Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: 1061-8600 (Print) 1537-2715 (Online) Journal homepage: https://www.tandfonline.com/loi/ucgs20

Stable Multiple Time Step Simulation/Prediction From Lagged Dynamic Network Regression Models

Abhirup Mallik & Zack W. Almquist

To cite this article: Abhirup Mallik & Zack W. Almquist (2019): Stable Multiple Time Step Simulation/Prediction From Lagged Dynamic Network Regression Models, Journal of Computational and Graphical Statistics, DOI: 10.1080/10618600.2019.1594834

To link to this article: https://doi.org/10.1080/10618600.2019.1594834

JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS
https://doi.org/10.1080/10618600.2019.1594834

Taylor & Francis
Taylor & Francis Group

Stable Multiple Time Step Simulation/Prediction From Lagged Dynamic Network Regression Models

Abhirup Mallik[a] and Zack W. Almquist[b]

[a]School of Statistics, University of Minnesota, Minneapolis, MN; [b]Department of Sociology and eScience, University of Washington, Seattle, WA

ABSTRACT

Changes in computation and automated data collection have greatly increased interest in statistical models of dynamic networks. Many of the models employed for inference on large-scale dynamic networks suffer from limited forward simulation/prediction capabilities. One major problem with many of the forward simulation procedures is a tendency for the model to become degenerate in only a few time steps, that is, the simulation/prediction procedure results in either null graphs or complete graphs. Here, we describe an algorithm for simulating a sequence of networks generated from lagged dynamic network regression models DNR(V), a subfamily of TERGMs. Further, we introduce a smoothed estimator for forward prediction based on smoothing of the change statistics obtained for a dynamic network regression model. We focus on the implementation of the algorithm, providing a series of motivating examples with comparisons to dynamic network models from the literature. We find that our algorithm significantly improves multistep prediction/simulation over standard DNR(V) forecasting. Furthermore, we show that our method performs comparably to existing more complex dynamic network analysis frameworks (SAOM and STERGMs) for small networks over short time periods, and significantly outperforms these approaches over long time time intervals and/or large networks. Supplementary materials for this article are available online.

ARTICLE HISTORY
Received July 2017
Revised February 2019

KEYWORDS
Dynamic networks; ERGM; Logistic regression; Logit; Network simulation; TERGM

1. Introduction

Dynamic network analysis, prediction, and simulation have a long history in statistics, computer science, and the sciences (e.g., Almquist and Butts 2013, 2014b; Farmer et al. 1987; Foulds et al. 2011; Casteigts et al. 2011; Goetz et al. 2009; Hanneke, Fu, and Xing 2010; Kolar et al. 2010; Krivitsky 2012; Leskovec 2008; Snijders 2005, 1996; Zimmermann, Eguìluz, and San Miguel 2004). Interest in dynamic systems arises from change in either the relation of interest (e.g., friendship) or the nodes (e.g., individuals). In the social sciences, dynamic network models have been used to understand important issues of disease transmission (e.g., sexual contact networks Morris [1993]), information transmission (e.g., communication during disasters Butts [2008]), and other important phenomena (e.g., friend formation, peer influence, etc. [McFarland et al. 2014; Centola 2010]). In statistics and computer science, new methods and models have been developed for understanding panel data (e.g., Kolar et al. 2010), sampled data (e.g., Ahmed and Xing 2009; Almquist and Butts 2017), and continuous time data (Butts 2008). In the physical sciences and engineering, dynamic network models have been employed to understand server load, and other complex systems. Recently, the advent of "Big Data"—that is, large-scale behavioral trace data—have increased the interest in scalable models such the lagged logistic regression models introduced by Robins and Pattison (2001) and expanded by Hanneke, Fu, and Xing (2010), Cranmer and Desmarais (2010), Desmarais and Cranmer (2010), Leifeld, Cranmer, and Desmarais (2015), Almquist and Butts (2014a), Almquist and Butts (2017), and others. Currently, methods for multistep forecasting and simulation from classic DNR models is quite limited, but has ready applications in the social sciences (e.g., agent-based modeling [Helbing 2012], prediction [Liben-Nowell and Kleinberg 2007], and simulation based experimentation [Rahmandad and Sterman 2012]) as well as applications to computer science (e.g., predicting server load [Prodan and Nae 2009]).

Lagged dynamic network logistic-regression (DNR) models provide a scalable framework for inference on large scale temporal networks collected as panel data (e.g., network data collected hourly, daily, weekly, monthly, etc. [Almquist and Butts 2013]). In addition, DNR models readily allow for missing data (Almquist and Butts 2017) and vertex dynamics (DNR(V)) (e.g., change in the network via population dynamics [Almquist and Butts 2014b]). DNR models are a subset of the *Temporal Exponential-family Random Graph Models* (TERGM) (Hanneke, Fu, and Xing 2010) and are employed in computer science (Kolar et al. 2010), social science and the physical sciences (Almquist and Butts 2013; Desmarais and Cranmer 2010) to great effect. DNR models are also conceptually similar to vector autoregressive (VAR) models and depend only on the past and exogenous variables, and therefore do not require information on the current time point such as the general TERGM case which makes them an ideal framework for problems of prediction.

CONTACT Zack W. Almquist zwa@uw.edu Department of Sociology and eScience, University of Washington, 410 Spokane Ln, Seattle, WA 98105.
Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/r/JCGS.
Supplementary materials for this article are available online. Please go to www.tanfonline.com/r/JCGS.

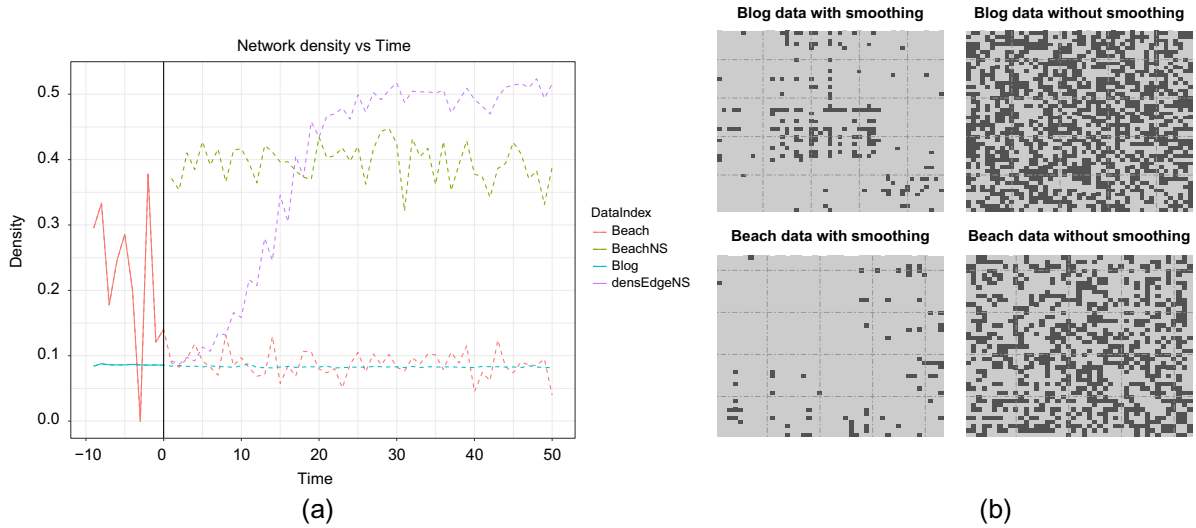


Figure 1. Comparison of simulated networks for Beach and Blog data with or without smoothing. (a) Time series plot of network density of original sequence of networks and 50 time point forward simulation. Negative time points indicate original networks. Dotted lines indicate density of simulated networks. "NS" in ending indicate "no-smoothing" for the lines in the plot. Without smoothing, networks seem to saturate. (b) Visualization of adjacency matrix of simulated network at time point 50 for Blog and Beach data, with and without smoothing. Both plots indicate that without smoothing, simulated networks seem to saturate.

A common problem with using DNR for either simulation modeling or prediction is that the model often leads to degenerate results (e.g., all networks are predicted to complete or unconnected) in only a few time steps (Hanneke, Fu, and Xing 2010). This is a standard problem in the larger literature in statistical network models (for a review of this problem see the work of van Duijn, Gile, and Handcock [2009] or Schweinberger and Handcock [2015]). The issue of instability in simulation of networks is quite common in the network literature. In Figure 1, we show the effect of our proposed smoothing algorithm on simulated networks for both fixed and dynamic vertex cases. Here we have simulated up to time point 50 in future for both cases. We plot the network density of the simulated networks with time as well as the final network adjacency matrix. It is clear that without smoothing, the networks seem to saturate. In several of our experiment, we have also noticed the case when without smoothing the networks become more sparse eventually becoming a disconnected network. This effect in its turn affects the future predictions, as the network statistics that are calculated from these network statistics will also display tendency toward degeneracy; this can be readily observed in Figure 3.

In this article, we introduce a smoothed estimator for forard prediction, based on smoothing of the change statistics (see Section 3 for details) obtained from a dynamic network regression model. We focus on the implementation of the algorithm, providing a series of motivating examples with comparisons to dynamic network models from the literature. We find that our algorithm significantly improves multi-step prediction/simulation over standard DNR forecasting. Furthermore, we show that our method performs comparably to existing more complex dynamic network analysis frameworks (stochastic actor oriented models and separable temporal exponential random graph models) for small networks over short time periods, and significantly outperforms these approaches over long time time intervals and/or large networks.

