Statistical Mechanics of Networks: Estimation and Uncertainty

Forthcoming: *Physica A*

B.A. Desmarais[a,*], S.J. Cranmer[b]

[a]*Department of Political Science, University of Massachusetts at Amherst, Amherst, Massachusetts 01003, United States*

[b]*Department of Political Science, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina 27599, United States*

Abstract

Exponential random graph models (ERGMs) are powerful tools for formulating theoretical models of network generation or learning the properties of empirical networks. They can be used to construct models that exactly reproduce network properties of interest. However, tuning these models correctly requires computationally intractable maximization of the probability of a network of interest – maximum likelihood estimation (MLE). We discuss methods of approximate MLE and show that, though promising, simulation based methods pose difficulties in application because it is not known how much simulation is required. An alternative to simulation methods, maximum pseudolikelihood estimation (MPLE), is deterministic and has known asymptotic properties, but standard methods of assessing uncertainty with MPLE perform poorly. We introduce a resampling method that greatly outperforms the standard approach to characterizing uncertainty with MPLE. We also introduce ERGMs for dynamic networks – temporal ERGM (TERGM). In an application to modeling cosponsorship networks in the United States Senate, we show how recently proposed methods for dynamic network modeling can be integrated into the TERGM framework, and how our resampling method can be used to characterize uncertainty about network dynamics.

Keywords: Networks, Dynamic Network, ERGM, Bootstrap, Congress

[*]Corresponding author

Email addresses: desmarais@polsci.umass.edu (B.A. Desmarais), skyler@unc.edu (S.J. Cranmer)

1. Introduction

In work on the statistical mechanics of networks, probability models—families of probability distributions—are often derived to represent a mathematical feature of a network that has been observed empirically, is of theoretical importance, or both. Consider transitivity in a static network defined on a fixed set of vertices. Transitive networks are those in which edges are more likely to exist between vertices that share a neighbor than between those that do not share a neighbor. Burda, Jurkiewicz and Krzywicki [1] develop a model that characterizes the transitivity in a static network defined on a fixed set of vertices. Considering dynamic networks, Grindrod and Parsons [2] derive a model to infer the temporal patterns of edge creation and elimination. Models that represent individual generative processes permit focused theoretical analysis and offer parsimonious descriptions of empirical networks. However, they may fall short of providing accurate models for complex real-world networks.

Moving beyond models built to represent or infer individual generative features of networks, exponential family random graph models (ERGMs), which were introcuded to the physics literature by Park and Newman [3], can be specified to represent multiple processes that underly the probabilistic generation of a static directed or undirected network. ERGMs have seen recent application to topics in statistical mechanics [4, 5]. Building upon the theoretical presentation of the basic ERGM framework for static single networks in [3], we present exponential family models for time serial network data. We also present general algorithms for fitting these models to network data and discuss critical limitations in existing algorithms' abilities to capture uncertainty in the estimates. We then introduce a novel method, based on the nonparametric bootstrap [6], for appropriately summarizing uncertainty in estimates. The models and methods we present are illustrated through an application on cosponsorship networks in the United States Senate.

2. Exponential Family Models of Networks

It is instructive to consider the probability of a particular configuration of a network (graph, denoted $\boldsymbol{G}$) defined on a fixed set of v vertices in an ERGM.

$$\mathcal{P}(\boldsymbol{G},\boldsymbol{\theta}) = \frac{\exp\{\boldsymbol{\theta}'\,\boldsymbol{x}(\boldsymbol{G})\}}{\sum_{\boldsymbol{G}^*\in\boldsymbol{\mathcal{G}}}\exp\{\boldsymbol{\theta}'\,\boldsymbol{x}(\boldsymbol{G}^*)\}}, \tag{1}$$

where $\boldsymbol{\mathcal{G}}$ is the set of all networks defined on v vertices, $\boldsymbol{\theta} \in \mathbb{R}^p$ is the vector-valued parameter of the ERGM, $\boldsymbol{\theta}'$ is the transpose of $\boldsymbol{\theta}$ (i.e., $\boldsymbol{\theta}^T$), and $\boldsymbol{x}$ is a p-vector-valued function of the network.[1] Note that $\boldsymbol{\theta}'\,\boldsymbol{x}(\boldsymbol{G})$ – the dot product of the parameter vector and the vector of network statistics – is often referred to as the Hamiltonian of $\boldsymbol{G}$, and the denominator of (1) is the partition function (also called a normalizing constant). In this model, each individual parameter value θ corresponds to an indivudual scalar-valued statistic (x) computed on the network. The parameter moderates how the corresponding feature of the network configuration effects the probability of that particular configuration. For instance, if θ is positive, and x is the number of edges in the network, then the probability of a network configuration increases with the number of edges in the network.

The ERGM provides a simple and useful correspondence between measures of network features (e.g., clustering, homophily, reciprocity in a directed network) and the probability of a network. In constructing a probabilistic model of a network, the parametrs separate out the influences of multiple processes on the generation of the network. In theoretical exercises, $\boldsymbol{\theta}$ can be directly manipulated to control the direction and magnitude of the effect of network features on the frequency with which networks exhbiting those features are generated. For instance, to favor the generation of networks that do not exhibit clustering, a negative parameter value can be assigned to a statistic equal to the clustering coefficient [7] of the network. The ERGM can be, and often is, used to infer the effect of network features in generating empirically observed networks [8]. Suppose $\boldsymbol{G}_0$ is observed, $\boldsymbol{x}$ is posited to be the vector of features that regulate the distribution from which $\boldsymbol{G}_0$ was drawn, and $\boldsymbol{\theta}_0 = \arg\max_{\boldsymbol{\theta}} [\mathcal{P}(\boldsymbol{G}_0, \boldsymbol{\theta})]$, then analysis of $\boldsymbol{\theta}_0$ indicates the effects of each network feature, accounting for the effects of the other features included in $\boldsymbol{x}$. This ability of the ERGM to separate out effects permits simultaneous consideration of generative determinants of a network's structure. For example, the number of triangles – triples of vertices in which every vertex is a neighbor of the other two – is often used as a measure of transitivity in a network [1]. Networks that tend to be very sparse will, based simply on general connectivity, tend to have fewer triangles than those that are very dense. Thus, adding statistics to $\boldsymbol{x}$ that measure both the number

[1] We use alternative notation for the transpose operation due to our use of a time superscript (t) throughout the manuscript.

of edges and the number of triangles permits inference on whether there is an unusually high or low number of trianlges in $\boldsymbol{G}_0$, *given* (i.e., accounting for) the number of edges in $\boldsymbol{G}_0$.

A useful and precise relationship exists between $\boldsymbol{\theta}_0$ and $\boldsymbol{G}_0$ in the ERGM. If the probability of $\boldsymbol{G}$ is $\mathcal{P}(\boldsymbol{G},\boldsymbol{\theta}_0)$, then the expected value (i.e., vector-valued arithmetic mean) of $\boldsymbol{x}(\boldsymbol{G})$ is equal to $\boldsymbol{x}(\boldsymbol{G}_0)$. In other words, parameterizing the ERGM with $\boldsymbol{x}$ and the parameter values estimated/learned by maximizing the probability of a particular configuration of the network $\boldsymbol{G}_0$ results in a distribution of networks in which the average network features are equal to the features of $\boldsymbol{G}_0$ [3]. This illustrates an additional advantage of using ERGMs: if parameter values are set equal to the maximum likelihood estimates computed on a network of interest, then the probabilistic model derived will generate networks that exhibit features that are, on average, equal to the features of the network of interest.

