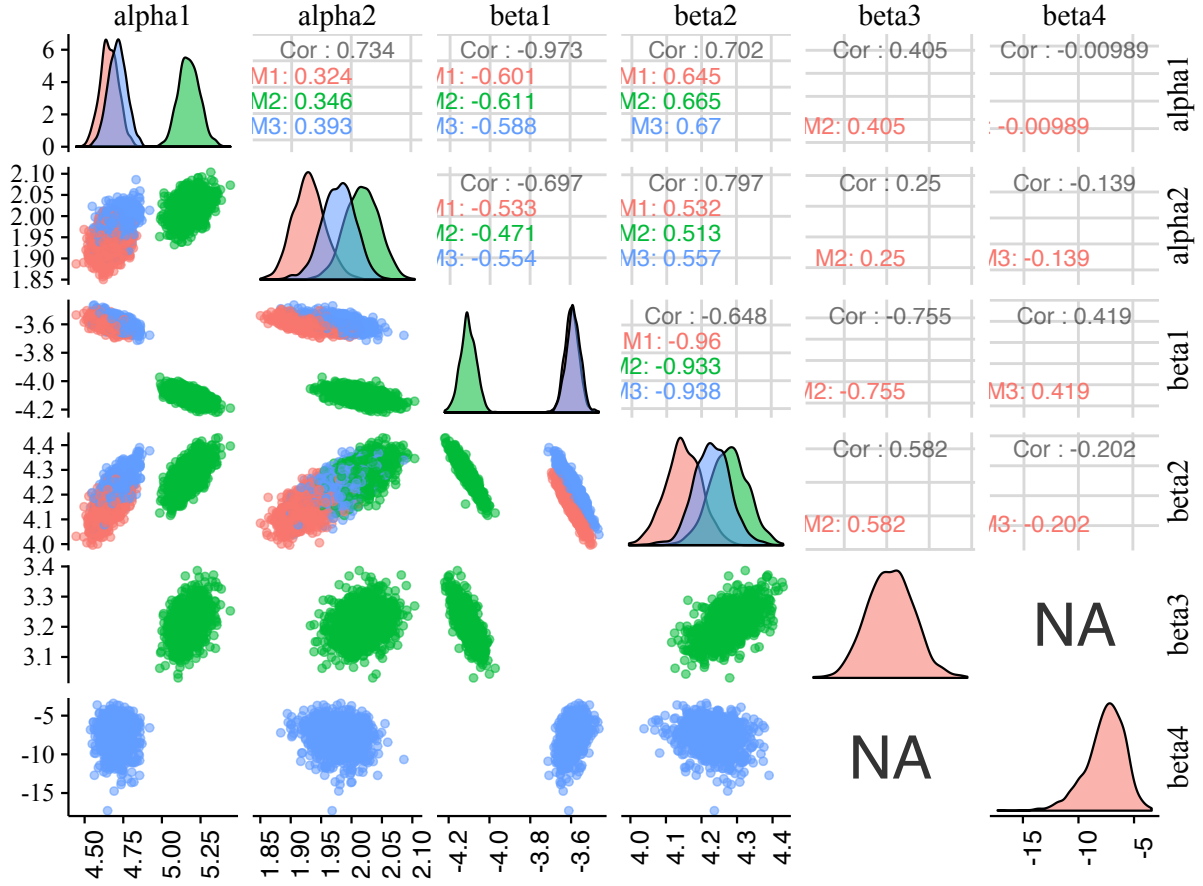
Figure 14: A matrix of plots demonstrating the strong correlations between parameter estimate in our SAOMs. The strongest correlation within each model is between β_1 and β_2.

4.4 Explore algorithms, not just end result

The last principle of model visualization is to explore the process of fitting the model, instead of just focusing on the end result. This principle is perhaps the most important for SAOMs because the model fitting process in `RSiena` involves several simulation steps that are hidden from the user. Hiding the MCMC steps is practical and efficient if a researcher is primarily interested in fitting one model to a set of longitudinal network data, obtaining parameter estimates, and drawing conclusions or making predictions. We are more interested in *how* the models are fit, so we extracted and explored the different steps of that process instead of allowing them stay hidden.