In the following sections, we will begin by introducing the general TERG model and subfamily DNR (with and without vertex dynamics). We then cover our setting for inference, and our smoothing algorithm for prediction and simulation. This is then followed by a comparison of DNR prediction with our smoothing algorithm and without our smoothing algorithm. Next, we compare the predictive properties of our algorithm against the two main competitors in the dynamic network literature—(i) the separable temporal exponential random graph model (STERGM) (Krivitsky and Handcock 2014), and (ii) the stochastic actor oriented models (SAOM) (Snijders 1996)—on key metrics in the computer science and social network literatures. Finally, we concluded with brief discussion of our findings.

2. Dynamic Network Analysis

A dynamic network is composed of entities (e.g., actors, respondents, computers, etc.) and relations (e.g., friendship, communication, needle-sharing, etc.) and is typically represented as the mathematical object known as a graph $G = (E, V)$, where E represents the set of relations and V represents a set of vertices. This can be readily extended to handle time by adding an index t. In practice, we represent a graph as binary adjacency matrix, $[Y]_{n\times n}$, and temporal network as time indexed array of adjacency matrices (typically the diagonal is treated as 0 or NA in most settings). In this work, we follow the notation of Almquist and Butts (2014b). We begin by considering networks with a fixed number vertices, that is, $|Y_t| = n$ for all time points t. (See Almquist and Butts [2014b] for a discussion of dynamic networks with vertex dynamics for comparison.)

In the following sections, we will use the notation Y_t as a random variable denoting an adjacency matrix, with an instance of this random variable denoted by y_t. The shorthand notation Y_a^b is used to denote the set of adjacency matrices $(Y_a, \ldots, Y_b)$. We use X_t to denote covariates associated with edges of a network

Y_t. The function $s(.)$ will be used to denote the set of sufficient statistics for an ERGM model described next.

2.1. Temporal Exponential-Family Random Graph Models

The framework for TERGMs is based on extending the classic exponential random graph models (ERGM) (Holland and Leinhardt 1981; Handcock 2003) to the temporal case via a VAR-type process with a kth order temporal Markov assumption. This assumption is as follows, for all times t, $Y_t \mid Y_{t-1}, \ldots, Y_{t-k}$ is independent of $Y_{t-k-1}, \ldots$ (Almquist and Butts 2014b), and allows us to write the TERGM likelihood in the following form (following Almquist and Butts's [2014b] notation)

$$\begin{aligned}\Pr(Y_t = y_t \mid Y_{t-k}^{t-1} = y_{t-k}^{t-1}, X_t) \\ = \frac{\exp\left(\theta^T s(y_t, y_{t-k}^{t-1}, X_t)\right)}{\sum_{y_t' \in \mathcal{Y}_t} \exp\left(\theta^T s(y_t', y_{t-k}^{t-1}, X_t)\right)} \mathbb{I}_{\mathcal{Y}_t}(y_t),\end{aligned} \tag{1}$$

for y_t belonging to the support, $\mathcal{Y}_t$. s here is a vector of real-valued sufficient statistics with parameter vector θ, and $Y_{t-k}^{t-1} = Y_{t-1}, \ldots, Y_{t-k}$. Notice that the denominator of (1) is intractable in the general case as it is in the static ERGM.

There have been two core methods of inference attempted for the TERG models: (i) treat this as a pooled ERGM problem (likelihood), where the past time points are treated as covariates and estimate θ using MCMC-MLE (Geyer 1991) or psuedo-likelihood methods (Strauss and Ikeda 1990), and (ii) treat the transitions between time points as a separable process where one distinguishes between tie formation and tie dissolution—this is known as separable TERGM or STERGM and will be discussed later in this article.

2.2. Dynamic Network Regression

DNR as subfamily of TERGM makes the same VAR assumption, but also conditional independence given past information. This simplification has a number of advantages due to known issues in TERGMs of degeneracy (Schweinberger and Handcock 2015; van Duijn, Gile, and Handcock 2009) and scalability (Almquist and Butts 2014b). Further, this model lends itself naturally to problems of network prediction as it does not rely on the current time-step for inference and updating and handles missing data (Almquist and Butts 2017). Here, again, we follow the language of Almquist and Butts (2014b),

$$\begin{aligned}\Pr(Y_t \mid Y_{t-k}^{t-1}, \theta, s, X_t) \\ = \prod_{(i,j) \in V_t \times V_t}^{n} \text{Bern}\left(Y_{ijt} \left| \text{logit}^{-1}\left(\theta^T s(Y_{t-k}^{t-1}, X_t)\right)\right.\right),\end{aligned} \tag{2}$$

where Bern(.) is understood to be the Bernoulli pmf, $\mathbb{I}$ is the indicator function, X_t is a covariate set (potentially including dynamic latent variables, see supplement for discussion), $Y_{t-k}^{t-1} = Y_{t-1}, \ldots, Y_{t-k}$ is the graph structure given the vertex set from time $t-k$ to $t-1$, and s is the sufficient statistics for the graph with θ being the real valued parameters of interest. Typical examples of sufficient statistics constructed from the edges of a set of networks can be found in the documentation of Hunter et al. (2008) and many of them have been implemented in the R package *ergm*.

2.3. Dynamic Network Regression With Vertex Dynamics

DNR can be extended to handle vertex dynamics in a natural way through a separability condition introduced by Almquist and Butts (2014b). Similar to the notation of edges, we use V_t to denote the set of vertices at time point t for the graph $G_t = (E_t, V_t)$. Vector of sufficient statistics for the vertices is calculated using the function $W(.)$ and the corresponding coefficients for the likelihood would be denoted by a vector ψ. So we will represent a graph at time t using (V_t, Y_t). If we take $N_v = |\cup V_t|$ to be the maximal set of nodes and Y_a^b denote the adjacency matrix in a time series $Y_a, \ldots, Y_b$ then we can write $P(G_t|G_{t-k}^{t-1}, \psi, \theta, w, s, X_t) = P(V_t = v_t|G_{t-k}^{t-1}, \psi, w, X_t) \times P(Y_t = y_t|V_t = v_t, G_{t-k}^{t-1}, \theta, s, X_t)$. Following this logic, Almquist and Butts proposed a dynamic network regression with vertex dynamics (DNRV) as a double logistic process

$$\begin{aligned}&P(G_t|G_{t-k}^{t-1}, \psi, \theta, w, s, X_t) \\ &\quad = P(V_t = v_t|G_{t-k}^{t-1}, \psi, w, X_t) \\ &\qquad \times P(Y_t = y_t|V_t = v_t, G_{t-k}^{t-1}, \theta, s, X_t) \\ &\quad = \frac{\exp(\psi^T w(v_t, G_{t-k}^{t-1}, X_t))}{\sum_{v' \in \mathcal{V}} \exp(\psi^T w(v_t', G_{t-k}^{t-1}, X_t))} \\ &\qquad \times \frac{\exp(\theta^T s(y_t, v_t, Y_{t-k}^{t-1}, X_t))}{\sum_{y' \in \mathcal{Y}_{v_t}} \exp(\theta^T s(y_t, v_t, G_{t-k}^{t-1}, X_t))} \\ &\quad = \prod_{(i) \in V_t}^{N_v} \text{Bern}\left(V_{it} \left| \text{logit}^{-1}\left(\psi^T w(G_{t-k}^{t-1}, X_t)\right)\right.\right) \\ &\qquad \times \prod_{(i,j) \in V_t \times V_t}^{n} \text{Bern}\left(Y_{ijt} \left| \text{logit}^{-1}\left(\theta^T s(Y_{t-k}^{t-1}, X_t, V_t)\right)\right.\right).\end{aligned} \tag{3}$$

Here, the sufficient statistics for vertex model is denoted by $w(.)$ and for the edge model it is $s(.)$. The coefficients for the vertex set is given by ϕ and the edge model coefficients are given by θ as in the previous model. The model is conditional on fixed lag k, which determines the previous state of the networks given by Y_{t-k}^{t-1}. The set of covariates X_t can encode any exogenous covariates for the model. The multiplicative nature of this model implies that they are separable for inference. So, the parameters for the vertex model and the edge model conditional on vertex can be interpreted separately.

2.4. Brief Discussion of Model Assumptions and Parameterization

DNR(V) generally assumes that much of the graph dependence can be captured in the past and generally follows the logic of VAR type process. DNR(V) with current time points will be the classic psuedo-likelihood estimator which has been shown to have issues in estimation of the parameter and standard errors (Hunter, Krivitsky, and Schweinberger 2012). Recently, Cranmer and Desmarais (2010) and others have employed the psuedo-likelihood estimator with a bootstrap to improve parameter and standard error estimation; however, Almquist and Butts (2013) demonstrated that full TERGM estimated with the bootstrap estimator with standard specifications generally

does not outperform the DNR(V) model based on predictive validity checks. Currently, model specification is determined through social science theory and predictive model assessment (see, e.g., Almquist and Butts 2014b). In this work we specify the model with common statistics chosen from the social science literature which are comparable across the different statistical models so as to allow for comparability.

