Positional Estimation within the Latent Space Model for Networks [1]

Susan Shortreed
and
Mark S. Handcock
University of Washington, Seattle

Abstract

Recent advances in latent space and related random effects models hold much promise for representing network data. The inherent dependency between ties in a network make modeling data of this type difficult. In this paper we consider a recently developed latent space model that is particularly appropriate for the visualization of networks. We suggest a new estimator of the latent positions and perform two network analyses, comparing four alternative estimators. We demonstrate a method of checking the validity of the positional estimates. We also provide a development of the implementation of this model via a package in the freeware statistical language R. The package allows researchers to efficiently fit the latent space model to data, and to visualize the results.

KEY WORDS: Random graph models; Markov Chain Monte Carlo; Visualization

1 Introduction

Social network data is used to study actors and the relationships between them. Models designed to fit this data have been developed and used, yet few of these models behavior is well understood. In a recent paper, Hoff, Raftery and Handcock (HRH) suggested an approach to modeling networks based on the notion of a "social space" (Hoff et al., 2002). This is a canonical idea in social network theory that has a long history (McFarland and Brown, 1973; Faust, 1988). The HRH model posits the existence of an unobserved latent space of characteristics of the actors. In the particular, we focus here on social distances in the space measured by Euclidean distance.

This paper provides an expansion of the practical use of the latent space model for the social network community. In particular it investigates parameter estimates generated from fitting the model to two classic network social data sets, Padgett's Florentine Marriage data (Padgett and Ansell, 1993) and the Sampson's Monastery data (Sampson, 1968). Section 2 reviews a social network modeling framework and one of the dependency problems inherent in social data with which the latent space model specifically deals. Section 3 reviews the latent space model and the estimation of parameters. Section 4 focuses on the theory and practice of the algorithm used and evaluates four alternative point estimates of the positions in latent space. Section 5 contains an analysis of the Florentine undirected network and Monastery directed data.

As it is a primary purpose of this paper to illustrate the practical use of these models, we provide freeware that implements this model and was used for analysis in the paper. We provide a brief description of how to obtain the software and its use.

2 Stochastic Models for Social Networks

Social network data typically consist of a set of n actors and a relational tie y_{ij}, measured on each ordered pair of actors $i, j = 1, \ldots, n$. In the most simple cases, y_{ij} is a dichotomous variable, indicating the presence or absence of some relation of interest, such as friendship, collaboration, transmission of information or disease, etc. Here we focus on the case of a binary relationship and the $n \times n$ sociomatrix $y = [y_{ij}]$. $y_{ij} = 1$, will denote that there exists a relation from actor i to actor j, while $Y_{ij} = 0$ will denote that no such directed relation exists. x_{ij} will be the covariate vector associated with possible ties between actors i and j. This can be thought of as a graph in which the nodes are actors and the edge set is $\{(i, j) : y_{ij} = 1\}$. When (i, j) is in the edge set we can write $i \rightarrow j$.

The network matrix Y can be viewed as a random variable with a sample space of

$\mathcal{Y} \subseteq \{0,1\}^g$, where g is the total number of possible ties in a network. Each Y_{ij} can be treated as a Bernoulli random variable with marginal probability $p_{ij} = P(Y_{ij} = 1)$; it is this probability, as well as the joint distribution, these methods try to model (Hoff et al., 2002). The family of distributions most commonly used to model social networks is referred to as exponentially parameterized random graphs (Frank and Strauss, 1986). The probability mass function is:

$$P_{\boldsymbol{\theta}}(Y = y) = \frac{\exp\left[\boldsymbol{\theta}^T \mathrm{t(y)}\right]}{c(\boldsymbol{\theta})} \qquad y \in \mathcal{Y} \tag{1}$$

where $\boldsymbol{\theta} \in \Theta \subseteq \Re^q$ is the model parameter and $\mathrm{t} : \mathcal{Y} \to \Re^q$ are network statistics (Frank and Strauss, 1986). Under this model the $\mathrm{t(y)}$ are jointly sufficient for $\boldsymbol{\theta}$ and the model is the maximum entropy distribution based on the statistics. The normalizing function is

$$c\left(\boldsymbol{\theta}\right) = \sum_{y \in \mathcal{Y}} \exp(\boldsymbol{\theta}^T \mathrm{t(y)}).$$