A key component of each step of the SIENA method of moments algorithm is the "microstep" process. A series of microsteps is obtained by simulating from the model in

its current state, $x(t_m)$ with current parameter values $\theta_0 = \{\alpha_{1_0}, \ldots \alpha_{m-1_0}, \beta_{1_0}, \ldots, \beta_{K_0}\}$, to the next state, $x(t_{m+1})$. This microstep process stops when the simulated network has achieved the same number of differences, C, from $x(t_m)$ as $x(t_{m+1})$, where

$$C = \sum_{i \neq j} |x_{ij}(t_{m+1}) - x_{ij}(t_m)|.$$

This simulation process follows the steps of the continuous-time Markov chain. Each tie change in the CTMC is referred to as one "microstep". At each microstep, an "ego node" is selected to make a change, and the chosen ego node randomly makes one change in its ties according to the probabilities, $\{p_{ij} : i \neq j \in \{1, \ldots, n\}\}$ defined in Equation 5, determined by its objective function. The options for change are (1) removing a current tie, (2) adding a new tie, or (3) making no change at all. Saving and exploring all of these steps is not computationally efficient if one is only interested in estimating parameter values, but they can be saved and extracted using options in `RSiena`, which is what we did to create our visualizations. Between two network observations $x(t_m)$ and $x(t_{m+1})$, there can be dozens, hundreds, or even thousands of microsteps, depending on the size of the network and the number of changes between two network observations. We wanted to view these in-between steps in order to better understand the behavior of the underlying continuous-time Markov chain.

The first vizualization we present here is an animation of the simulated microsteps that form the transition steps of the CTMC from wave 1 to wave 2 of the small friendship network example shown in Figure 4 when fitting model M1. Movies similar to this animation were used to visualize the changes of dynamic networks in Moody et al. (2005). When each ego node is selected in a microstep, it is emphasized in the animation, then the associated edge either appears or disappears. If there are no changes at a particular microstep, no changes are seen. Some frames of the animations are shown in Figure 15, and the full movie can be viewed at INSERT LINK TO GIF.

The top row of Figure 15 shows an edge being removed, and the bottom row shows one being added. In both cases, the ego node chosen to act change color from black to red, and they also increase greatly in size. In the case of an edge being removed, in the top row, the edge that currently exists is emphasized with the same color and size change that the node gets, and as the animation proceeds the edge shrinks to nothing, as the ego node shrinks
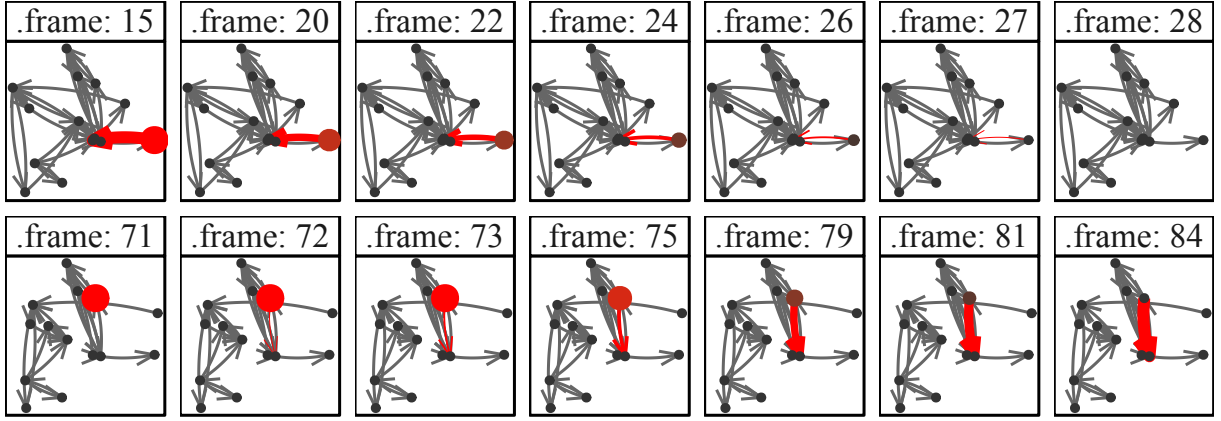
38

Figure 15: A selection of images in the microstep animation. The selected edges and nodes are emphasized by changing the size and the color, then when edges are removed, they fade out, shrinking in size, while the nodes change color and shrink to blend in with the other nodes.

and changes color back to its original black. If an edge is being added, as in the bottom row of the figure, the ego node's appearance changes in the same way as when the edge is being removed, while the edge appears colored red from nothing, and grows to a large size, then changes color and size to match the rest of the edges, while the node shrinks and changes color to match the other nodes.

In the network animation, we see the possible steps of the unobserved CTMC process that is underlying the SAOM fit to the data. We see each part of the model come into play. First, we see the rate at which the nodes are selected to change. Then, we see the result of the actor maximizing its objective function by either deleting or adding a node. In addition, the layout of the nodes changes as edges are removed or added, which gives us a better sense of how the overall network structure changes with these individual tie changes.