In the literature, Almquist and Butts (2014b) considered a set of lagged statistics for both the edge set (s) and vertex set (w). The authors focused on inertia (the lag term), lagged embeddedness (network clustering effects like triangles), popularity (lagged degree effects), and exogenous covariates such as gender in both the vertex and edge sets. Specific statistics are typified by the theory or problem at hand. For example in Almquist, Spiro, and Butts (2017), the authors considered multiple lags, cluster (embeddedness), and popularity (degree) edge statistics. In this work will focus a similar set of core network metrics which also have the feature of being implementable across the different dynamic network models.

3. Simulation/Prediction From DNR

Model specification is often done through expert judgment and/or theory (Schwarz 1978) along with formal statistical methods (e.g., likelihood ratio test, BIC, etc.). If we assume a known model specification and an empirical dataset we can simulate or predict from this model. In practice the parameter values are typically obtained from empirical network data, which can be estimated through either MLE (Almquist and Butts 2014b), Bayesian (Almquist and Butts 2014a), or penalized maximum likelihood methods (for a general discussion penalized methods see Tibshirani [1996] and Hans [2009]). Depending on the complexity of the model and the lag term, the number of coefficients to be estimated can be quite large, hence it is often a good idea to employ some feature selection methods for fitting the model. In this work, we employ Lasso regression (Friedman, Hastie, and Tibshirani 2010) to both infer the parameters and perform model selection on our *training* dataset and use the algorithm discussed in this section to predict the *held out* network panel data. For the current algorithm, we prespecify the length of the lag term. We use a collection of consecutive network panels approximately of the length of the maximum lag to predict the unknown network. This collection of networks will be referred to as a *window* and the algorithm shifts the window forward as we make future predictions. We have explored this space through simulation and found that while larger windows improve prediction a bit, the gains do not warrant the loss of useable data. In cases where one wants to simulate from a known generative procedure where prediction of a real-world network is not the goal one may employ a static ERGM to inform the initial time-points. The initial window is selected by the researcher. In instances when we have an input sequence of networks larger than the size of the window we calculate the change statistics of each window by shifting the window through the sequence of networks. The "change statistics" (Hunter et al. 2008) or "change scores" (Snijders et al. 2006) underlies the core estimation algorithm for general ERGM estimation. Hunter, Krivitsky, and Schweinberger (2012) derives the change score via the odds ratio of the conditional graph for each dyad such that $\text{Odds}_\theta(Y_{ij} = 1 | Y - (i,j) = y - (i,j)) = \exp\{\theta^T \Delta_{ij} s(y)\}$, where (i,j) is the i,jth edge. (It is also noted that this formulation allows for a "local" interpretation of ERGMs.) We then calculate the mean of these window of change statistics as our estimate of the (average) change statistic matrix (this results in what is effectively a moving average of the change score statistics). This matrix is used as predictor in our simulation/prediction algorithm.

3.1. Estimation of Sufficient Statistics for Prediction/Forecasting

Given a set of input parameters θ, we would like to be able to forecast the future networks. For this, we would be using the likelihood given in Equation (2) and we need to estimate the sufficient statistics $s(Y_{t-k}^{t-1})$. The least number of networks needed to estimate this sufficient statistics is at time points $(t-1), \ldots, (t-k)$, henceforth will be referred to as time window of length k. Hence, a simple way of estimating these would be to use the network statistics calculated based on $Y_{(t-k)}, \ldots, Y_{(t-1)}$. However, we have found this estimate is not a stable one. This is expected as network statistics calculated based on one instance of the realization of underlying probabilities is subjected to noise in that realization. As this quantity is quite essential for predicting the future states of the network, a poor estimate would result in poor quality simulations. We have demonstrated this in simulation studies later in Section 6.

Under the assumption that the model is sufficiently explaining the state of the network, we assume the set of sufficient statistics in a window of time points to be stable. In some cases this may be a strong assumption and in other cases a weaker assumption. For example, it is typical to assume stable mean degree in static and dynamic networks (Butts and Almquist 2015) which would show up as a stable effect in our model. In other cases this smoothing (given an appropriately chosen window) may thought of as an approximation to true temporal effect. Recent work by Lee, Li, and Wilson (2017) attempts to loosen the assumption of homogeneity on the parameter space and lets evolve over time. Such an alternative formulation could be very useful when networks are rapidly evolving or simply out of equilibrium. Under these assumptions, we propose the following estimates for the network statistics.

$$\hat{s}(Y_{t-k}^{t-1}, X_t) = \frac{1}{(t-k)} \sum_{\tau=k+1}^{t} s(Y_{\tau-k}^{\tau-1}, X_\tau). \tag{4}$$

We then plug in this estimator in the likelihood in Equation (2) to produce the future states of the networks. Using this smoothing window we are able to obtain future predictions/simulations from our model which have better properties than pure DNR/DNRV. We summarize this algorithm as follows:

The researcher chooses the initial sequence of networks of size l, and our maximum lag size is k, we start simulating the $(k+1)$th network from the first l networks. Also, we assume that the parameters supplied are obtained from the same sequence of networks of length l. We then use the likelihood specified in

Table 1. Simulation results from various scenarios, with initial network size of (10, 20, 50, 100).

	Network size 10			Network size 20			Network size 50			Network size 100		
Parameter	Initial	Mean	SD	Initial	Mean	SD	Initial	Mean	SD	Initial	Mean	SD
DNC_{11}	−6.28	−5.05	0.54	−5.21	−4.39	0.53	−5.21	−4.39	0.53	−5.11	−3.13	0.37
DNC_{01}	−6.20	−5.04	0.53	−4.87	−4.33	0.52	−4.87	−4.33	0.52	−4.36	−3.03	0.30
DNC_{10}	−6.26	−5.07	0.54	−4.77	−4.28	0.51	−4.85	−4.41	0.48	−4.45	−3.16	0.31
DNC_{00}	−6.31	−5.09	0.55	−4.77	−4.28	0.51	−4.77	−4.28	0.51	−3.95	−2.91	0.30
TriadCensus(Y_{t-1})	−0.03	−0.01	0.01	0.02	0.04	0.01	0.02	0.04	0.01	0.04	0.08	0.01
tdcs2.1	−0.02	−0.01	0.00	0.02	0.02	0.01	0.02	0.02	0.01	0.04	0.05	0.01
tdcs3.2	0.02	−0.00	0.01	0.07	0.04	0.01	0.07	0.04	0.01	0.08	0.09	0.01
tdcs2.2	0.02	−0.00	0.00	0.04	0.03	0.01	0.04	0.03	0.01	0.04	0.05	0.01
Y_{t-1}	3.51	3.75	0.20	3.25	4.98	0.31	3.25	4.98	0.31	3.63	5.46	0.28
Y_2	6.90	4.28	0.87	8.13	5.14	0.43	8.13	5.14	0.43	8.19	5.45	0.39

Algorithm 1: Algorithm for simulating networks in static vertex case.

Input : $(G_1, \ldots, G_T), \theta, K, (X_1, \ldots, X_T)$
Output: $G_{T+1}, \ldots, G_{T+L}$
for *step* $l = 1$ *to* L **do**
  Estimate network statistics $\hat{s}(Y_{t+l-k}^{t+l-1}, X_t)$ using Equation (4);
  Generate Y_{t+l} from likelihood from Equation (2);
  Construct network $G_{t+l} = (V_{t+l}, E_{t+l})$, using adjacency matrix Y_{t+l}.;
end

Equation (2) to estimate the coefficients in the model. As mentioned in Section 3, selecting the generative features of a complex model is a quite hard problem (Tibshirani 1996), especially for dynamic network models, where the number of coefficients can be quite large depending on the set of sufficient statistics specified. To solve this problem, here we employ L_1 penalized likelihood methods for model selection. For constructing the predictor matrices for vertex and edge models, we use a moving window method and stack the matrices of network statistics together as the window moves forward. We use the notation $(w(.))_{t\in T}$ to denote the stacking operation on the matrices for the time index $t \in T$. We then use these stacked matrices in the likelihood equation (3) to estimate the coefficients (θ, ψ). For completeness we specify the algorithm for parameter estimation in Algorithm 2.

$$\tilde{w}(V_t, G_{t-k}^{t-1}, X_t) = \left(w(V_\tau, G_{\tau-k}^{\tau-1}, X_\tau)\right)_{\tau=k+1, V_\tau\in\mathcal{V}}^{t}, \quad (5)$$

$$\tilde{s}(Y_{t-k}^{t-1}, X_t) = \left(s(Y_{\tau-k}^{\tau-1}, X_\tau)\right)_{\tau=k+1}^{t}. \quad (6)$$

Algorithm 2: Algorithm for model selection and parameter estimation for Variable vertex models.