A basic example of this model is the Bernoulli random graph. In this density $t(y_{ij}) = y_{ij}$ and θ_{ij} is the logit probability of the ij^{th} edge occurring. When all of the edges have a common probability this density simplifies to:

$$P(Y = y) = \frac{\exp(\theta \sum_{i \neq j} y_{ij})}{c(\theta)} \qquad y \in \mathcal{Y}$$

where $\theta = \theta_{ij} = \mathrm{logit}(P(Y_{ij} = 1))$ and $c(\theta) = n(n-1)(\log(1 + \exp(\theta)))$. This can be reduced to $\log(P(Y = y)) = \theta t(y) - \log(\mathrm{c}(\theta))$, where $t(y)$ is the total number of ties in the network

The Bernoulli random graph model is relatively simple because it assumes independence among all ties; unfortunately, this is rarely true. One of the goals in modeling social networks is to represent the structure of the graph as a function of covariates, which can be fairly straight forward in independence examples. A complication that arises in social data is the inherent dependency between ties. For example, if Jon and Peter are friends, and Kyle and Peter are friends, then it is more likely that Kyle and Jon are friends (or that any relationship exists) than if these previous relationships did not exist. This dependency is dealt with in various ways in models and sometimes even ignored altogether. The latent space model suggests a way to handle this dependency by introducing the idea of a "social space." Various characteristics (usually unknown) dictate actors positions in the social space. This idea of a latent space can help model the relationships between Jon, Peter, and Kyle. Since Jon and Peter are friends, they are most likely close to each other in this space, as are Peter and Kyle; the distance inequality implies Jon and Kyle are close as well. If we include distance in this latent space between actors when modeling the structure of the network, the probability that Jon and Kyle are friends will increase because their relative distance is small. In other words, by including the distance between two actors we are taking into account the inherent dependency among relations mentioned earlier (Hoff et al., 2002).

3 Latent Space Model

The central assumption of the latent space model is that an edge between actor i and actor j is independent of an edge between actor j and actor k given their relative positions in social space and covariates. In probabilistic notation: $Y_{ij} \perp Y_{jk} | Z_i, Z_j, Z_k, X_{ij}, X_{jk}$ where the Z's represent the positions of actors in the latent space. Under this assumption, we can now model the probability of an edge occurring between any two actors using logistic regression.

$$\text{logit}(Y_{ij} = 1 | \boldsymbol{\beta}, x_{ij}, d_{ij}) = \boldsymbol{\beta}' x_{ij} - d_{ij} = \eta_{ij}$$

where d_{ij} is the distance between i and j. The likelihood of the graph conditioned on $\boldsymbol{\eta}$ is then:

$$P(Y = y | \boldsymbol{\eta}) = \frac{\exp(\sum_{i \neq j} \eta_{ij} y_{ij})}{\sum_{i \neq j}(1 + \exp(\eta_{ij}))}$$

and the unconditional likelihood of the graph is:

$$P(Y = y) = \int P(Y = y | \boldsymbol{\eta}) P(\boldsymbol{\eta}) d\boldsymbol{\eta}$$

where $P(\boldsymbol{\eta})$ is the model for the distribution of the distances $\boldsymbol{\eta}$ prior to the data. The locations can be modeled directly for example with a mixture Gaussian distribution. Here we will posit a Gaussian prior for $\boldsymbol{\eta}$. Conditioning on the positions and the covariates gives us independence between actors. This allows summing over the log-likelihood of the individual ties in the matrix to get the log-likelihood of the entire network.

The true positions are in a dimensionless latent "social" space with an unknown distance metric. In implementing this model the Euclidean distance metric is used and the actors are positioned in $\Re^k$. In this model the distance matrix, denoted D, is used not the position matrix, denoted Z. Graphs with different positions for the actors but which yield the same relative distances should be treated as identical. Because of the reflection, rotation, and translation operators, there are infinitely many graphs which represent the same relative distances. In order to deal with this, and avoid overestimating the variability in the positions of the actors, the Procrustean transformation of a graph is used instead of the graph itself. The Procrustean transformation of a graph Z around a graph Z_0 changes the graph to have the same reflection, rotation, and translation properties as Z_0. For more details see Sibson (1979) and Hoff et al. (2002).