We next use animation to view the changing structure of the adjacency matrix the microsteps. The adjacency matrices for the three waves of friendship data as shown in Figure 4 are ordered by node id. There are 16 nodes in the data, numbered 1-16, and that order is used on the x and y axes for the matrix visualization. Viewing the adjacency matrices with this arbitrary ordering does not provide much information to the viewer

about the underlying structure of the network. This lack of perceived structure would be exacerbated in an animation, so we adjust the ordering so that the viewer can better perceive the structure of the network. This process is known as matrix seriation (Liiv, 2010).
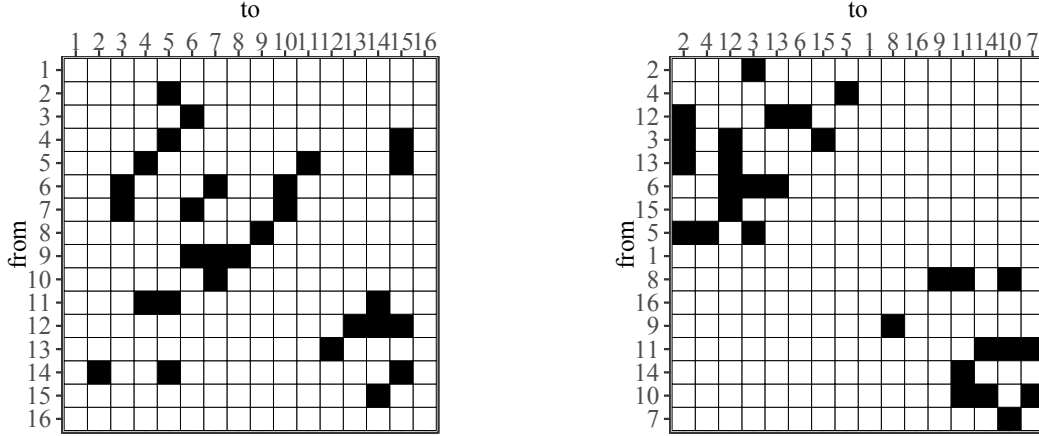

Figure 16: On the left, the starting friendship network represented in adjacency matrix form, ordered by vertex id. On the right, the same adjacency matrix is presented after ordering the vertices by one repetition of the microstep simulation process from wave one to wave two.

To reorder the vertices for the matrix visualization, we first constructed a cumulative adjaceny matrix, $\mathbf{A}^{cum}$, for the series of microsteps simulating the network from $x(t_m)$ to $x(t_{m+1})$. A single entry in the cumulative adjacency matrix, $\mathbf{A}^{cum}_{ij}$, is the total number of times the edge from node i to node j appears in the network from the initial observation, $x(t_m) \equiv X(0)$ to the final result of the last microstep, $X(R)$, where R is the total number of microsteps taken:

$$\mathbf{A}^{cum}_{ij} = \sum_{r=0}^{R} X_{ij}(r).$$

We then performed a principal component analysis (PCA) on $\mathbf{A}^{cum}$, and used the values of the first principal component to order the vertices on the x and y axes for the adjacency matrix animation. For one such series of microsteps simulated by RSiena, we present the adjacency matrix ordered by the (arbitrary) vertex id alongside the seriated adjacency matrix using the first principal component loading on the cumulative adjacency matrix, $\mathbf{A}^{cum}$, in Figure 16.
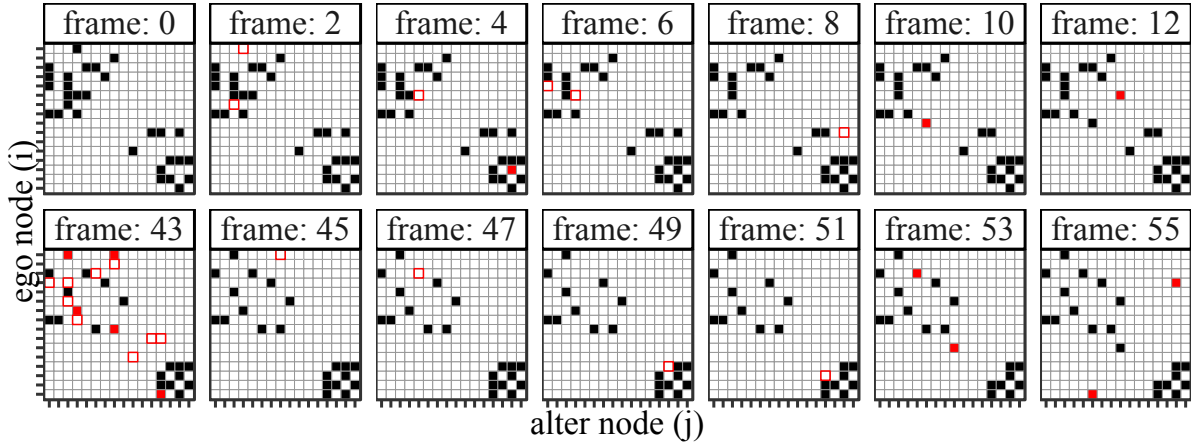
Figure 17: A selection of frames from the adjacency matrix visualization animation for one series of microsteps. (Ego and alter node labels are removed to declutter the graph.) At the beginning of the animation, shown in the top row, there are two clearly connected components: one in the top left corner, and one in the bottom right corner. By the end, the component in the top left has spread out, while the bottom right component has shrunk, but remains closely connected.