Input : $(G_1, \ldots, G_T), X_1, \ldots, X_T$
Output: θ, ψ
Construct $\tilde{w}(V_t, G_{t-k}^{t-1}, X_t)$ using Equation (5);
Match corresponding vertices;
Construct $\tilde{s}(Y_{t-k}^{t-1}, X_t)$ using Equation (6);
Solve for (θ, ψ) L_1 penalized logistic regression using the likelihood given in Equation (3);

It is to be noted that the smoothing estimator proposed in Equation (4) is just one of the possible smoothing estimators. To justify our use of mean as a smoothing estimator we compared the drift in estimates under several alternative smoothing estimators including median, minimum and maximum values of the network statistics. We have also compared with the estimate from maximum a posteriori probability by fitting a kernel density estimator to each element of the estimated network statistics. We have called this estimate as "Mode" as this is implementing similar idea as definition of mode.

In Figure 3, we show the plots of the parameters from the 100 iteration of the simulation engine. As we can see the parameter values decay differently based on which smoothing method was used. It is clear that using the mean as smoothing estimator produces the least drift in parameters. The means of the estimated parameters of the simulated networks are reported next to the input parameters in Table 1. Full details will be discussed in Section 6.

3.2. Bayesian Extension

Following similar development as Almquist and Butts (2014a), the likelihood equation in Equation (3) also allows us to do Bayesian inference in the usual way. We are interested in the posterior of (θ, ψ) given $Y_1, \ldots, Y_t$. The posterior can be written as

$$P(\psi, \theta | G_1^t, s, w, X) \propto P(\psi, \theta | s, w, X) \times \prod_{t=1}^{t} P(G_t | G_{t-k}^{t-1}, \psi, \theta, w, s, X_t). \quad (7)$$

For simplification, we would assume the prior on edges and vertex are independent conditional on X. So, this allows the following

$$\begin{aligned} P(\psi, \theta | G_1^t, s, w, X) &\propto P(\psi | w, X) \times P(\theta | S, X) \\ &\quad \times \prod_{t=1}^{t} P(G_t | G_{t-k}^{t-1}, \psi, \theta, w, s, X_t) \\ &= P(\psi | w, X) \frac{\exp(\psi^T w(v_t, G_{t-k}^{t-1}, X_t))}{\sum_{v'\in\mathcal{V}} \exp(\psi^T w(v'_t, G_{t-k}^{t-1}, X_t))} \\ &\quad \times P(\theta | S, X) \frac{\exp(\theta^T s(y_t, v_t, Y_{t-k}^{t-1}, X_t))}{\sum_{y'\in\mathcal{Y}_{v_t}} \exp(\theta^T s(y_t, v_t, G_{t-k}^{t-1}, X_t))}. \end{aligned} \quad (8)$$

So, the decomposition in Equation (8) allows us to specify the priors of the vertex and the edge model separately and use the joint likelihood from Equation (3) for calculating the posterior.

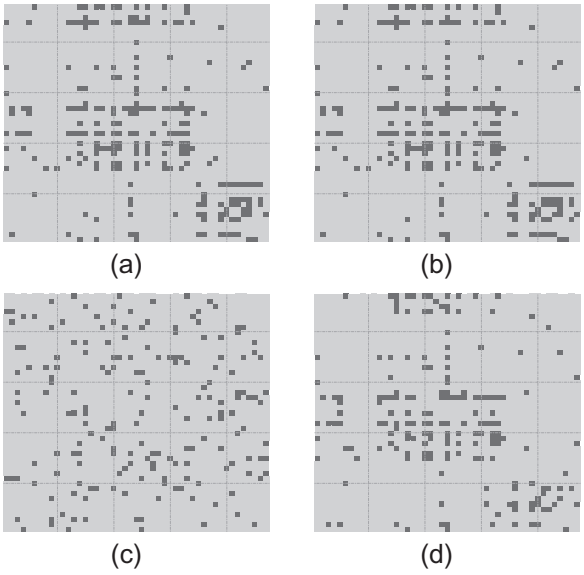


Figure 2. Bayesian prediction under model misspecification. (a) The network at time point 50, (b) at 51, (c) prediction with flat prior, and (d) prediction with data dependent prior. These plots indicate that using a data dependent prior seems to improve simulation accuracy under model misspecification.

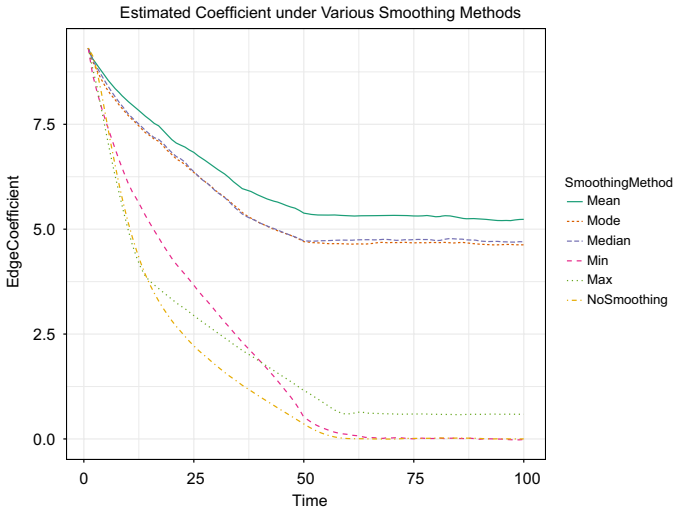


Figure 3. Comparison of simulation drift of estimates using various smoothing methods. We compare smoothing using (mean, median, mode, min, max) of the window of network statistics. It is clear that smoothing using mean results in most stable set of parameters in the simulated networks. Also, using Min/Max as smoothing strategy did not seem to improve the problem with degeneracy of the coefficients.

Besides making posterior inference possible, by specifying a prior it, also, helps us to better estimate the parameters where we have some information about them. In Figure 2, we compare the predicted networks with two kinds of priors. In this example, we have used a intercept only model with no lag terms and no other graph statistics. It can be logically argued that this is not a well suited model for this dataset. We have used the first 50 time points of the blog data for training the model. We show the next step prediction of the algorithm with a flat prior, which clearly shows that the predicted network is not close to the true network; however, the use of a prior computed from the previous data points vastly improves the results.

4. Simulation in Dynamic Vertex Case

In the case of dynamic (variable) vertices, we are using estimated vertex regression parameters from the observed networks. The number of parameters is fixed by the model and maximum considered lag. As in the parameter estimation case, the missing vertex statistics corresponding to the vertices were imputed with zeros. The matrices of vertex statistics will be used to construct the predictors for the vertices. We use a window based average as in fixed edge case to produce smoothed estimate of the vertex statistics. The coefficients estimated from the parameter estimation process are used to produce the predictor vector for the vertices. At each time point, the vertices are simulated as a Bernoulli trial with the corresponding probabilities of presence for each vertex.

For simulating the edges conditioned on the vertices we need to first simulate the vertex set then edge set to obtain the matrix of change statistics used for parameter estimation. As, we did not use all possible edges in the regression for parameter estimation, we would not have change statistics corresponding to some edges with possibly missing vertices in some time points in the training set. So, we impute the missing edges with zero and use mean smoothing to construct a stable estimator of the change statistics corresponding to all possible edges. They are used to get the edge probabilities using the estimated parameters. The edges are simulated from a Bernoulli trial with the edge probabilities. We keep any attributes associated with the vertices to be present in the simulated networks as well. It is assumed that the vertex attributes are not evolving with time. The set of generated edges are conditioned on the simulated vertex set. This method of generating edges conditioned on the vertices worked assuming the pattern of absence of the vertices are relatively uniform across all the time points.

For simulation in the dynamic vertex case, we need stable estimates of the network statistics for the same reasons as mentioned in Section 3.1. As we use the likelihood from Equation (3), we need to estimate the vertex statistics $w(.)$ and edge statistics $s(.)$. We follow the same strategy as the static vertex case to get a stable estimate of the vertex statistics. We propose the following estimate

$$\hat{w}(V_t, G_{t-k}^{t-1}, X_t) = \frac{1}{(t-k)} \sum_{\tau=k+1, v_\tau \in \mathcal{V}}^{t} w(V_\tau, G_{\tau-k}^{\tau-1}, X_\tau). \quad (9)$$

Here, we use the moving average on the vertex statistics to construct the estimate of statistic at time t. The networks in previous time points will be of different order, hence use the set of all vertices $\mathcal{V} = \cup_{t=1}^{T} V_t$ as the reference set of vertices. The absent vertex statistics are replaced by zero to compute the moving average equation (9). For a stable estimate of the edge statistics $S(.)$, we use the same estimate in static vertex case in Equation (4). Of course to calculate that estimate, we need to complete the previous networks with the bigger vertex set $\mathcal{V}$.

The algorithm summarizing both the case for vertex and edge simulation is presented.

Table 2. Comparison of network simulation algorithms for three datasets.

	Blog data			Davis data		Hospital data	
	Smooth	STERGM	SAOM	Smooth	STERGM	Smooth	STERGM
Misclassification	0.96	0.89	0.96	0.95	0.93	0.99	0.98
Precision	0.98	0.98	0.98	0.98	0.96	1.00	0.99
Recall	0.77	0.02	0.73	0.18	0.09	1.00	1.00
ΔTriangles	62.32	314.76	61.56	0.00	0.33	32.95	32.90
ΔClusterCoef	0.07	0.46	0.03	0	0.05	0	0
ΔExpDeg	0.58	7.02	0.90	1.12	1.05	0.60	1.38

NOTES: NAs represent network statistics which were not calculable under a given model. Smooth represents our smoothing algorithm for the DNR/TERG model. ΔTriangles denote the absolute difference of the number of triangles with the truth. Similarly for Cluster coefficient and Expected degree.