A class of models for network data that shares the form of (1), and thus shares the properties of ERGMs with respect to network features and MLEs, is the temporal exponential random graph model (TERGM) for modeling a time series of networks in which time is discrete [9]. The network at time t has an ERGM distribution in which $\boldsymbol{x}$ includes functions of the network at time t ($\boldsymbol{G}^t$) and the q preceding time points. The elements of $\boldsymbol{x}$ that include both the current and previous networks can measure, for example, stability in the edges, delayed reciprocation, and delayed cluster formation. In the TERGM,

$$\mathcal{P}(\boldsymbol{G}^t,\boldsymbol{\theta}) = \frac{\exp\{\boldsymbol{\theta}'\,\boldsymbol{x}(\boldsymbol{G}^t, \boldsymbol{G}^{t-1}, \ldots, \boldsymbol{G}^{t-q})\}}{\sum_{\boldsymbol{G}^* \in \boldsymbol{\mathcal{G}}^t} \exp\left\{\boldsymbol{\theta}'\boldsymbol{x}\left(\boldsymbol{G}^*, \boldsymbol{G}^{t-1}, \ldots, \boldsymbol{G}^{t-q}\right)\right\}}, \qquad (2)$$

where $\boldsymbol{\mathcal{G}}^t$ changes over t with the set of vertices.[2] Given a series of observed, or theoretically interesting, networks of length T $\{\boldsymbol{G}_0^1, \ldots, \boldsymbol{G}_0^T\}$, let $\boldsymbol{\theta}_0 = \arg\max_{\boldsymbol{\theta}} \left[\prod_{t=q+1}^{T} \mathcal{P}(\boldsymbol{G}_0^t, \boldsymbol{\theta})\right]$.[3] Similar to the static ERGM, if the probability of $\boldsymbol{G}^t$ is $\mathcal{P}(\boldsymbol{G}^t,\boldsymbol{\theta}_0)$, then the expected value of $\boldsymbol{x}(\boldsymbol{G}^t, \boldsymbol{G}^{t-1}, \ldots, \boldsymbol{G}^{t-q})$

[2] Note that the TERGM takes the set of vertices, and how it changes over time, as given.

[3] An alternative to excluding the initial k networks from the calculation is to specify an alternative distribution for them. This is what [9] suggest. Since a different distribution would be estimated for these networks, it is not clear that this approach improves upon dropping them from the analysis in estimating $\boldsymbol{\theta}$.

is $\frac{1}{T-q}\sum_{t=q+1}^{T}\boldsymbol{x}(\boldsymbol{G}_0^t, \boldsymbol{G}_0^{t-1}, \ldots, \boldsymbol{G}_0^{t-q})$ [9]. Thus, the TERGM parameterized with the MLEs derived from a series of network of interest will, on average, exhibit the average features of that series of networks.

3. Estimation

The above models probabilistically reproduce selected features of observed or otherwise interesting data when parameterized with maximum likelihood estimates computed on the data of interest. Unfortunately, direct maximum likelihood estimation is computationally intractable. This is due to the size of $\boldsymbol{\mathcal{G}}$. Note that the computation of the partition function requires the summation over all of the elements of $\boldsymbol{\mathcal{G}}$. If, for example, $\boldsymbol{G}$ is undirected with 10 vertices, then $\boldsymbol{\mathcal{G}}$ contains $2^{\binom{10}{2}} = 35,184,372,088,832$ network configurations. Aside from special cases of exponential family graphical models for which parsimonious formulae for computing the partition function have been discovered [10], it is not computationally practical to directly compute $\mathcal{P}$ for all but small (i.e., < 10) vertex sets. Thus, in order to derive MLEs of a given network configuration of interest, the MLEs must be approximated.

Two approximation methods are prominent in the literature. The first is a simulation method: Markov chain Monte Carlo maximum likelihood estimation (MCMC-MLE) [11, 12], which uses a sample of networks from a Markov chain to approximate the partition function in (1). The algorithm begins with a starting value for the parameter vector. The samples are drawn using the current estimate of the parameters, and new estimates are chosen to maximize the stochastically approximated likelihood function. The approximation and maximization procedures iterate until there is little change in the parameter estimates (i.e., convergence). We present pseudo-code for this algorithm in figure 1. A general MCMC-MLE procedure that can be used with multiple networks (e.g., a time series of networks), represents a minor adjustment to MCMC-MLE procedures designed for single networks [11, 12] in that multiple samples—one for each network in the data—are taken at the approximation step [9].

The second approximation method is maximum pseudolikelihood estimation (MPLE) [13]. Suppose $\boldsymbol{G}$ is composed of elements such that $G_{ij} = 1$ if vertex i and j share an edge and 0 otherwise. If $\boldsymbol{G}$ is directed, then G_{ij} is an indicator of an edge from i to j. Instead of approximating the likelihood directly, MPLE replaces the joint likelihood of the elements of the network with the product over the conditional probability of each element given the

```
$m$ is the MCMC sample size
$v$ is the number of vertices in the network
$\boldsymbol{\theta}^{[0]} = \mathbf{0}$ or $\boldsymbol{\theta}^{[\mathrm{MPLE}]}$
$i = 0$
repeat
  $i = i + 1$
  draw $\tilde{\boldsymbol{G}} \sim \mathcal{P}(\boldsymbol{G}, \boldsymbol{\theta}^{[i-1]})$ by MCMC, $m$ networks with $v$ vertices
  $\widehat{C(\boldsymbol{\theta})} = \ln \left( \sum_{j=1}^{m} \exp \left[ (\boldsymbol{\theta} - \boldsymbol{\theta}^{[i-1]})' \boldsymbol{x}(\tilde{\boldsymbol{G}}_j) \right] \right)$
  $\boldsymbol{\theta}^{[i]} = \arg\max_{\boldsymbol{\theta}} \left[ \sum_{t=1}^{n} \boldsymbol{\theta}' \boldsymbol{x}(\boldsymbol{G}_t) - \widehat{C(\boldsymbol{\theta})} \right]$
until convergence
```

Figure 1: Estimation by MCMC-MLE

rest of the network. The conditional probability of the ij^{th} element of the network being 1 is

$$\pi_{ij}(\boldsymbol{\theta}) = Pr(G_{ij} = 1 | \boldsymbol{G}_{-ij}, \boldsymbol{\theta}) = 1 / \left[1 + \exp\{-\boldsymbol{\theta}' \delta_{ij}(\boldsymbol{x}(\boldsymbol{G}))\}\right], \qquad (3)$$

where $\boldsymbol{G}_{-ij}$ indicates all elements of the network other than the ij element and the expression $\delta_{ij}(\boldsymbol{x}(\boldsymbol{G}))$ is the vector of changes in $\boldsymbol{x}(\boldsymbol{G})$ when G_{ij} is toggled from 0 to 1, holding the rest of $\boldsymbol{G}$ at the observed values [14]. A hill-climbing algorithm is then used to find

$$\arg\max_{\boldsymbol{\theta}} \sum_{t=1}^{T} \sum_{\langle ij \rangle} \ln \left[\left(\pi_{ij}^{(t)}(\boldsymbol{\theta}) \right)^{G_{ij}^{(t)}} \left(1 - \pi_{ij}^{(t)}(\boldsymbol{\theta}) \right)^{1 - G_{ij}^{(t)}} \right], \qquad (4)$$

where T is the number of networks in the sample, and $\langle ij \rangle$ denotes all pairs of vertices (i.e., all unordered pairs in an udirected network and ordered pairs in a directed network). Thus, the computation of the MPLE does not involve simulation. The MPLE has been shown to converge in probability to the MLE (i.e., the MPLE is a consistent estimator), meaning that it approaches the MLE in distribution as the size of the network being analyzed increases. [14, 15].