Using PCA on $\mathbf{A}^{cum}$ to order the rows and columns of the adjacency matrix visualization clearly shows the two distinct connected components in the first wave of the network, which are difficult to find in the arbitrarily ordered visualization. We also use the PCA seriated layout to fix the layout in the animation of one of the microstep process simulations. This animation, some frames of which are shown in Figure 17 is very simple: a square appears or disappears in the animation as that edge appears or disappears in the microstep process. Through this animation, we can see edges appearing, and then later on disappearing. These in-between steps are not shown when we look at the network at our discrete observation points, so by viewing this animation we can gain a better understanding of the underlying dynamics of this model.

We also attempt to better understand the microstep process by visualizing the observed transition probabilites for the first microstep in the process. We only do the first step of many because the `RSiena` transition probabilities for any two edges i, j after the first step are only directly comparable for identical ego node states due to the conditioning on the current network state in the model. By ego node state, we mean the current set of incoming and outgoing ties to the ego node, i. Put more precisely, let the vector

$X_i = (\{x_{ik}\}, \{x_{ki}\})_{k \neq i} \in \{0, 1\}^{2(n-1)}$ represent the set of all tie variables involving node i in the current network. For p_{ij} to be comparable for any two network states x and x', the value of the vector X_i must be the same in both states. It is possible to incorporate this information into our visualizations, but for now we look at only the first step, because we know without needing any complicated conditioning steps that the previous node state X_i is identical in all cases. Thus we have 1,000 transition probabilities to examine: one transition probability for the first microstep taken for each of the simulations. In Figure 18, we present each ego node and the resulting probabilities of tie changes. The probability shown by the bars is the theoretical probability, according to the objective function, of the ego node changing its tie to the alter node, while the probability shown by the points is the empirical probability of that change being made. The empirical probability is calculated by counting all instances of the ego node first, then computing the proportion of each different alter node change. In most cases, they are almost identical, which demonstrates that the algorithm is performing about as expected. There are, however, many steps that are never taken. Some ego nodes, like 1, 4, and 16, really explore the space of all possible outgoing ties. On the other hand, nodes like 13 and 15 explore very few possible outgoing ties. It is unclear why this would be happening; it may have something to do with the ties in wave 1, where 13 and 15 have multiple connections, while 1 and 16 have none, but there are several other counterexamples of well connected nodes that explore the space more. In addition, we can see that the ties changed most often are removing ties, not adding ties. This tracks with the overall pattern of the data: we see the most ties in the first wave of data, and the least ties in the third wave.

Another way to view these transition probabilities is through the adjacency matrix visualization. In Figure 19, we build on the concept of the ordered adjacency matrix of Figure 16. This heatmap shows the transition probabilities of all ties that are changed in the first microstep of the 1,000 simulations. The heatmap is noticeably sparse: of the $16^2 = 256$ possible steps for the CTMC to take, only 103, or about 40%, are taken in the 1,000 simulated chains. This reinforces what we saw in Figure 18, where there are many paths not taken. This effect could only be exacerbated as more steps are taken in the CTMC, leading to a very large area of our network space, $\mathcal{X}$, completely unexplored by

We also wanted to better understand the entire microstep process from the first wave, all the way through to the second wave. The number of steps taken from wave 1 to wave 2 varies. In one set of 1,000 simulations from Model 1, the smallest number of steps taken was 58, and the longest was 248, with a mean of 106 and a median of 103. In the 1,000 simulations, the standard deviation of the number of microsteps was 22.8. In Figure 20, we see two simulations of the process from wave 1 to wave 2, with wave 1 shown on the left, and wave 2 shown on the right. In each of the three plots, the y-axis contains the edges sorted by how often they appear in the networks along the way. We can see that some edges are there in the beginning, but disappear and never come back, while others appear a few steps in, only to disappear again. There are also some edges that were observed in wave 2 that don't appear at all in the microstep process in a given simulation. So, even though the CTMC makes about the right *number* of changes as it was designed to do, the changes it is making are not necessarily in the right direction.