Algorithm 3: Algorithm for simulating networks in the dynamic vertex case.

Input : $(G_1, \ldots, G_T), \theta, \psi, K, (X_1, \ldots, X_T)$

Output: $G_{T+1}, \ldots, G_{T+L}$

for *step* $l = 1$ *to* L **do**

Estimate vertex statistics $\hat{w}(V_t, G_{t-k}^{t-1}, X_t)$ using Equation (9);

Generate vertex set V_{t+l} using likelihood Equation (3) ;

Estimate network statistics $\hat{s}(Y_{t+l-k}^{t+l-1}, X_t)$ using Equation (4);

Generate Y_{t+l} from edge likelihood conditional on V_{t+l} ;

Construct network $G_{t+l} = (V_{t+l}, E_{t+l})$, using adjacency matrix Y_{t+l}.;

end

5. Software Implementation

We implemented all software in R (R Core Team 2016). We use combination of custom code and software from the `statnet` (Handcock et al. 2003) software-suite for computing the change score statistics and the underlying simulation engine. For the estimation and model selection, we employed `glmnet` (Friedman, Hastie, and Tibshirani 2010) function, and wrote custom code posterior inference on the simulated networks. The associated R package for computation is available through Github at *https://github.com/SSDALab/dnr* or via CRAN (Mallik and Almquist 2018). All network measures compared in Table 2 were computed using the `igraph` package (Csardi and Nepusz 2006).

6. Simulation Study

To demonstrate the usefulness of our smoothing algorithm we focus on two key features: (i) we compare the stability of our algorithm with traditional methods, and (ii) how our algorithm functions in comparison to the two main temporal network competitors in the literature. To do this we employ three classic datasets in the social network literature (described in Section 6.1). To compare the forecasting ability of our algorithm with two existing methods in the literature, we take the standard machine learning approach of splitting the data in half for the training set and the holdout set. We then run all comparing methods to forecast the rest of the series. It is to be noted that for certain datasets, predicting far ahead in time can be challenging as the predictions usually converge to a limiting case. We experienced this in all the algorithms in our study, and the problem is largely one of model selection, that is, none of the models perform well when the feature set is poorly chosen.

The next several subsections lay out the real world data under consideration. The model features we consider for prediction within each dynamic network model, all focus on a limited set of features which are largely comparable across model types. Further, we restrict ourselves to a rather limited feature set to keep the discussion relatively parsimonious and understandable. Model prediction in all cases can be improved by considering a larger range of parameters—however many of these models have limited scalability and cannot perform either estimation or simulation on large number of features or time points. For this reason and general call to parsimony we focus on only a small set of lagged network statistics.

6.1. Description of the Datasets

We focus on three publicly available datasets for our comparison study. Each is discussed below.

6.1.1. Blog Citation Network

The *Blog Citation Network* is a temporal inter- and intra-group blog citation network collected by Butts and Cross (2009) and analyzed with DNR in Almquist and Butts (2013). This dynamic network consists of relations (hyperlinks) between all blogs credentialed by the U.S. Democratic National Committee (DNC) or Republican National Committee (RNC) for their respective 2004 conventions. Each of these conventions is paramount for selecting their individual presidential nomination for president of the U.S. The set of actors consists of 47 nodes with 34 DNC and 14 RNC credentialed blogs and 1 credentialed in both. This dataset consists of 484 time points covering 7/22/04 (shortly before the DNC convention) to 11/19/04 (shortly after the Presidential election). The data were collected in 6 hr increments consisting of the URLs linking the main page of one blot to any page within another blog. Here, we will consider various subsamples of the data. We refer to this dataset as the *blog data*.

6.1.2. Davis's Cocktail Party Data

The *Davis Cocktail Party Data* set is a classic social network originally collected and analyzed by Davis, Gardner, and Gardner (2009). The dataset covers social interaction between 18 women over a period of nine months in 14 informal events over the aforementioned period. The data records which women met

for which events. For our purpose, we have collapsed the data into monthly levels, having one network for each month. In our tests, we have used 6 months for training the models and used 3 remaining months for prediction and comparison. We refer to this dataset as the *cocktail party*.

6.1.3. Hospital Data

The *Hospital Data* is a relatively new dynamic social network originally collected and analyzed by Vanhems et al. (2013). It contains the network of contacts between the patients, and Health Care Workers (HCW) in a hospital ward in Lyon, France. The vertices are labeled with their role within the hospital, that is, Nurse, Patient, Doctor or Staff. The data were collected 12/6/2010 at 1:00 pm to 12/10/2010 at 2:00 pm at 20 sec intervals via RFID chips. We collapsed the time axis into hourly level to reduce the resolution of the data for our use. We have used first 50 time points of data for training and used next 20 time points for testing. We refer to this dataset as the *hospital data*.

6.1.4. Beach Data

The *Beach Data* is a classic network dataset, involving a dynamically changing network of interpersonal communications among the visitors of a Beach in Southern California. These data are observed over a one-month period producing 31 observations. There were 95 members being observed and in an average 15 people were present in one day with the maximum presence being 37. The proportion of edges in an average network was about 30%. This dataset is considered in detail by Almquist and Butts (2014b) describing their model of dynamic logistic regression with vertex dynamics (DNRV). We will refer to this dataset as the *beach data*.

6.2. Alternative Models

While there are a number of potential formulations for dynamic network models for both estimation and simulation, we focus on the two which are in common use and implemented in software packages, both of which are have been implemented in `R`, available to larger social network field. First, we consider STERGM and SAOM, and note that both these models rely on similar underlying framework to the DNR model and can be constructed within a similar parameter space for ready comparison. Further, we want to point out that both STERGM and SAOM rely on the current time points as well as the past time points which means they will have more information for both inference and prediction than the DNR model.

6.2.1. Separable TERGM

The separable TERGM was introduced by Krivitsky and Handcock (2014), and has been implemented in R as part of the `statnet` suite of software. STERGM is based around the positing of two models for dynamic networks: one for tie formation and a second for tie dissolution. This is done through composing an ERGM style formulation for both the formation and dissolution process. This formulation further relies on assuming that a researcher has observed two components, (i) a cross-sectional network, and (ii) a mean relational duration. This model is fit via MCMC-MLE methods (see, Krivitsky and Handcock 2016, for details).

6.2.2. Stochastic Actor Oriented Models

The stochastic AOM was originally developed by Snijders (1996), and published in software as SIENA (Snijders et al. 2007). It was later made available in R through `rsiena` (Ripley, Boitmanis, and Snijders 2013). This work has been developed substantially by Snijders (2001, 2002, 2005, 2011), Snijders and Van Duijn (1997), Snijders et al. (2006), and Mercken et al. (2010). The underlying assumptions of the stochastic AOM is that a dynamic network arises as a cross-sectional samples from a latent continuous time Markov process in which an actor's possible ties and behavior constitute the state space—this latent process is then simulated via a Markov chain Monte Carlo algorithm. (For computational details see Ripley, Boitmanis, and Snijders 2013). This framework was largely developed to de-couple questions of influence versus selection (e.g., the relationship between smoking and network structure [Lakon et al. 2015]).

6.3. Model Features for Simulation Study

Because the underlying framework for obtaining the sufficient statistics for the lagged network panel data is derived from the ERGM formulation developed for `statnet` (Handcock et al. 2003) we use similar model term discussions. However, our model is described by the maximum lag period and the parameters for each lagged time period (including the present time for the STERGM and Siena models). Structurally, the model can be decomposed into five components.

- Fixed effects: terms that are fixed across time. Examples include the intercept (edge or density), degree or sender effects. The change statistics for these terms will be denoted by I_δ.
- Group: The categorical predictors for each edge (these represent homophily terms and stand in for the standard edge or density term). The change statistics for these terms will be denoted by G_δ.
- Model terms: The model for time t network. These terms are used to specify the type of model that a static ERGM (this is used for SAOM and STERGM), for a detail descriptions of the terms possible here, we refer to the `statnet` documentation (Handcock et al. 2008). The change statistics for these terms will be denoted by $s(Y_\delta)$. Here $s(.) : \mathbb{G} \mapsto \mathbb{R}^q$ is the sufficient statistics for the network in the classic ERGM model.
- Lagged model terms: The model terms corresponding to past networks. These models are same as the current models, however their presence is controlled by a binary matrix, called lag matrix. This allows for finer control on the model specification. The lag matrix $M \in \{0,1\}^{L\times q}$, where L and q are the maximum lag and the number of network statistics in the model.
- Lagged networks: The networks from previous time point up to a finite lag. These terms will be denoted by Y_{t-j} for a network of lag j at time t. The lag terms can also be selected

and this is achieved by a binary lag vector of size L, with 1 in the elements indicating the presence of the corresponding lag term.