There are two advantages to MCMC-MLE. First, the estimates converge to the MLE as the number of simulated networks used in the approximation of the likelihood reaches infinity. This means that the MCMC-MLE can be rendered arbitrarily similar to the MLE by increasing m. Second, in a simulation study, MCMC-MLE implemented with a sufficiently large m, was shown to be more efficient (i.e., lower variance) than MPLE [16]. Indeed, MCMC-MLE has been favored in recent work with ERGMs, though not without exception (see, e.g., [17] and [18]).

The MCMC-MLE procedure, however, has limitations when compared to MPLE. First, it is never known, in practice, whether m, the number of simulated networks used in the MCMC approximation, is sufficiently large to render MCMC-MLE more efficient than MPLE. Second, the realization of efficiency gains from MCMC-MLE over MPLE requires *multiplicatively* more simulation effort as the size of the network(s) increases. This is because, due to its consistency, the performance of the MPLE improves with the size of the network. Thus, in order to maintain a particular level of relative efficiency of the MCMC-MLE to the MPLE, additional simulations of a larger network are required (i.e., m in figure 1 must be increased with larger v). The problem is that, as the volume of data being analyzed increases, the computational burden of realizing the benefits offered by MCMC-MLE can be prohibitive. We illustrate this point in a simulation study below.

4. Uncertainty

When data arise probabilistically, and the model that generated the data is unknown, there is always some degree of uncertainty when inferring the parameters of the model based on the data. Suppose, for instance, we are interested in inferring a parameter in an ERGM that corresponds to a statistic measuring the reciprocity in a directed network (e.g., the number of symmetric dyads ij in which there is an edge from i to j and one from j to i). If the observed network exhibits a high reciprocity statistic, it is unlikely that it was drawn from an ERGM with a non-positive reciprocity parameter. However, the likelihood of a non-positive reciprocity parameter is not zero, as all networks can be observed under any parameterization. So it is useful to be able to accurately assess this likelihood. When testing theory about the parameter values that generated an observed network, it is important to be able to make precise and accurate statements summarizing uncertainty about the parameter values. The sampling distribution of an estimator is the distribution of the estimator over multiple samples of data from the same population. In order to make precise and accurate statements about uncertainty, we require a method for computing the sampling distribution of the estimator used to arrive at the parameter estimates [19].

The sampling distribution of the MLE of exponential family random graph models is multivariate normal with mean vector equal to the MLE and variance equal to the inverse of the negative Hessian matrix ($[-\boldsymbol{\mathcal{H}}]^{-1}$) of the log-likelihood function at the MLE [16]. Confidence intervals, intervals intended to cover some percentage of the sampling distribution, are typically used to test theory about the parameters of a model. For instance, if a 95% confidence interval for some parameter estimate contains only positive values, this is strong evidence against the hypothesis that the true parameter is not positive As mentioned above, the MPLE provides a consistent approximation of the MLE. However, $[-\boldsymbol{\mathcal{H}}]^{-1}$ of the log-pseudolikelihood function computed at the MPLE underestimates the variance of the MPLE [16]. This means that summarizing uncertainty using $[-\boldsymbol{\mathcal{H}}]^{-1}$ results in an underestimate of the width of confidence intervals. When confidence intervals are constructed accurately, an α% confidence interval will contain the true parameter value α% of the time. Constructing confidence intervals for the MPLE using $[-\boldsymbol{\mathcal{H}}]^{-1}$ can result in 95% confidence intervals that only contain the true parameter value less than 75% of the time [16]. Because their use derives from the facts that (a) the MPLE is computed using logistic regression software, and (b) the use of $[-\boldsymbol{\mathcal{H}}]^{-1}$ is the default for estimating the sampling variance in most logistic regression software, we refer to these confidence intervals as "logistic regression confidence intervals."

We render MPLE more useful for probabilistic modeling of networks by deriving a bootstrap method for constructing consistent confidence intervals. Our algorithm is designed for data structures in which multiple networks are under study.[4] The networks are assumed to be independent or conditionally independent of each other. This includes the k-order Markov conditional independence assumption in the TERGM (see [9]).

Our bootstrapped MPLE algorithm works as follows. Let $\hat{\boldsymbol{\Theta}}_s$ be a sample of s estimates of $\boldsymbol{\theta}$ constructed by computing $\hat{\boldsymbol{\theta}}$ on s samples of n networks drawn with replacement from $\{\boldsymbol{G}_1, \boldsymbol{G}_2, \ldots, \boldsymbol{G}_n\}$. It is important to note

[4] This method is similar to bootstrap methods designed for hierarchical models (see e.g., [20] and [21]).

that networks are resampled without disturbing the configurations within the networks. We establish that this algorithm consistently estimates the confidence intervals of the MPLE by showing that the MPLE is a multivariate M-estimator,[5] since it has been demonstrated that bootstrap resampling of multivariate observations results in consistent estimates of the confidence intervals of any consistent multivariate M-estimator [19]. Let h be any scalar-valued function of the data $\boldsymbol{G}$ and the vector-valued parameter $\boldsymbol{\theta}$. Any estimator $\hat{\boldsymbol{\theta}}$ that solves the equation

$$\sum_{i=1}^{n} \frac{\partial h(\boldsymbol{G}_i, \hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}} = \mathbf{0},$$

is an M-estimator [19]. We note that the MPLE is such an M-estimator with

$$h(\boldsymbol{G}, \boldsymbol{\theta}) = \sum_{\langle ij \rangle} \ln \left[(\pi_{ij}(\boldsymbol{\theta}))^{G_{ij}} \left(1 - \pi_{ij}(\boldsymbol{\theta})\right)^{1-G_{ij}} \right].$$

Thus, the bootstrap sample of MPLEs provides a consistent estimate of confidence intervals for the MPLE, as long as resampling is conducted with respect to the networks and not the individual elements of the networks. The sample of bootstrap estimates provides an approximation to the sampling distribution of the pseudolikelihood estimator that does not rely on the inappropriate assumption that the maximum pseudolikelihood estimator is equivalent to the MLE.

5. Monte Carlo Validation

5.1. Finite Sample Performance of Confidence Intervals

The theoretical result motivating our bootstrap method is that the MPLE is a consistent estimator of $\boldsymbol{\theta}$, a result that renders the bootstrap a consistent estimator of $\boldsymbol{\theta}$'s sampling distribution. This is an asymptotic result that must be vetted in a finite-sample setting. To this end, we conduct a simulation study to asses the applicability of the consistency result for the bootstrap method in a finite sample. Our simulation study evaluates the bootstrap method on a TERGM.

[5] M-estimators [22] are a class of robust estimators computed by minimizing the sums of functions of the data. Both maximum likelihood estimation and least squares estimation are special cases of M-estimation.