We also combine 1,000 simulations from model M1 into a visualization like the one shown in Figure 20. We first assign each possible edge an edge ID number so that we can keep track of it throughout all the microsteps and all the simlations. Then, we count up the total number of times each edge appears in the microstep process in each of the 1,000 simulations for use as an ordering variable later. We also count up the number of times an edge occurs in each microstep number in the 1,000 simulations. Since the number of microsteps in the process varies, the number of times an edge occurs decreases as the microstep number increases. Next, we compute a proportion, which we call the occurrence percentage, which is the number of times the edge occurred in a microstep divided by 1,000. Finally, we visualize all this information together in Figure 21. In this plot, all possible edges are shown, and we see that every one of the $16 \times 15 = 240$ possible edges in the network occurs at some point in the simulation process. We also see, however, that the process struggles to focus in on the edges in the second wave of the data. Ideally, we would like to see more occurrences of the edges which appear in the second wave of data. But, about half of the edges in wave two are in the bottom half when ordered by number of occurrences, meaning they do not appear as much as they would if the model was truly

excellent at capturing the mechanisms of tie change in the network. This solidifies what we found in Figures 18 and 19: the model M1 and the SAOM fitting process do not explore the data space enough to adequately capture the network change mechanism.

The visualizations should address most elements of structure described before, i.e. follow along in the setup (for descriptive and teaching purposes), the fitting (understanding the model and interpreting the results) and then diagnostics (how well does the model fit the data - where are the most differences to the actual data/what are the sensitive parameters?)

5 Discussion

We have used novel visualization methods in order to better understand the family of models known as stochastic actor-oriented models for social network data. By looking at the underlying algorithms, visualizing collections of these models, and viewing the model in the data space, we have been able to gain knowledge and appreciation for these complicated models and everything that goes into them.

XX HH help please!

We have only just begun to scratch the surface of these complicated and multi-layered models for social networks. The `RSiena` software is incredibly powerful, and can fit a whole slew of much more flexible stochastic actor-oriented models than we have examined here. If a researcher thinks the network structure or an actor covariate effects the rate of change of the network, there is a way to incorporate that belief into the rate function of the SAOM. More than one actor-level covariate can be included in the model, and way more than three parameters can be included in the objective function itself. In addition, `RSiena` allows the user to tell it which parameters lead to tie creation, and which parameters lead to tie endowment, or dissolution. We have used "evaluation" parameters, which assume that creation and endowment are equal (Ripley et al., 2017). Finally, SAOMs and `RSiena` are able to also model behavior change of the actors in the network, which again is a capability we did not explore here.

References

Buja, A., Cook, D., Hofmann, H., Lawrence, M., Lee, E., Swayne, D., and Wickham, H. (2009), "Statistical Inference for Exploratory Data Analysis and Model Diagnostics," *Royal Society Philosophical Transactions A*, 367, 4361–4383.

Donald E. Knuth, R. W. and John J. Watkins, e. (2013), *Combinatorics: Ancient and Modern*, Oxford University Press, chap. Two thousand years of combinatorics.

Fekete, J.-D. (2009), "Visualizing Networks using Adjacency Matrices: Progresses and Challenges," in *11th IEEE International Conference on Computer-Aided Design and Computer Graphics*, pp. 636– 638.

Fruchterman, T. M. and Reingold, E. M. (1991), "Graph Drawing by Force-Directed Placement," *Software: Practice and Experience*, 21, 1129–1164.

Ghoniem, M., Fekete, J.-D., and Castagliola, P. (2005), "On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis," *Information Visualization*, 4, 114, copyright - Palgrave Macmillan Ltd 2005; Document feature - charts; graphs; tables; references; equations; Last updated - 2012-01-28.

Gibson, H., Faith, J., and Vickers, P. (2013), "A survey of two-dimensional graph layout techniques for information visualisation," *Information Visualization*, 12, 324–357, copyright - SAGE Publications Jul 2013; Document feature - Tables; ; Last updated - 2013-08-05.

Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. (2010), "A Survey of Statistical Network Models," *Foundations and Trends in Machine Learning*, 2, 129–233.

Gross, J. H., Kirkland, J. H., and Shalizi, C. R. (2008), "Cosponsorship in the U.S. Senate: A Multilevel Two-Mode Approach to Detecting Subtle Social Predictors of Legislative Support," *Unpublished Manuscript*.

Hofmann, H., Follett, L., Majumder, M., and Cook, D. (2012), "Graphical tests for power comparison of competing designs," *Visualization and Computer Graphics, IEEE Transactions on*, 18, 2441–2448.

Kamada, T. and Kawai, S. (1989), "An Algorithm for Drawing General Undirected Graphs," *Information Processing Letters*, 31, 7–15.