Combining all the terms mentioned, we can write the model as follows

$$\begin{aligned}\text{logit}[P(Y_{L+t}|(Y_L,\ldots,Y_{L+t-1}))] &= \sum_{i=1}^{n_I}\beta_{0i}I_{\delta i}+\sum_{i=1}^{n_q}\beta_{1i}G_{\delta i}+\sum_{i=1}^{q}\beta_{2i}s(Y_{(L+t)\delta i}) \\ &\quad+\sum_{i=1}^{q}\sum_{j=1}^{L}\beta_{3i}M_{ij}(Y_{(L+t-j)\delta i})+\sum_{j=1}^{L}\beta_{4j}L_j(Y_{L+t-j}).\end{aligned} \tag{10}$$

6.4. Simulation Design

We conduct simulation in four scenarios. These scenarios differ by the size of the starting network fed into the engine. We use initial network size of $(100, 20, 15, 10)$. The length of the predicted networks used in these four cases is 100. We replicated this experiment $(50, 100, 100, 100)$ times to produce the standard deviation of the average estimated parameters in each case. We calculate the averages using the coefficients extracted using the given model, from the first prediction, which essentially is same as the input networks. Hence, the first estimated coefficient is same as the input coefficients. We then report the mean of the time series of the coefficient series. We calculate standard deviation among all these means in different replications. We have devised 13 different models to fit while extracting the coefficients. We report the result from one of the cases here in Table 1. We compare all the results in the supplementary file.

7. Comparison of Smoothing Versus No Smoothing on Drift

We fit a simple model of Net $\sim$ Edges + Lag(1) to the blog dataset with first 50 time points as the starting network. We forecast the model for next 100 time points and estimate the parameters from the forecasted series. In the version of the nonsmooth prediction, we do not use the smoothed estimate of the change statistics, we only plug in the change statistics from the last step for that iteration. For the "mean" estimator, we use the smoothing estimator defined in Equation (4). For other estimators, we use element wise operation to get the median, min and max of the estimated network statistics. In Figure 3, we compare the time series of the estimated coefficient for the edge parameter of the predicted networks using different smoothing estimators. The figure justifies our use of mean of the network statistics as smoothing operator on the predictor matrix to slow down the decay of the parameters, reducing the degeneracy problem of network simulation.

8. Comparison With Related Methods

To compare the quality of the predictions among the algorithms we use several metrics common in the literature. The misclassification rate and precision recall metrics are calculated from the adjacency matrix of the predicted networks. We also report the number of triangles in the graphs and the clustering coefficient using the igraph library. Degree distribution of each vertex is also of importance, so we report the expected degree distribution for each graph.

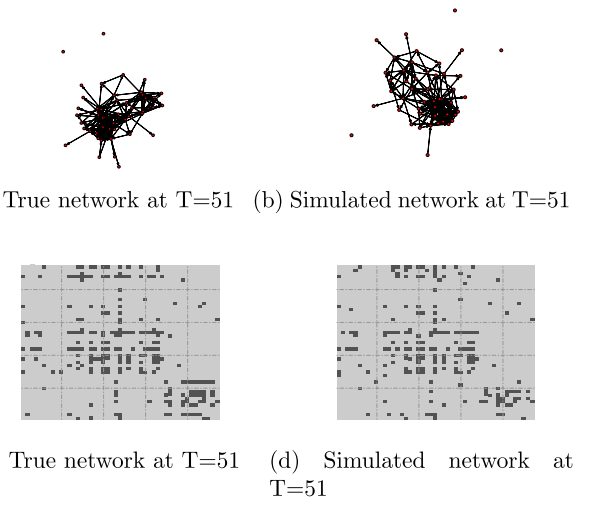
Each dataset has been split into a training and testing set. The size of the training data depends on the dataset and the testing set has always been immediately following the training split. For Blog citation dataset, we have used the first 50 days as training data, and the next 10 days as the testing set. For Davis's cocktail dataset, we have used first 6 months as training data to predict for next 3 months. For hospital data, we have used the first 50 time points as training set and the next 20 time points as the testing set. To keep the comparisons among different simulation algorithms, we needed to keep the model comparable. Hence, we chose simplest possible model for each dataset that has common parameters in each of the methods. The common model for all datasets used edges and triad terms as sufficient statistics. For example, for the Blog citation data, we have chosen a single intercept only model with edges for all methods. For the Davis data, we have chosen the model with edges and triad terms.

In Table 2, we compare the results using the above metrics for three simulation algorithms. In some cases the metric was not calculable, as some of the networks were possibly degenerate or the metric was not defined in those cases. We can see that the smooth ERGM algorithm performs competitively for the Blog data with SOAM (`RSiena`). In this case we used 50 time points to train the models, and the STERGM was the worst performer. In the other two cases, we could not train Siena as it ran into singularity issues while estimating the parameters. Therefore, we only report the results for Smooth TERGM/DNR and STERGM for those cases. The smooth algorithm is performing relatively better than STERGM, especially when the training network had a small size. In Figure 4, we compare the true network at timepoint 51 with the simulated network using smoothing algorithm.

The CPU time for each of the methods depended on the choice of the model used. All the computation in this document has been run in a computer with dual Intel Xeon E5-2670 processor each having 16 threads, and with 32GB of RAM. In models of comparative complexity, Siena took longest time for parameter estimation and simulations, taking almost twice as much time as DNR smooth algorithm. STERGM was slightly faster than smoothed ERGM method for parameter estimation. For simulation, STERGM was also slightly faster than DNR smooth algorithm. However, we do note that CPU time comparison would depend on the parameters chosen for each specific model and the size of the input networks. In our simulation study, we also kept running parameter estimation, so the CPU times are also affected by the estimation procedure.

9. Performance on Beach Data

We consider the *beach data* as an exemplar of the dynamic vertex case. We follow the DNRV framework for estimation, and are thus able to estimate the parameters as a joint logistic process. The set of statistics for the vertex case we included are the degree of each vertex; several measures of centrality



(a) True network at T=51 (b) Simulated network at T=51

(c) True network at T=51 (d) Simulated network at T=51

Figure 4. The comparison of the true network and the predicted networks for Blog data at first prediction time point. The method seems to produce networks relatively consistent with truth, we provide detailed accuracy comparison in the text.

(e.g., eigen centrality, between centrality, information centrality, and closeness centrality); we also included a clustering term, in this case the count of cycles. In our final model, we kept only the significant features in each lag level only. We considered maximum of lag 3. Another important feature for the vertex only model was the group membership of the people. Persons who were regular at the beach, had much higher chances of showing up at the next days.

For the conditional edge model, the important features were the lagged networks from previous time points, edge counts and the categorical nodal attribute variable on if the person was regular or not.

For selecting variables for the vertex model, we have separately modeled them as the vertex model is on the top of our model hierarchy (the full model can be viewed in Table 8). The selected variables are presented in Tables 3 and 7. The coefficient for the lag terms are all negative indicating possibly lower turnaround for consecutive days in the beach. All the coefficients for the lag terms are significant up to lag three. Interestingly, the coefficient for the indicator of week day and weekend effect is also negative. All the coefficients for the vertex parameters are positive. For variable selection for the edge model, we had to use conditional model. The only significant terms are the lag terms and the edge count. The signs for the lag terms in the edge model is mostly different from the vertex model.

In Table 4, we fit the model for beach data using 30 days of data and report the results on the 31st day. We have repeated the process for 30 times to compute the standard deviations.

We are also interested in evaluating the performance of the models across multiple prediction horizons. So, instead of using one step prediction we also simulated a sequence of networks from the prior networks. In this simulation, we have used the training data incrementally. This means, we take the first 22 days for training, then predict day 23, and in the next iteration, we take first 23 days for training and predict day 24 and so on. So, for each training dataset, we predict the rest of the observed future networks and make comparisons. For this experiment, we have used days 23–31 for prediction of

Table 3. Parameters for vertex model for Beach data.

	Estimate	SE	z value	Pr(>|z|)
lag1	−1.6955	0.3374	−5.02	0.0000
lag2	−3.2098	0.5982	−5.37	0.0000
lag3	−2.7903	0.4785	−5.83	0.0000
Day	−0.7995	0.0914	−8.75	0.0000
regularLag1	1.4481	0.3401	4.26	0.0000
regularLag2	2.2312	0.6027	3.70	0.0002
regularLag3	1.7980	0.4793	3.75	0.0002
EigenCentralityLag1	1.0703	0.2910	3.68	0.0002
ClosenessCentralityLag1.	2.3730	0.7405	3.20	0.0014
EigenCentralityLag2.	1.0499	0.3190	3.29	0.0010
EigenCentralityLag3.	0.9734	0.3223	3.02	0.0025

Table 4. Comparison of network metrics between smooth and nonsmooth algorithm for simulating network for one step prediction (time point 31).

	Smooth	NonSmooth	True
NVertices	40.16	44.40	34.00
	(4.60)	(3.99)	
Nedges	70.66	386.82	79.00
	(20.94)	(75.13)	
nTriangles	21.48	959.52	91.00
	(13.24)	(305.55)	
ClustCoef	0.14	0.41	0.59
	(0.04)	(0.02)	
ExpDeg	7.94	35.54	10.29
	(1.69)	(3.93)	

the networks. We present the results of this comparison in Table 5. It is apparent that for the sequential simulation case the results are better than one step prediction. Large standard deviation for this case compared to one step case is resulting from the wide variation between days. The day effect was highly significant in our vertex model and was used in the simulation model to account for the variation among the weekdays and weekends.