The TERGM is applied to an artificial time series of 25 undirected networks, each with 25 vertices.[6] The $\boldsymbol{x}$ used to parameterize the TERGM are drawn directly from the literature. We include a count of the number of edges in the network ($Edge$) [10]

$$\sum_{\langle ij \rangle} G_{ij}^{(t)},$$

a count of the number or "two-stars" in the network ($2Star$) [10]

$$\sum_{\langle i \rangle} \sum_{\langle jk \neq i \rangle} G_{ji}^{(t)} G_{ki}^{(t)},$$

where the notation $\langle jk \neq i \rangle$ indicates all pairs of vertices that do not include the vertex i. We also include the number of triangles (Tri) [1]

$$\sum_{\langle ijk \rangle} G_{ij}^{(t)} G_{ik}^{(t)} G_{jk}^{(t)},$$

and a count of the number of stable elements of the network from $t-1$ to t ($AR1$) [9]

$$\sum_{\langle ij \rangle} G_{ij}^{(t)} G_{ij}^{(t-1)} + \left(1 - G_{ij}^{(t)}\right)\left(1 - G_{ij}^{(t-1)}\right).$$

The last network statistic included in the Hamiltonian is referred to as a "dyadic covariate" (X) in the literature on social network analysis with the ERGM [23]. A dyadic covariate is a feature that is valued for each pair of vertices. For instance, in a legislative network [24], an example of a dyadic covariate is an indicator of whether two legislators are of the same political

[6] We select the size of the networks simulated in our experiments such that the TERGM data are slightly smaller than those that appear in our application below. Since, due to the convergence of the MPLE in the size of the network, the properties of the method will improve as the network size increases. We can, thus, be confident that the performance of the MPLE in our empirical application will be at least as good as the performance in our simulation study. To simulate the time series, we first initiate the network with a random Bernoulli graph. To eliminate the effect of initial conditions, we draw a series of 1,000 networks where each network is conditional upon the previous network. The Metropolis Hastings algorithm given by Snijders [12] is used to draw each network conditioned on the previous network. The last 25 networks in the series of 1,000 gives one series in the Monte Carlo study. This simulation process is repeated 500 times in the Monte Carlo study.

party. A dyadic covariate $X^{(t)}$ effects the average edge values in the network through the statistic

$$\sum_{\langle ij \rangle} G_{ij}^{(t)} X_{ij}^{(t)}.$$

When the parameter corresponding to this statistic is positive, then the mean value of G_{ij} increases with X_{ij} [23]. The parameter values are set at -0.25, -0.2, 0.5, 1, and 0 respectively. [7] In each case 1,000 resamples are used in the bootstrapping.[8] For each model, the simulation study consists of 500 iterations.[9]

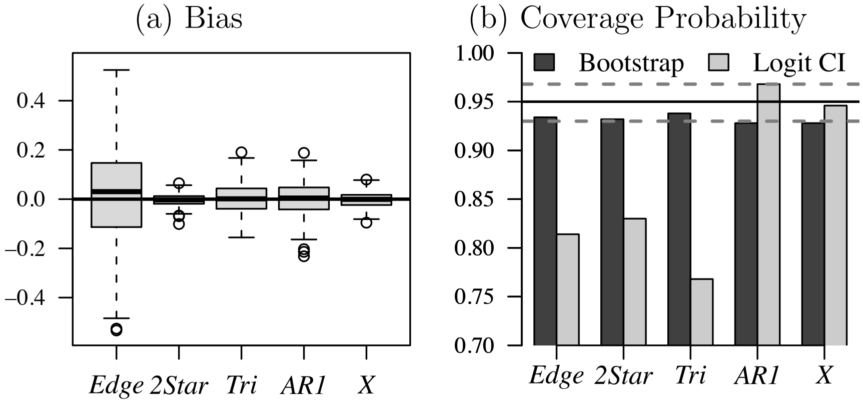


Figure 2: Monte Carlo Results. Box plots of the iteration-wise bias ($\hat{\theta} - \theta$) over the 500 iterations in the Monte Carlo study are given in the first plot. The second plot gives the empirical coverage probability of 95% confidence intervals. The dashed-grey lines in the coverage probability plots are placed at the 0.05-level critical values of a two-sided binomial test of the null hypothesis that the true coverage probability is 0.95.

The results, presented in figure 2, do not show evidence that the MPLE is biased (i.e., different from the true value, on average). Also, the coverage probability (i.e., the proportion of the simulations in which the interval covers

[7] We select parameter values on a comparable scale to those estimated in the cosponsorship application below.

[8] The results of our experiments are unaffected if we use 5,000 instead.

[9] The number 500 permits less than a 3% margin of error in assessing the coverage probability close to 0.95. All computations, in both the simulation study and the applications, are performed in the R statistical environment. Code and data necessary to replicate our results are available upon request.

the true value) of the 95% bootstrap confidence intervals, given by the 2.5^{th} and 97.5^{th} percentiles of the bootstrap sample of MPLEs, is very close to 0.95. In most cases, the coverage probability of the bootstrap technique is an order of magnitude closer to 0.95 than that of the logistic regression confidence intervals. These results provide evidence that the bootstrap technique offers a way to take advantage of the consistency of the MPLE, while maintaining the validity of confidence intervals.

5.2. Relative Efficiency of MCMC-MLE and MPLE

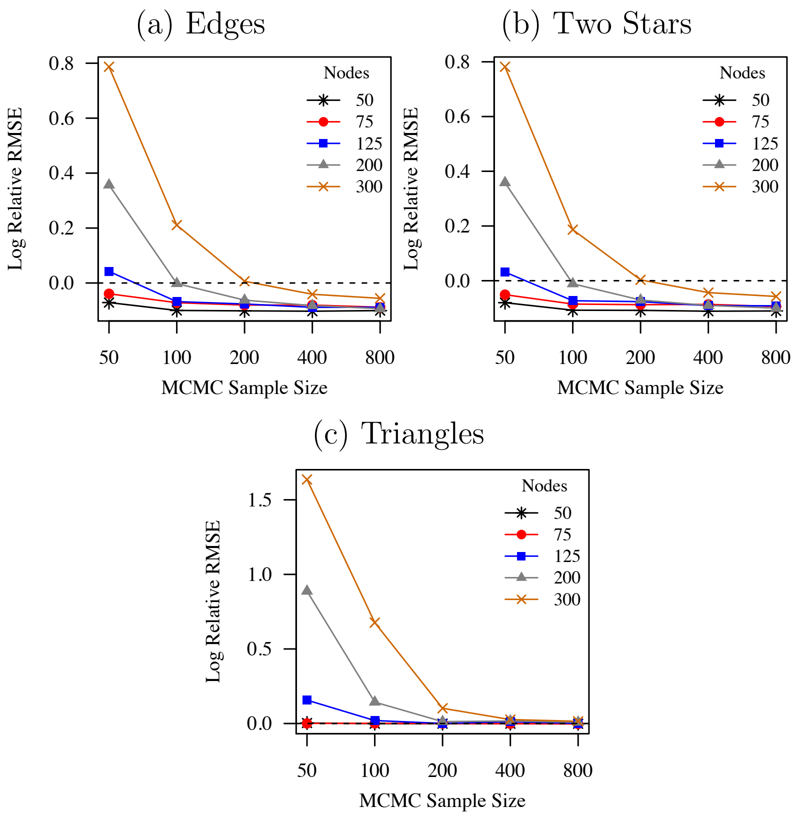
The advantage in estimating parameters by MCMC-MLE lies in the greater efficiency of MCMC-MLE when the number of simulated networks used to approximate the likelihood (m) is set sufficiently high. Here, we illustrate two properties of the MCMC-MLE compared to the MPLE with a simulation study. First, it is possible to set the number of draws low enough for the MPLE to be more efficient than the MCMC-MLE. Second, and perhaps more troubling for the practical implementation of MCMC-MLE, the larger the network, the larger m must be in order for MCMC-MLE to outperform MPLE.

We use a simple example to illustrate these features of MCMC-MLE. The model we use to study the two algorithms is the exponential random graph model (ERGM) parameterized with edges, two-stars, and triangles. We held the parameter values at 0.25, -0.5, and 0.25 respectively and ran the simulation study with MCMC sample sizes of 50, 100, 200, 400, and 800, and with undirected networks of 50, 75, 125, 200, and 300 vertices. We ran each of the 25 conditions, five MCMC sample sizes times five network sizes, for 500 iterations each, using the `ergm` package in R to compute both the MPLE and the MCMC-MLE [25].