Liiv, I. (2010), "Seriation and Matrix Reordering Methods: An Historical Overview," *Statistical Analysis and Data Mining*, 3, 70–91.

Michell, L. and Amos, A. (1997), "Girls, pecking order and smoking," *Social Science & Medicine*, 44, 1861 – 1869.

Moody, J., McFarland, D., and Bender-deMoll, S. (2005), "Dynamic Network Visualization," *American Journal of Sociology*, 110, 12061241.

Newman, M. E. J. (2010), *Networks : an introduction*, Oxford New York: Oxford University Press.

Ripley, R., Boitmanis, K., Snijders, T. A., and Schoenenberger, F. (2016a), *RSiena: Siena - Simulation Investigation for Empirical Network Analysis*, R package version 1.1-304/r304.

— (2016b), *RSienaTest: Siena - Simulation Investigation for Empirical Network Analysis*, R package version 1.1-294.

Ripley, R. M., Snijders, T. A., Boda, Z., Vrs, A., and Preciado, P. (2017), *Manual for RSiena*, University of Oxford: Department of Statistics; Nuffield College; University of Groningen: Department of Sociology, `https://www.stats.ox.ac.uk/~snijders/siena/RSiena_Manual.pdf`.

Robbins, H. and Monro, S. (1951), "A stochastic approximation method," *The Annals of Mathematical Statistics*, 22, 400–407.

Schloerke, B., Crowley, J., Cook, D., Hofmann, H., Wickham, H., Briatte, F., Marbach, M., and Thoen, E. (2016), *GGally: Extension to ggplot2.*, R package version 1.3.0.

Snijders, T. A. (2005), *Models and Methods in Social Network Analysis*, New York: Cambridge University Press, chap. Models for Longitudinal Network Data.

— (2016), *Siena Algorithms*, University of Oxford: Department of Statistics, `https://www.stats.ox.ac.uk/~snijders/siena/Siena_algorithms.pdf`.

Snijders, T. A. B. (1996), "Stochastic actor-oriented models for network change," *Journal of Mathematical Sociology*, 21, 149–172.

— (2001), "The Statistical Evaluation of Social Network Dynamics," *Sociological Methodology*, 31, 361–395.

Snijders, T. A. B., Koskinen, J., and Schweinberger, M. (2010), "Maximum Likelihood Estimation for Social Network Dynamics," *The Annals of Applied Statistics*, 4, 567–588.

Tyner, S. and Hofmann, H. (2016), *geomnet: Network Visualization in the 'ggplot2' Framework*, r package version 0.2.0.9001.

Wickham, H., Cook, D., and Hofmann, H. (2015), "Visualizing statistical models: Removing the blindfold," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 8, 203–225.