For testing the performance of long range prediction, we forecast next 50 time points of the beach data using both the smooth and nonsmooth version of the algorithm. In the nonsmooth version, instead of using the average predictor matrix, we only use the final value of the predictor matrix. We then compute the network statistics from the simulated networks and compare with the observed network statistics from the beach data. We present the results in Table 6. In most metrics, the smooth version of simulations is much closer to the past metrics. We also observe that the nonsmooth version of the algorithm produces a much denser network, with number of edges far exceeding the past number of edges. This problem is less prominent in the smooth version of algorithm. The SD of the metrics from the smooth networks are also less as this produces a stabler results.

10. Discussion

Here we have introduced a novel technique for improving network simulation and prediction for DNR(V) models and finally we have compared these results against the current state of the art in statistical models for dynamic network data. In addition, we have, as far as the authors are aware, been the first to use penalized likelihood methods for model selection in DNR(V)

Table 5. Comparison of metrics from simulation of sequence of networks using incremental training data.

	Triangles	ClusterCoef	ExpectedDegree
Simulated	85.38	0.71	35.24
SD	43.89	0.30	19.47
True	161.38	2.57	29.07

Table 6. Comparison between smooth and nonsmooth version of dynamic vertex algorithm using average of the testing data.

	Smooth	Beach	NonSmooth
Number of vertices	44.86	15.67	44.88
	(6.08)	(7.99)	(3.73)
Number of edges	93.92	29.13	396.76
	(31.27)	(27.35)	(72.71)
Number of triangles	35.62	35.27	1017.44
	(25.86)	(47.29)	(319.00)
Cluster coefficient	0.15	0.65	0.41
	(0.04)	(0.15)	(0.02)
Expected degree	9.23	7.23	36.11
	(1.94)	(3.48)	(3.68)

Table 7. Parameters for conditional edge model for Beach data.

	Estimate	SE	*z* value	Pr(>|*z*|)
Edges	−3.3624	0.0397	−84.77	0.0000
Lag1	−0.1140	0.2098	−0.54	0.5870
Lag2	1.1424	0.1721	6.64	0.0000
Lag3	5.1041	0.1101	46.35	0.0000

Table 8. Parameters used for one step prediction.

	Estimate	SE	*z* value	Pr(>|*z*|)
lag1	−1.7012	0.3423	−4.97	0.0000
lag2	−3.1539	0.5976	−5.28	0.0000
lag3	−2.8204	0.4802	−5.87	0.0000
Day	−0.8550	0.1005	−8.51	0.0000
attrib1	1.3694	0.3451	3.97	0.0001
attrib2	2.1787	0.6028	3.61	0.0003
attrib3	1.7883	0.4802	3.72	0.0002
Vstat4Lag1.	1.2435	0.3013	4.13	0.0000
Vstat7Lag1.	2.4526	0.7467	3.28	0.0010
Vstat4Lag2.	0.9250	0.3271	2.83	0.0047
Vstat4Lag3.	1.0612	0.3263	3.25	0.0011
edges	−0.5421	0.5456	−0.99	0.3205
edgecov.regular11	0.0001	0.0003	0.37	0.7087
edgecov.regular00	−0.0006	0.0004	−1.47	0.1418
edgecov.regular10	0.0000	0.0003	0.01	0.9960
logCurrNetSize	−0.6399	0.1524	−4.20	0.0000
dayEffect	0.5746	0.0658	8.74	0.0000
lag2	4.6833	0.0866	54.08	0.0000

framework in contrast with typical AIC/BIC methods employed currently in the field. Given that quality of feature selection in ERGMs and TERGMs is based on in-sample prediction of macro-level graph statistics (typically those not in the feature set), it suffices to show good predictive validity for demonstrating the usefulness of this technique for model selection. The appeal of this formulation for model selection is that it is more readily scalable as penalization performs model selection in a single model run, and thus does not require one to attempt to a full factorial design (or subsample) of possible model parameters which could be quite large (e.g., lag statistics k, graph statistics l, and exogenous variables v). As Tibshirani (1996) advised in seminal work one can apply the penalized methods to obtain parsimonious model by dropping the features that have weights approximately zero and refining the model with either standard likelihood based or Bayesian methods to obtain unbiased estimates of the parameters.

We find that DNR(V) performs favorably in comparison to STERGM and SAOM such that when SAOM and STERGM are performing at their best DNR(V) does comparable and when they are at their worst DNR(V) performs better on average. Computation time is always a very important aspect of dynamic network modeling and DNR(V) compares well to both models, though is a bit slower to STERGM (though we suspect this is due to its weaker integration with `ergm` package in R and graph statistics chosen).

This article extends the DNRV model introduced by Almquist and Butts (2014a) by improving its ability to simulate and predict in comparison to simple DNRV originally introduced. We believe this method will set the basic bar for which future dynamic vertex models will have to clear in the area of simulation and prediction.

As a proposal for future work, it should also be possible to extend this method for weighted graphs. Although this would require new hierarchy based on Equation (3), as the weights would need to be generated from a positive valued distribution conditional on the vertex and edge distributions. However, the general approach for using smoothing based on window of sufficient statistics can also be used in this case. It would be interesting to extend this study for weighted graphs as it would open up many possibilities for application of network simulation in real world situations.

Finally, this method adds greatly to the dynamic network literature and allows for the direct simulation and prediction of dynamic networks from DNR(V) models. Our results demonstrate that our method improves the prediction/simulation of multiple time steps from a DNR(V) process. This will allow in the future the ability to perform detailed sensitivity tests to measurement process underlying dynamic network data collection and for simulation based experiments centered around dynamic network data (e.g., collaboration during a disaster [Butts 2008] or communication patterns over time).

Supplementary Materials

The supplementary material provides the R scripts that have been used to generate the figures and tables of this paper. This includes all libraries and any other recursive dependency for the packages and all necessary data.

Funding

This work was supported in part by an ARO YIP Award # W911NF-14-1-0577.

References

Ahmed, A., and Xing, E. P. (2009), "Recovering Time-Varying Networks of Dependencies in Social and Biological Studies," *Proceedings of the National Academy of Sciences*, 106, 11878–11883. [1]

Almquist, Z. W., and Butts, C. T. (2013), "Dynamic Network Logistic Regression: A Logistic Choice Analysis of Inter- and Intra-group Blog

Citation Dynamics in the 2004 US Presidential Election," *Political Analysis*, 21, 430–448. [1,3,7]

——— (2014a), "Bayesian Analysis of Dynamic Network Regression With Joint Edge/Vertex Dynamics," in *Bayesian Inference in the Social and Natural Sciences*, eds. I. Jeliazkov and X.-S. Yang, New York: Wiley. [1,4,5,11]

——— (2014b), "Logistic Network Regression for Scalable Analysis of Networks With Joint Edge/Vertex Dynamics," *Sociological Methodology*, 44, 1–33. [1,2,3,4,8]

——— (2017), "Dynamic Network Analysis With Missing Data: Theory and Methods," *Statistica Sinica*, 28, 1245–1264. [1,3]

Almquist, Z. W., Spiro, E. S., and Butts, C. T. (2017), "Shifting Attention: Modeling Follower Relationship Dynamics Among US Emergency Management-Related Organizations During a Colorado Wildfire," in *Social Network Analysis of Disaster Response, Recovery, and Adaptation*, Philadelphia, PA: Elsevier, pp. 93–112. [4]

Butts, C. T. (2008), "A Relational Event Framework for Social Action," *Sociological Methodology*, 38, 155–200. [1,11]

Butts, C. T., and Almquist, Z. W. (2015), "A Flexible Parameterization for Baseline Mean Degree in Multiple-Network ERGMs," *The Journal of Mathematical Sociology*, 39, 163–167. [4]

Butts, C. T., and Cross, B. R. (2009), "Change and External Events in Computer-Mediated Citation Networks: English Language Weblogs and the 2004 U.S. Electoral Cycle," *The Journal of Social Structure*, 10, 1–29. [7]

Casteigts, A., Flocchini, P., Quattrociocchi, W., and Santoro, N. (2011), "Time-Varying Graphs and Dynamic Networks," in *Ad-hoc, Mobile, and Wireless Networks*, Berlin, Heidelberg: Springer, pp. 346–359. [1]

Centola, D. (2010), "The Spread of Behavior in an Online Social Network Experiment," *Science*, 329, 1194–1197. [1]

Cranmer, S. J., and Desmarais, B. A. (2010), "Inferential Network Analysis With Exponential Random Graph Models," *Political Analysis*, 19, 66–86. [1,3]

Csardi, G., and Nepusz, T. (2006), "The igraph Software Package for Complex Network Research," *InterJournal, Complex Systems*, 1695, 1–9. [7]

Davis, A., Gardner, B. B., and Gardner, M. R. (2009), *Deep South: A Social Anthropological Study of Caste and Class*, Columbia, SC: University of South Carolina Press. [7]

Desmarais, B. A., and Cranmer, S. J. (2010), "Consistent Confidence Intervals for Maximum Pseudolikelihood Estimators," in *Proceedings of the Neural Information Processing Systems 2010 Workshop on Computational Social Science and the Wisdom of Crowds*. [1]