The results from the simulation study are depicted in figure 3. We evaluate the two estimators based on root mean squared error (RMSE). The RMSE combines both bias and variance into an overall measure of the average accuracy of an estimator. The RMSE of the estimator $\hat{\theta}$ is

$$\text{RMSE} = \sqrt{\sum_{i=1}^{500}\left(\theta - \hat{\theta}\right)^2}.$$

The RMSE of MCMC-MLE decreases as the number of draws in the approximation of the partition function increases. It is possible to set the number


(a) Edges

(b) Two Stars

(c) Triangles

Figure 3: Simulation results. The y-axis gives the log of the ratio of the root mean squared error (RMSE) of the MCMC-MLE to that of the MPLE. Values below zero indicate MCMC-MLE has a lower RMSE. The number of iterations in the MCMC component of MCMC-MLE are depicted on the x-axis.

of draws in the MCMC high enough such that MCMC-MLE out-performs MPLE. However, the number of draws necessary for MCMC-MLE to out perform MPLE is larger in larger networks. For instance, in this parameterization, MCMC-MLE with $m = 50$ has a lower RMSE than MPLE with a network of 75 vertices. With a network of 300 vertices, m must be set four-fold higher, to 200, in order for MCMC-MLE to have a lower RMSE than MPLE. We note that it is not overly burdensome to run MCMC-MLE with 200 iterations on a network of 300 vertices, but for large network datasets that are observed longitudinally (e.g., Facebook), it may be prohibitively difficult to scale up MCMC-MLE such that gains over MPLE are realized.

6. Cosponsorship in the U.S. Senate

Networks among legislators in the U.S. Congress have been studied in physics recently (see, e.g., [24, 26, 27]). Here, we study the cosponsorship network in the U.S. Senate [28] using a TERGM, as was done in the original article developing the TERGM [9]. Every piece of legislation in the U.S. Congress has a single sponsor, and every other legislator may sign on as a cosponsor. An edge from vertex (i.e., senator) i to j in the cosponsorship network is created when senator i cosponsors legislation sponsored by j [9, 28]. We contribute beyond the results of [9] by analyzing the forecast performance of the TERGM and showing how the method for analyzing dynamic networks proposed by [2] can be approximated using a TERGM.

The time period in our analysis is the two year congress. We train the models on the 99^{th} – 107^{th} Senates, and use the estimates to forecast the 108^{th}.[10] The Hamiltonian in the TERGM is parameterized with statistics used by Hanneke, Fu, and Xing in their analysis of Senate cosponsorship network [9], which model dependence between consecutive time periods. These statistics are summarized in table 1.

Histograms of the parameter estimates from 1,000 bootstrap resamples of the MPLE are given in figure 4. Evaluation of the bootstrap confidence intervals indicates that we have selected appropriate features for inclusion in the TERGM. For every feature but *Edges*, the 95% bootstrap confidence interval does not contain zero. This gives us a high degree of confidence that most of the statistics we have included have non-zero effects on the distribution of the Senate cosponsorship network. The effect of *Same Party* is the largest in magnitude at approximately 0.8. This can be interpreted by considering the relative likelihood of observing two hypothetical instances of the cosponsorship network. Consider two network configurations $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$ that are equal on every feature besides *Same Party.* Specifically, assume $\boldsymbol{G}_1$ has one more within-party edge than $\boldsymbol{G}_2$. According to our estimates, we are $\exp(0.8) \approx 2.226$ times as likely to observe $\boldsymbol{G}_1$ as we are $\boldsymbol{G}_2$. This effect is consistent with the findings of Zhang et al. [24], who also identify a higher propensity for edges to form within parties than across parties. They show

[10] Data are available at `http://jhfowler.ucsd.edu/cosponsorship.htm`. Technically, we use data beginning with the 93^{rd} Senate. But, since our approximation of the method in [2] requires the use of histories from 6 previous networks, we condition on 93 – 98, and do not use these as outcome networks in the TERGMs.

Table 1: Features in the TERGM Hamiltonian

Name	Formula
Edges	$\sum_{\langle ij \rangle} G_{ij}^{(t)}$
Reciprocity	$\sum_{\langle ij \rangle} G_{ij}^{(t)} G_{ji}^{(t-1)}$
*Same Party**	$\sum_{\langle ij \rangle} G_{ij}^{(t)} X_{ij}^{(t)}$
Stability	$\sum_{\langle ij \rangle} G_{ij}^{(t)} G_{ij}^{(t-1)} + (1 - G_{ij}^{(t)})(1 - G_{ij}^{(t-1)})$
Popularity	$\sum_{\langle i \rangle} \sum_{\langle jk \neq i \rangle} G_{ji}^{(t)} G_{ki}^{(t-1)}$
Generosity	$\sum_{\langle i \rangle} \sum_{\langle jk \neq i \rangle} G_{ij}^{(t)} G_{ik}^{(t-1)}$
Transitivity	$\sum_{\langle i \rangle} \sum_{\langle jk \neq i \rangle} G_{ij}^{(t)} G_{ik}^{(t-1)} G_{kj}^{(t-1)}$
Reverse Transitivity	$\sum_{\langle i \rangle} \sum_{\langle jk \neq i \rangle} G_{ij}^{(t)} G_{jk}^{(t-1)} G_{ki}^{(t-1)}$
Co-Supporting	$\sum_{\langle i \rangle} \sum_{\langle jk \neq i \rangle} G_{ij}^{(t)} G_{ik}^{(t-1)} G_{jk}^{(t-1)}$

$^*X_{ij} = 1$ if i and j are of the same political party and 0 otherwise.

that, from the 96^{th} – 108^{th} Congresses, communities in the cosponsorship network defined along party lines exhibit modularity near to the maximum identified with unconstrained community identification.

Since Hanneke, Fu, and Xing [9] also study evolution of the Senate cosponsorship network, it is instructive to compare our results to theirs. In particular, we study Congress-to-Congress evolution of the network, whereas Hanneke, Fu, and Xing. analyze evolution within a single Congress. There are a couple of differences between our results and theirs. Hanneke, Fu, and Xing find a negative effect of *Transitivity* and a positive effect of *Reverse Transitivity*. We find the opposite. Also, they find a positive effect of *Cosupporting* where we find a negative effect. All three of these features represent forms of directed triadic closure (i.e., edges exist between each pair of vertices in a triad). Since triad closure is most commonly associated with clustering in networks [1, 7], the fact that we find fewer positive effects from triadic closure statistics might be indicative of weaker clustering across Congresses than within Congresses. This is logical given that 'communities' are likely driven by coalition formation around specific pieces of legislation or legislative agendas [29], which will be more cohesive within a Congress than across Congresses.