Yin, G. G. and Zhang, Q. (2010), *Continuous-Time Markov Chains and Applications: A Two-Time-Scale Approach*, New York: Springer, 2nd ed., iSBN 978-1-4614-4345-2.
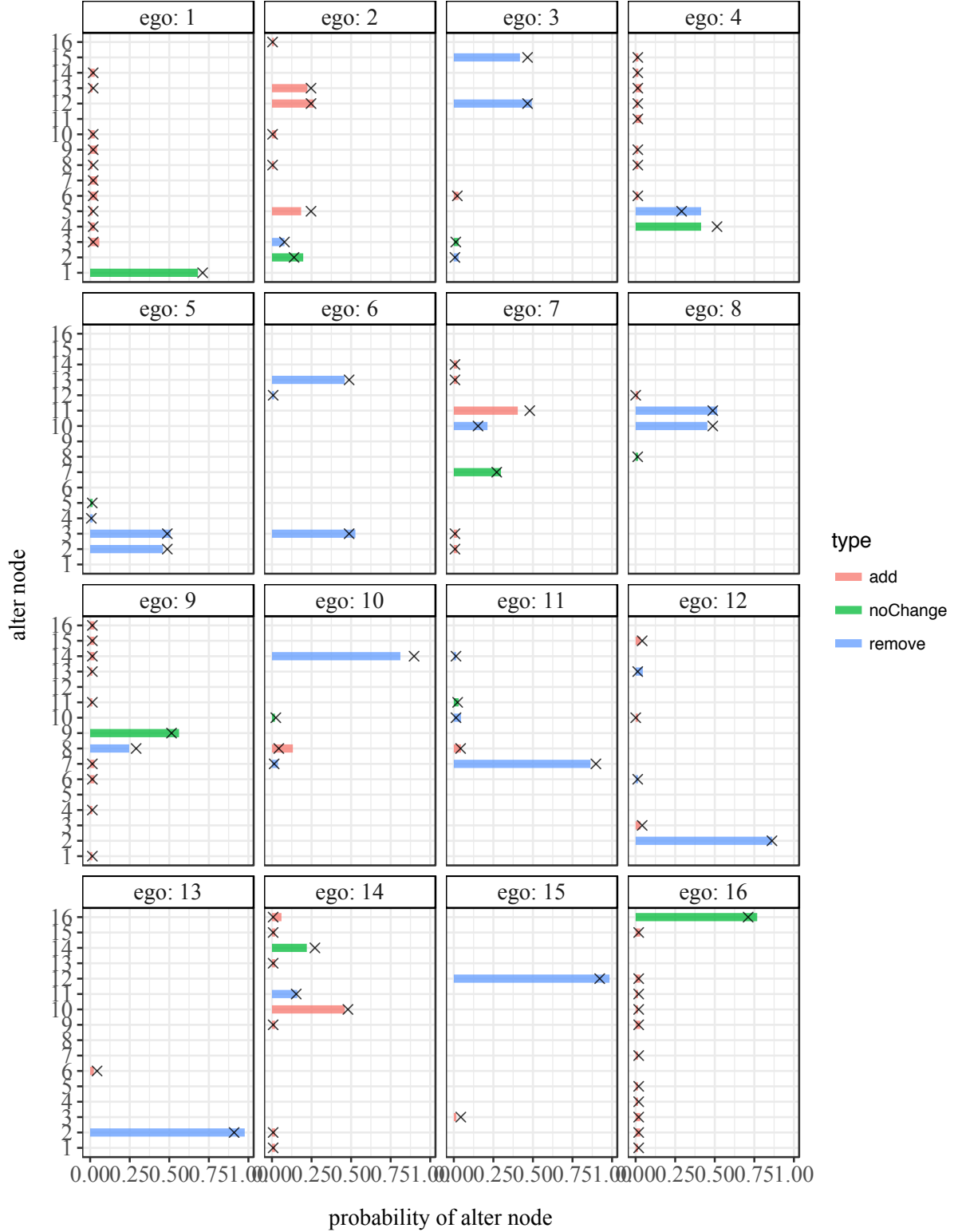
Figure 18: Each panel shows the theoretical (as lines) and empirical (as points) probabilities of the chosen ego node changing its tie to each of the other nodes. The color of the line indicates whether the tie from the ego to the alter node is being added, removed, or if there is no change to the network in this step.
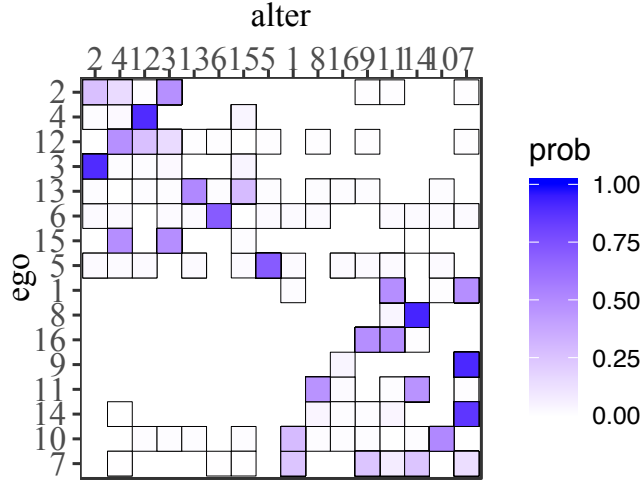
48

Figure 19: A heatmap showing the empirical transition probabilities for the first microstep in 1,000 simulations. The acting ego node is on the y-axis, and the alter node is on the x-axis. There were many ties with empirical probability zero.
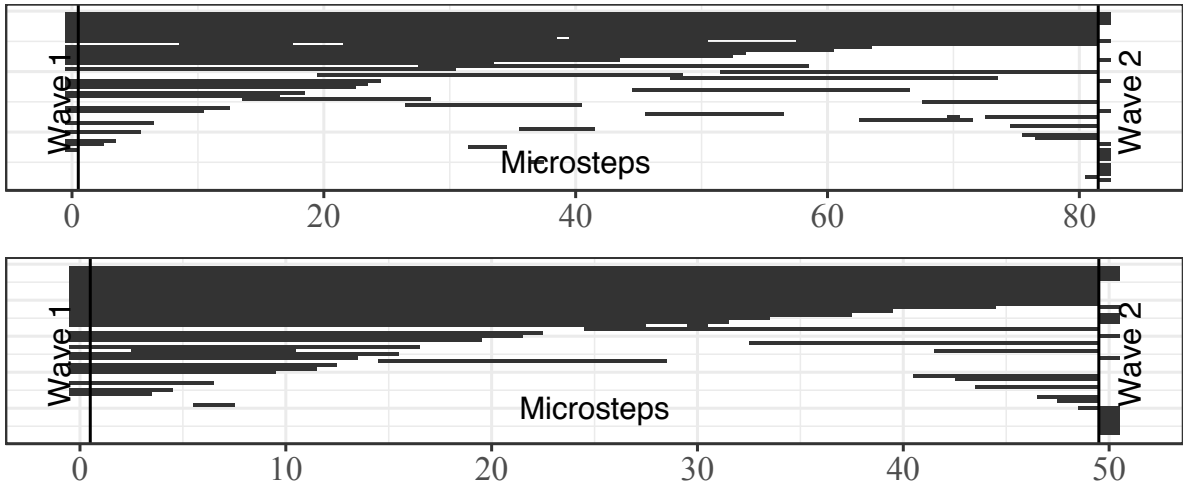