Farmer, J. D., Kauffman, S. A., Packard, N. H., and Perelson, A. S. (1987), "Adaptive Dynamic Networks as Models for the Immune System and Autocatalytic Sets," *Annals of the New York Academy of Sciences*, 504, 118–131. [1]

Foulds, J. R., DuBois, C., Asuncion, A. U., Butts, C. T., and Smyth, P. (2011), "A Dynamic Relational Infinite Feature Model for Longitudinal Social Networks," *Journal of Machine Learning Research—Proceedings Track*, 15, 287–295. [1]

Friedman, J., Hastie, T., and Tibshirani, R. (2010), "Regularization Paths for Generalized Linear Models via Coordinate Descent," *Journal of Statistical Software*, 33, 1–22. [4,7]

Geyer, C. J. (1991), "Markov Chain Monte Carlo Maximum Likelihood," Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface, pp. 156–163. [3]

Goetz, M., Leskovec, J., McGlohon, M., and Faloutsos, C. (2009), "Modeling Blog Dynamics," in *ICWSM*. [1]

Handcock, M. S. (2003), "Statistical Models for Social Networks: Inference and Degeneracy," in *Dynamic Social Network Modeling and Analysis*, eds. R. Breiger, K. M. Carley, and P. Pattison, Washington, DC: National Academies Press, pp. 229–240. [3]

Handcock, M. S., Hunter, D. R., Butts, C. T., Goodreau, S. M., and Morris, M. (2003), *statnet: Software Tools for the Statistical Modeling of Network Data*, Seattle, WA. [7,8]

——— (2008), "statnet: Software Tools for the Representation, Visualization, Analysis and Simulation of Network Data," *Journal of Statistical Software*, 24, 1–11. [8]

Hanneke, S., Fu, W., and Xing, E. P. (2010), "Discrete Temporal Models of Social Networks," *Electronic Journal of Statistics*, 4, 585–605. [1,2]

Hans, C. (2009), "Bayesian Lasso Regression," *Biometrika*, 96, 835–845. [4]

Helbing, D. (2012), "Agent-Based Modeling," in *Social Self-Organization*, Berlin: Springer, pp. 25–70. [1]

Holland, P. W., and Leinhardt, S. (1981), "An Exponential Family of Probability Distributions for Directed Graphs," *Journal of the American Statistical Association*, 76, 33–50. [3]

Hunter, D., Handcock, M., Butts, C., Goodreau, S., and Morris, M. (2008), "ergm: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks," *Journal of Statistical Software*, 24, 1–29. [3,4]

Hunter, D. R., Krivitsky, P. N., and Schweinberger, M. (2012), "Computational Statistical Methods for Social Network Models," *Journal of Computational and Graphical Statistics*, 21, 856–882. [3,4]

Kolar, M., Song, L., Ahmed, A., and Xing, E. P. (2010), "Estimating Time-Varying Networks," *The Annals of Applied Statistics*, 4, 94–123. [1]

Krivitsky, P. N. (2012), "Modeling of Dynamic Networks Based on Egocentric Data With Durational Information," Technical Report Series 12-01, The Pennsylvania State University, University Park, PA. [1]

Krivitsky, P. N., and Handcock, M. S. (2014), "A Separable Model for Dynamic Networks," *Journal of the Royal Statistical Society*, Series B, 76, 29–46. [2,8]

——— (2016), *tergm: Fit, Simulate and Diagnose Models for Network Evolution Based on Exponential-Family Random Graph Models*, R Package Version 3.4.0, The Statnet Project, available at *http://www.statnet.org*. [8]

Lakon, C. M., Wang, C., Butts, C. T., Jose, R., Timberlake, D. S., and Hipp, J. R. (2015), "A Dynamic Model of Adolescent Friendship Networks, Parental Influences, and Smoking," *Journal of Youth and Adolescence*, 44, 1767–1786. [8]

Lee, J., Li, G., and Wilson, J. D. (2017), "Varying-Coefficient Models for Dynamic Networks," arXiv no. 1702.03632. [4]

Leifeld, P., Cranmer, S. J., and Desmarais, B. A. (2015), "Temporal Exponential Random Graph Models With btergm: Estimation and Bootstrap Confidence Intervals," *Journal of Statistical Software*, 83, 1–36. [1]

Leskovec, J. (2008), *Dynamics of Large Networks*, Ann Arbor, MI: ProQuest. [1]

Liben-Nowell, D., and Kleinberg, J. (2007), "The Link-Prediction Problem for Social Networks," *Journal of the Association for Information Science and Technology*, 58, 1019–1031. [1]

Mallik, A., and Almquist, Z. (2018), *dnr: Simulate Dynamic Networks Using Exponential Random Graph Models (ERGM) Family*, R Package Version 0.3.2. [7]

McFarland, D. A., Moody, J., Diehl, D., Smith, J. A., and Thomas, R. J. (2014), "Network Ecology and Adolescent Social Structure," *American Sociological Review*, 79, 1088–1121. [1]

Mercken, L., Snijders, T., Steglich, C., Vartiainen, E., and de Vries, H. (2010), "Dynamics of Adolescent Friendship Networks and Smoking Behavior, Social Networks," *Social Networks*, 32, 72–81. [8]

Morris, M. (1993), "Epidemiology and Social Networks: Modeling Structured Diffusion," *Sociological Methods & Research*, 22, 99–126. [1]

Prodan, R., and Nae, V. (2009), "Prediction-Based Real-Time Resource Provisioning for Massively Multiplayer Online Games," *Future Generation Computer Systems*, 25, 785–793. [1]

R Core Team (2016), *R: A Language and Environment for Statistical Computing*, Vienna, Austria: R Foundation for Statistical Computing. [7]

Rahmandad, H., and Sterman, J. D. (2012), "Reporting Guidelines for Simulation-Based Research in Social Sciences," *System Dynamics Review*, 28, 396–411. [1]

Ripley, R., Boitmanis, K., and Snijders, T. (2013), *RSiena: Siena—Simulation Investigation for Empirical Network Analysis*. R package, version 1.2-3. [8]

Robins, G., and Pattison, P. (2001), "Random Graph Models for Temporal Processes in Social Networks," *Journal of Mathematical Sociology*, 25, 5–41. [1]

Schwarz, G. (1978), "Estimating the Dimension of a Model," *The Annals of Statistics*, 6, 461–464. [4]

Schweinberger, M., and Handcock, M. S. (2015), "Local Dependence in Random Graph Models: Characterization, Properties and Statistical Inference," *Journal of the Royal Statistical Society*, Series B, 77, 647–676. [2,3]

Snijders, T. A. (1996), "Stochastic Actor-Oriented Models for Network Change," *Journal of Mathematical Sociology*, 21, 149–172. [1,2,8]

——— (2001), "The Statistical Evaluation of Social Network Dynamics," in *Sociological Methodology*, eds. M. Sobel and M. B. Boston, London: Basil Blackwell, pp. 361–395. [8]

——— (2002), "Markov Chain Monte Carlo Estimation of Exponential Random Graph Models," *Journal of Social Structure*, 3, 1–40. [8]

——— (2005), "Models for Longitudinal Network Data" (Chapter 11), in *Models and Methods in Social Network Analysis*, eds. P. Carrington, J. Scott, and S. Wasserman, New York: Cambridge University Press. [1,8]

——— (2011), *Multilevel Analysis*, Berlin, Heidelberg: Springer. [8]

Snijders, T. A. B., Pattison, P. E., Robins, G. L., and Handcock, M. S. (2006), "New Specifications for Exponential Random Graph Models," *Sociological Methodology*, 36, 99–153. [4,8]

Snijders, T. A., Steglich, C. E., Schweinberger, M., and Huisman, M. (2007), *Manual for SIENA Version 3.1*, University of Groningen: ICS/Department of Sociology; University of Oxford: Department of Statistics. [8]

Snijders, T. A., and Van Duijn, M. (1997), "Simulation for Statistical Inference in Dynamic Network Models," in *Simulating Social Phenomena*, eds. R. Conte, R. Hegselmann, and P. Terna, Berlin: Springer, pp. 493–512. [8]

Strauss, D., and Ikeda, M. (1990), "Psuedolikelihood Estimation for Social Networks," *Journal of the American Statistical Association*, 85, 204–212. [3]

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society*, Series B, 58, 267–288. [4,5,11]

van Duijn, M. A. J., Gile, K., and Handcock, M. S. (2009), "Comparison of Maximum Pseudo Likelihood and Maximum Likelihood Estimation of Exponential Family Random Graph Models," *Social Networks*, 31, 52–62. [2,3]

Vanhems, P., Barrat, A., Cattuto, C., Pinton, J.-F., Khanafer, N., Régis, C., Kim, B.-A., Comte, B., and Voirin, N. (2013), "Correction: Estimating Potential Infection Transmission Routes in Hospital Wards Using Wearable Proximity Sensors," *PLoS One*, 8, e73970. [8]

Zimmermann, M. G., Eguìluz, V. M., and San Miguel, M. (2004), "Coevolution of Dynamical States and Interactions in Dynamic Networks," *Physical Review E*, 69, 65–102. [1]