Another way to determine whether the TERGM is an appropriate model is to compare its forecast performance to an alternative dynamic model. Grindrod and Parsons [2] present a model of edge creation and elimination where the status of an edge at the current time point depends upon the most recent span of its history. Specifically, the probability that an edge no longer exists at time t depends upon how long the edge has existed up to t, and the probability that an edge is created at time t depends upon how long it was absent prior to t. Though the model will not have the same exact functional form as the one in [2], these edge creation and elimination functions can be flexibly modeled within the TERGM framework. We do this by conditioning the current edges in the network on whether the edge has existed (or not existed) for 0 periods (i.e., i and j were not both in the previous Senate), 1 period or more, 2 periods or more, through 6 periods or more. We stopped at six periods because, empirically, the creation and loss functions flattened-out after five periods – a point we revisit momentarily. The Hamiltonian of the edge-history TERGM is

$$\sum_{\langle ij \rangle} G_{ij}^{(t)} \left[\lambda + \sum_{k=1}^{6} \alpha_k \prod_{r=1}^{k} G_{ij}^{(t-r)} + \sum_{k=1}^{6} \gamma_k \prod_{r=1}^{k} \left(1 - G_{ij}^{(t-r)} \right) \right],$$

where λ determines the probability of an edge in the first period that i and j are both in the network, α_k changes this probability to reflect the probability of an edge given that an edge has existed between i and j for at least the k previous periods, and γ_k changes this probability to reflect the probability of an edge given that an edge has not existed between i and j for at least the k previous periods. Because the probability of an edge at time t depends only upon its history (i.e., independent of the other edges [2]), the probability of an edge from i to j at time t can be computed without reference to the other edges in the network, and is

$$\left(1 + \exp \left(- \left[\lambda + \sum_{k=1}^{6} \alpha_k \prod_{r=1}^{k} G_{ij}^{(t-r)} + \sum_{k=1}^{6} \gamma_k \prod_{r=1}^{k} \left(1 - G_{ij}^{(t-r)} \right) \right] \right) \right)^{-1}.$$

The individual parameter estimates are less interesting than the relevant probability functions – the probability of edge loss given previous existence,

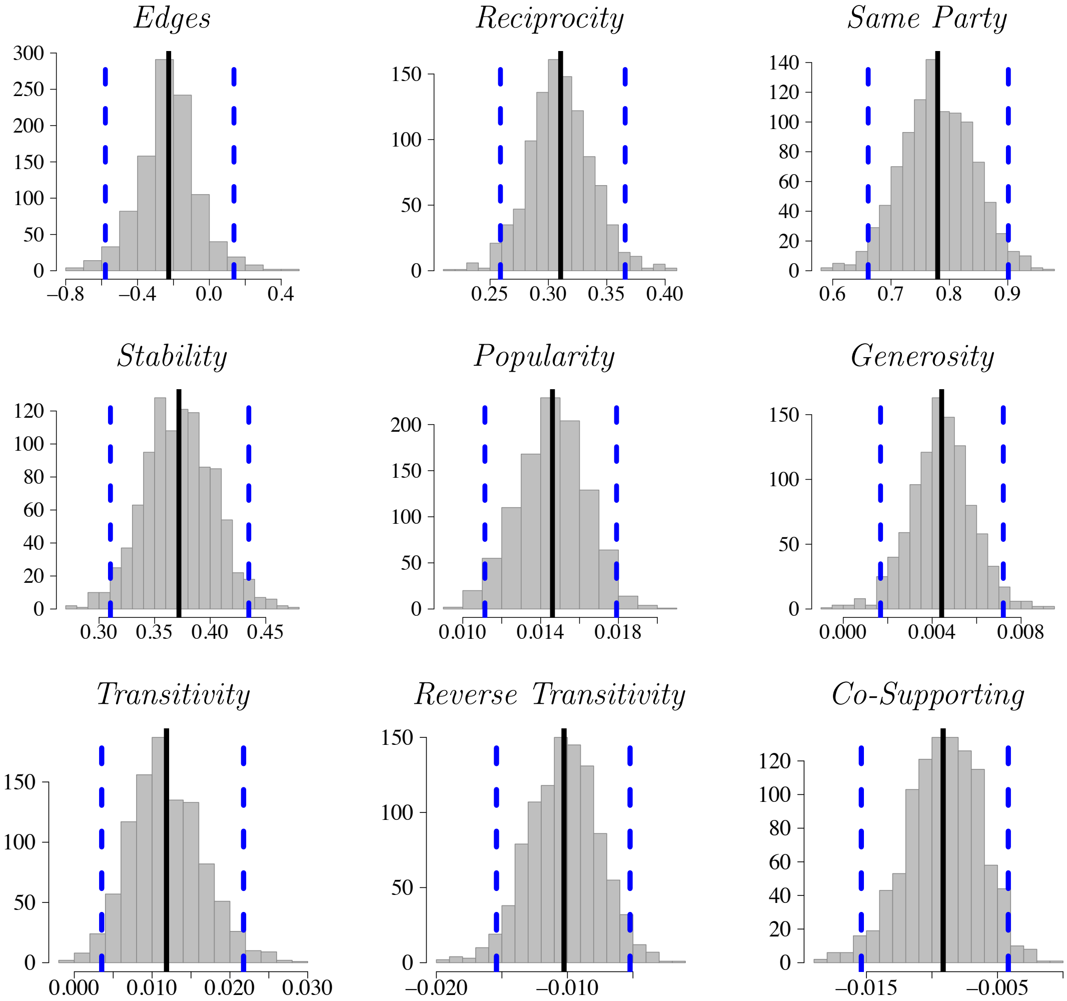


Figure 4: Bootstrap samples of TERGM estimates. Vertical black lines are placed at the estimate, and dashed blue lines bound the 95% bootstrap confidence intervals.

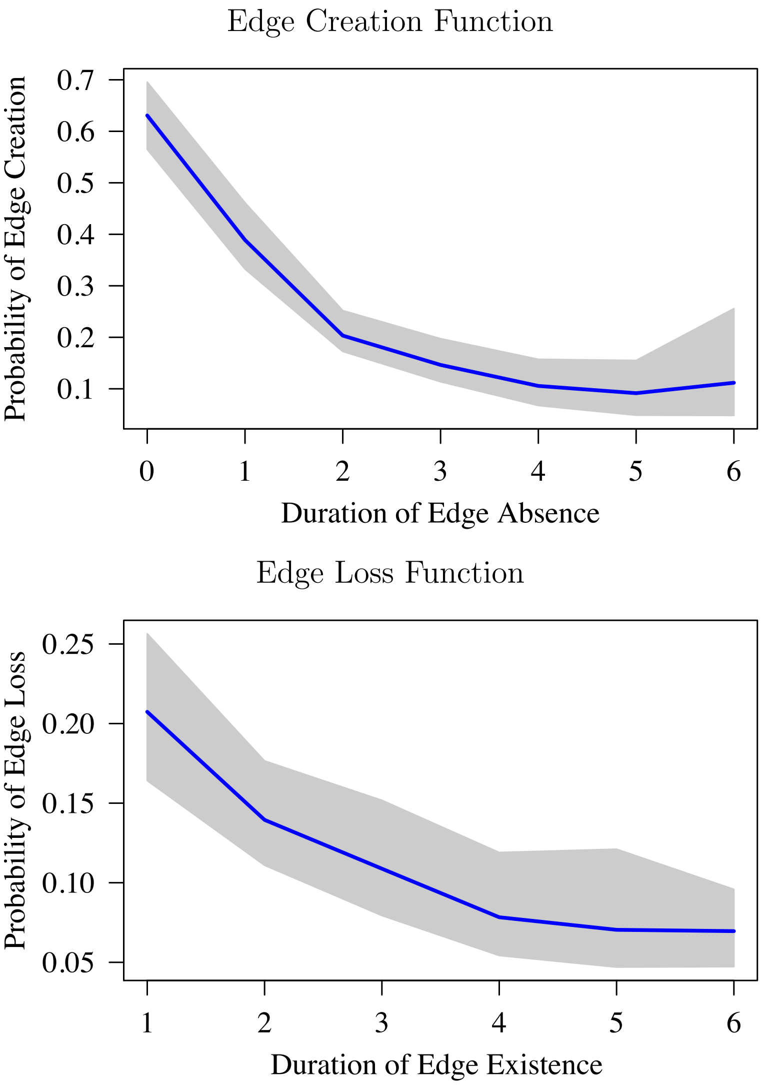
and the probability of edge creation given previous absence. Using the bootstrap method, we can (a) estimate these functions, and (b) construct confidence intervals for them by computing the relevant probabilities using each set of parameters derived in the bootstrap resampling. Figure 5 gives the estimated edge creation and edge loss functions along with 95% confidence regions around the probabilities of edge creation and loss derived from 1,000 bootstrap iterations. We see that the probability of an edge being terminated decreases at a fairly constant rate of approximately 0.05 for every Senate that the edge has existed up to four Senates, and then levels out at 0.075 thereafter. In contrast, the probability of edge creation decreases dramatically from 0.65 in the first Senate that two senators are both in the Senate, to 0.2 after two Senates in which both have been members but have not shared an edge. As noted above, both the edge creation and edge loss functions level out after about five periods.