Figure 20: Two simulations (out of 1,000) of the microstep process from wave 1 to wave 2. The x axis is the microstep number, with step 0 representing the first wave of data and the final step representing the second wave of data. We can see that many edges are underrepresented in this process: they are in the second wave, but never appear in the microsteps.
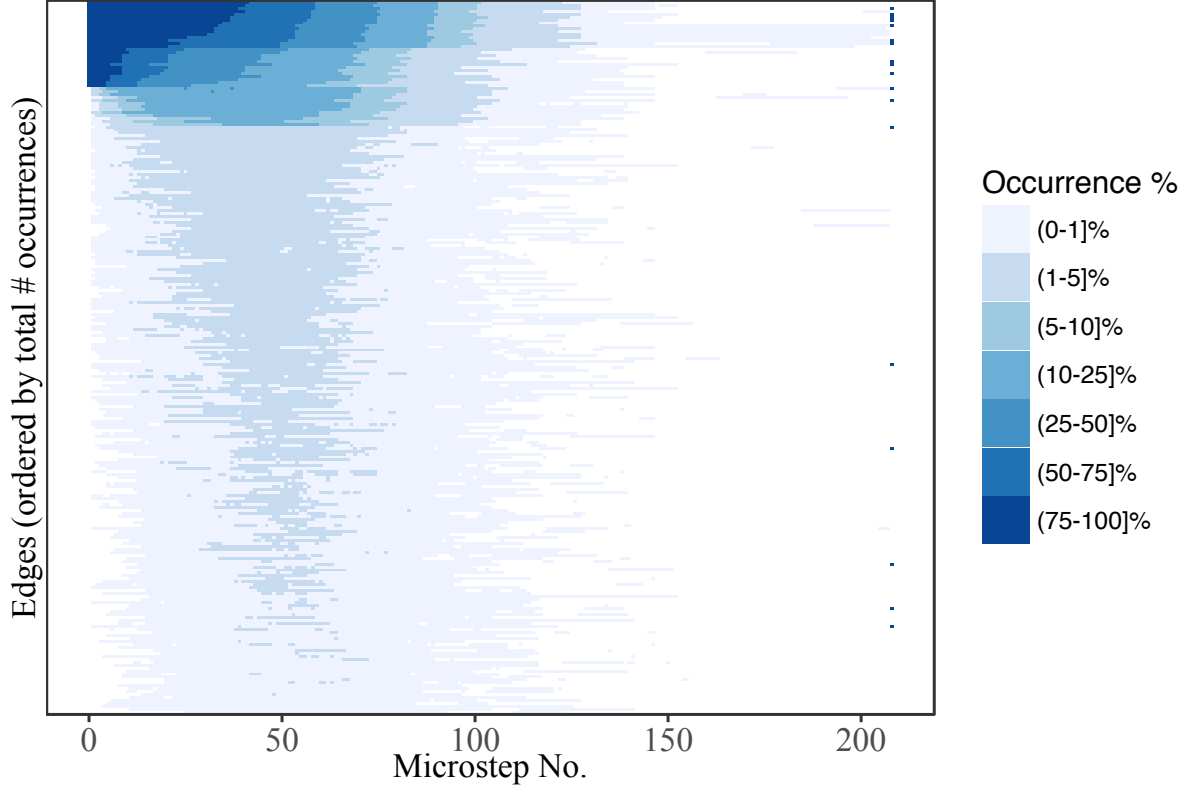
Figure 21: Visualizing all microsteps taken in 1,000 simulations from the model M1. The occurrence percent is split up into groups to correspond with its distribution: only about 10% of the edges appear more than 10% of the time in the 1,000 simulations, while about 60% appear less than 1% of the time. The first wave network is shown at microstep 0, and the second wave of the network is shown as the last microstep for comparison. We see that it is rare for a microstep process to last longer than 150 steps, and also that the edges that appear past the 150th step tend to be in either the first wave or the second wave.