Our final exercise with this application is to compare the forecast accuracy of the TERGM with single period dependence features (TERGM) to that of the edge history TERGM (EdgeHist). Recall that the models reported above were trained on the $99^{th} - 107^{th}$ Senates. We use the models estimated above to simulate 1,000 draws of the cosponsorship network in the 108^{th} Senate. The forecast accuracy of the models is assessed with respect to their ability to predict (a) the value of each edge, (b) the total degree (i.e., total number of in-coming and out-going edges) of each Senator, and (c) the graph transitivity of the entire network. The graph transitivity is

$$\frac{\sum_{\langle i\rangle}\sum_{\langle jk\neq i\rangle} G_{ij}G_{ik}G_{kj}}{\sum_{\langle i\rangle}\sum_{\langle jk\neq i\rangle} G_{ik}G_{kj}},$$

which is the number of transitive triples in the network, divided by the potential number of transitive triples [30]. Considering these three measures, we compare the models' perfomances along edge, vertex and graph level indices.

We compare the two main models of interest to three additional models. As a baseline, we consider the simple Erdös-Rényi (i.e., Bernoulli) graph where the probability of an edge is constant and set equal to the proportion of edges in the training networks (ErdosRenyi). This is a particularly parsimonious model in that it is given by a single parameter – the probability of an edge. The second model is a highly parameterized TERGM where the Hamiltonian is equal to the sum of the Hamiltonians from the


Edge Creation Function

Edge Loss Function

Figure 5: Bootstrap estimates of edge creation and loss functions. Blue lines are placed at the point estimates and gray region spans 95% confidence intervals.

other two TERGMs (Combined). The third additional model we include fixes the degree distribution of the network (Degree). This model places equal probability on each graph with the same in and out-degree distributions as the training network. We use the 107^{th} Senate, a popular baseline model in the literature on community detection [31], as the training network. It is given by an ERG distribution with Hamiltonian equal to $\ln\left[\kappa(\boldsymbol{G}, \boldsymbol{G}^0)\right]$, where $\kappa(\boldsymbol{G}, \boldsymbol{G}^0) = 1$ if $\boldsymbol{G}$ has the same degree distribution as the training network $\boldsymbol{G}^0$ and 0 otherwise. Figure 6 gives box plots of the mean absolute difference of the forecast statistics computed on the 1,000 simulated networks and the statistics computed on the 108th Senate cosponsorship network.

The edge history and single year dependence TERGMs perform comparably in the forecast experiment. The dependence TERGM is the best performing model when it comes to predicting edge values and the graph transitivity, but the edge history model performs best at predicting the degree of Senators. The dependence TERGM shows a slight edge in that it is the only model that out-performs the Erdös-Rényi and fixed degree distribution model on each metric.[11] The combined model does not perform well – indicating that just combining the dependence Hamiltonian with the edge history Hamiltonian constitutes over-fitting [32].

7. Conclusion

Exponential family random graph models and their time serial extensions can be used to model networks with many interesting generative properties. Tuning of these models requires the maximization of the likelihood function. The (typically) intractable partition function necessitates the use of approximation methods to accomplish this task. MPLE represents an alternative to simulation methods that is computationally efficient, deterministic, and offers a consistent estimator.

The major drawback to the conventional implementation of MPLE is that confidence intervals are biased downward, meaning they overstate certainty in the parameter estimates. This can result in poor inferences about the processes generating networks of interest. We proposed a method by which

[11] It may appear odd that the fixed degree distribution model performs poorly on the degree forecast metric. However, recall that this metric measures the model's ability to predict the specific degree of each vertex, not the degree distribution overall (i.e., it is a vertex-level, not a graph-level metric).

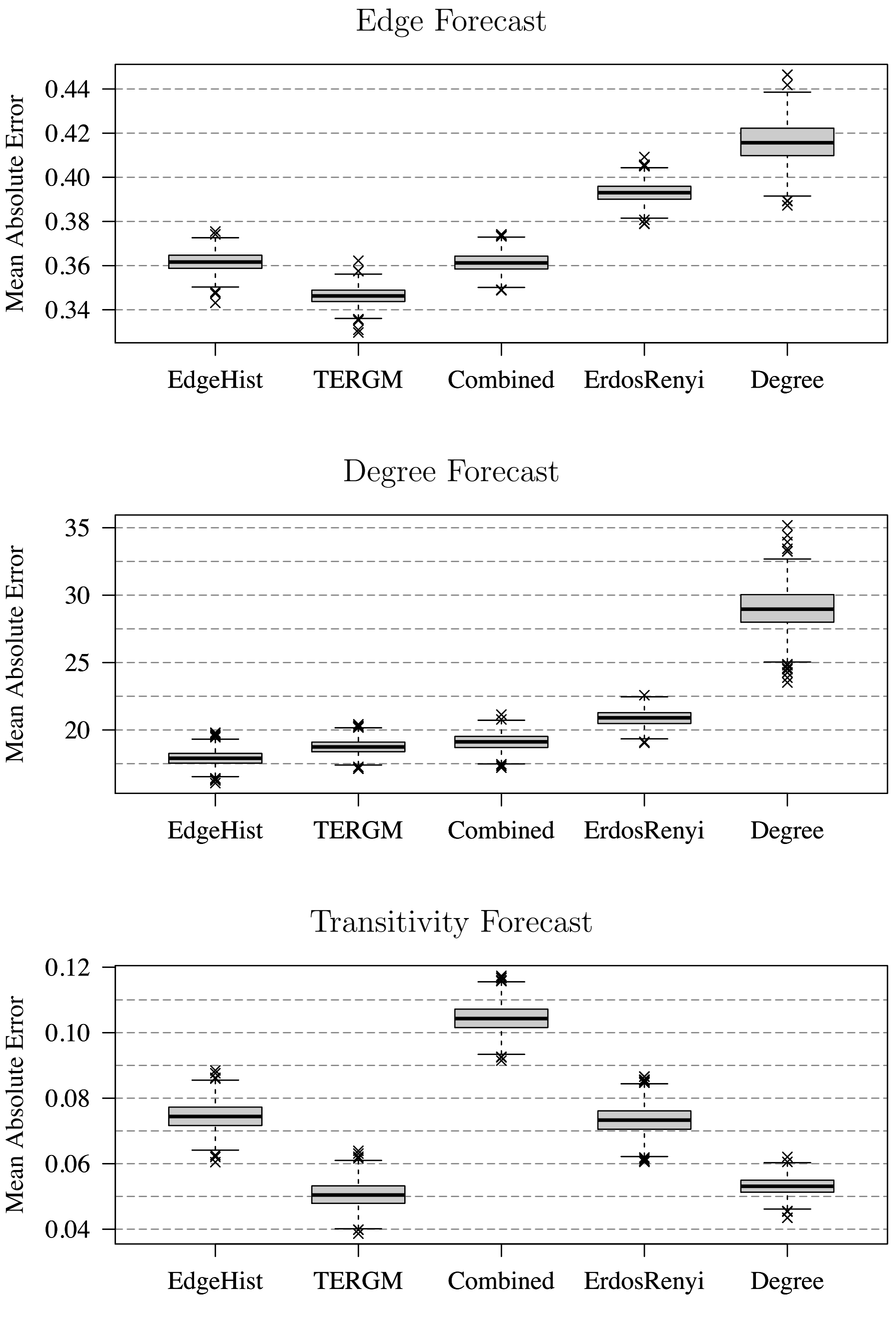


Figure 6: Comparison of forecast networks to observed 108th Senate cosponsorship network. Box plots depict the distribution of the mean absolute forecast error in 1,000 simulations. Horizontal black lines are placed at the median. Boxes span the first to third quartiles. Outer lines span two standard deviations from the mean.

the nonparametric bootstrap can be used to construct consistent confidence intervals for the MPLE and examined its efficacy. In a Monte Carlo study, we found that the method works well in finite samples.

We also introduced ERGM for dynamic networks, TERGM, to the physics literature. We illustrated the utility of the bootstrap method, as well as the flexibility of the TERGM, on cosponsorship networks in the U.S. Senate. We showed how recently proposed methods for modeling network dynamics can be integrated into the TERGM framework and how the bootstrap can be used to characterize uncertainty about the process generating the empirical networks. These methods are useful in statistical mechanics in that they can be used to (1) construct and train models that represent different and multiple generative processes, (2) directly compare alternative models based on forecast accuracy, and (3) accurately characterize uncertainty about the estimates derived from empirical observation.

Looking forward, several extensions of the ERGM would be valuable. First, improvements in estimation routines that could allow for ERGM analysis of large networks is a natural next step. Furthermore, an extension of the ERGM/TERGM methods to handle weighted networks [33]– networks in which edges can assume multiple values, as they are currently limited to discrete-valued networks, would greatly expand the range of network systems that may be modeled using this technology. Because the space that an MCMC-MLE estimation routine would have to explore to estimate an exponential family random graph model on a valued network will be much larger than for binary networks (how much larger would depend on the granularity of the edge weighting), we suspect that MPLE will be a fruitful estimation strategy for such models. The bootstrap method we proposed could be directly applied to MPLE of an ERGM with weighted edges.

References

[1] Z. Burda, J. Jurkiewicz, A. Krzywicki, Network transitivity and matrix models, Phys. Rev. E 69 (2) (2004) 026106.

[2] P. Grindrod, M. Parsons, Social networks: Evolving graphs with memory dependent edges, Physica A 390 (21-22) (2011) 3970–3981.

[3] J. Park, M.E.J. Newman, Statistical mechanics of networks, Phys. Rev. E 70 (6) (2004) 066117.

[4] J. D. Noh, Percolation transition in networks with degree-degree correlation, Phys. Rev. E 76 (2) (2007) 026116.

[5] A. Fronczak, P. Fronczak, J. A. Hołyst, Thermodynamic forces, flows, and Onsager coefficients in complex networks, Phys. Rev. E 76 (6) (2007) 061106.

[6] B. Efron, Nonparametric estimates of standard error: The jackknife, the bootstrap and other methods, Biometrika 68 (3) (1981) 589–599.

[7] M. A. Serrano, M. Boguñá, Clustering in complex networks. I. General formalism, Phys. Rev. E 74 (2006) 056114.

[8] S. Wasserman, P. E. Pattison, Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p*, Psychometrika 61 (3) (1996) 401–425.

[9] S. Hanneke, W. Fu, E. P. Xing, Discrete temporal models of social networks, The Electronic Journal of Statistics 4 (2010) 585–605.

[10] J. Park, M. E. J. Newman, Solution of the two-star model of a network, Phys. Rev. E 70 (6) (2004) 066146.

[11] C. J. Geyer, E. A. Thompson, Constrained Monte Carlo maximum likelihood for dependent data, Journal of the Royal Statistical Society. Series B (Statistical Methodology) 54 (3) (1992) 657–699.

[12] T. A. Snijders, Markov chain Monte Carlo estimation of exponential random graph models, Journal of Social Structure 3 (2) (2002) 1–40.

[13] J. Besag, Spatial interaction and the statistical analysis of lattice systems, Journal of the Royal Statistical Society. Series B (Statistical Methodology) 36 (2) (1974) 192–236.

[14] D. Strauss, M. Ikeda, Pseudolikelihood estimation for social networks, Journal of the American Statistical Association 85 (409) (1990) 204–212.

[15] A. Hyvarinen, Consistency of pseudolikelihood estimation of fully visible Boltzmann machines, Neural Computation 18 (10) (2006) 2283–2292.

[16] M. A. J. van Duijn, K. J. Gile, M. S. Handcock, A framework for the comparison of maximum pseudo-likelihood and maximum likelihood estimation of exponential family random graph models, Social Networks 31 (1) (2009) 52 – 62.

[17] K. Faust, J. Skvoretz, Comparing networks across space and time, size and species, Sociological Methodology 32 (1) (2002) 267–299.

[18] Z. M. Saul, V. Filkov, Exploring biological network structure using exponential random graph models, Bioinformatics 23 (19) (2007) 2604–2611.

[19] S. N. Lahiri, On bootstrapping M-estimators, Sankhya: The Indian Journal of Statistics, Series A 54 (2) (1992) pp. 157–170.

[20] C. A. Field, A. H. Welsh, Bootstrapping clustered data., Journal of the Royal Statistical Society: Series B (Statistical Methodology) 69 (3) (2007) 369 – 390.

[21] J. J. Harden, A bootstrap method for conducting statistical inference with clustered data., State Politics and Policy Quarterly 11 (2) (2011) 223–246.

[22] P. J. Huber, Robust Statistics, 2nd Edition, Wiley, New York, 2009.

[23] S. J. Cranmer, B. A. Desmarais, Inferential network analysis with exponential random graph models, Political Analysis 19 (1) (2011) 66–86.

[24] Y. Zhang, A. J. Friend, A. L. Traud, M. A. Porter, J. H. Fowler, P. J. Mucha, Community structure in Congressional cosponsorship networks, Physica A 387 (7) (2008) 1705–1712.

[25] M. S. Handcock, D. R. Hunter, C. T. Butts, S. M. Goodreau, P. N. Krivitsky, M. Morris, ergm: A package to fit, simulate and diagnose exponential-family models for networks, Seattle, WA, version 2.2-7. Project home page at `urlhttp://statnetproject.org` (2010).
URL `http://CRAN.R-project.org/package=ergm`

[26] M. A. Porter, P. J. Mucha, M. E. J. Newman, A. J. Friend, Community structure in the United States House of Representatives, Physica A 386 (1) (2007) 414 – 438.

[27] T. Richardson, P. J. Mucha, M. A. Porter, Spectral tripartitioning of networks, Phys. Rev. E 80 (3) (2009) 036111.

[28] J. H. Fowler, Connecting the Congress: A study of cosponsorship networks, Political Analysis 14 (4) (2006) 456–487.

[29] D. Kessler, K. Krehbiel, Dynamics of cosponsorship, The American Political Science Review 90 (3) (1996) 555–566.

[30] P. W. Holland, S. Leinhardt, Some evidence on the transitivity of positive interpersonal sentiment, American Journal of Sociology 77 (6) (1972) 1205–1209.

[31] M. E. J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2004) 026113.

[32] D. D. Jensen, P. R. Cohen, Multiple comparisons in induction algorithms, Machine Learning 38 (3) (2000) 309–338.

[33] X. Sun, E. Feng, J. Li, From unweighted to weighted networks with local information, Physica A 385 (1) (2007) 370 – 378.