Described here is a Bayesian analysis, an analysis which fits a probability distribution to the parameters of interest as opposed to making a point estimate for the parameters and then evaluating the uncertainty of these estimates using repeated sampling theory, as is done in traditional Frequentist methods. A prior distribution on the parameters reflects prior beliefs or knowledge about the possible values that the parameters can take on. For

example, if it is known that a parameter value is positive and believed that its value is close to one, then a prior distribution that only has positive probabilities associated with positive values and has a mean of one, would reflect these beliefs about the parameter a priori data collection. A main goal of a Bayesian analysis is to describe a posterior distribution of the parameters, which is a combination of the prior distribution and the data collected. Prior distributions can be chosen to reflect confidence in the a priori opinions. If beliefs are strong beforehand (there have been many previous studies and knowledge is plentiful) then more weight can be place on the prior distribution than in a situation where little or no strong beliefs exist a priori. In many situations where the latent space model is used the latter is usually true. Thus diffuse or non-informative priors are chosen, these priors normally are distributions centered around a non-informative value (i.e. zero for covariate coefficients) with a large variance. In this situation the data should drive the posterior with the prior distribution having little effect. For more information on Bayesian statistical methods, see Gelman et al. (1995).

4   Algorithmic Development

4.1   Theory

As with many statistical models, the estimation of parameters is an important goal. In this model we would like to estimate both the positions of the actors in latent space and the covariate coefficients. This algorithm uses two techniques to do this. First, the parameters are estimated with the maximum likelihood (ML) method, by finding the value of the parameters, which maximize the likelihood function, this is the same as maximizing the log-likelihood function discussed in the previous section. ML estimation procedure is equivalent to finding the parameter values that give the highest likelihood to the data as observed. Second, Bayesian analysis starting with diffuse priors (π), is used to estimate the posterior distribution of the parameters by a algorithm called Markov Chain Monte Carlo (MCMC). Given certain conditions (all met here), the MCMC will converge to the posterior distribution, which means that the chain will be taking samples from the joint posterior distribution of the parameters. These samples can then be used to make inference about the parameter values and their marginal posterior distributions. The general algorithm used is as follows (Hoff et al., 2002):

1. Find the maximum likelihood estimate (MLE) for β, and Z matrix

2. Run a MCMC chain. To start, Set $k = 0$ and Z^0 to be the MLE of the Z matrix. Repeat steps a-c $N \times M$ times, storing $\{Z^{Nj}\}_{j=1}^{M}$. Here N is the interval between

4

samples and the result is a sample of size M from the posterior distribution of Z.

    (a) Sample a proposal $\check{Z}$ from an independence
        multivariate normal centered around Z^k

    (b) With probability equal to the minimum of 1 and

$$\frac{P(Y|\check{Z},\boldsymbol{\beta}^k,X)\pi(\check{Z})}{P(Y|Z^k,\boldsymbol{\beta}^k,X)\pi(Z^k)} \quad \text{Accept } \check{Z}^{k+1} \text{ to be } Z^{k+1}$$

        Otherwise $Z^{k+1} = Z^k$

    (c) Update $\boldsymbol{\beta}$ conditioned on Z^{k+1}

3. Store the Procrustean transformation of $\{Z^{Nj}\}_{j=1}^{M}$, the M samples of the position matrix from the posterior around Z_{mle}, denoted $\{\tilde{Z}^{Nj}\}_{j=1}^{M}$

    where $\tilde{Z}$ of Z around Z_{mle} is defined as: $\tilde{Z} = Z_{mle}Z'(Z'_{mle}Z_{mle}Z')^{-1/2}Z$

This algorithm is modified from the algorithm implemented by HRH, where the Procrustean transformation of the position matrix was taken during the chain after acceptance.

4.2   Implementation of the Algorithm

The latent space method described here is found in the R package `latentnet`. It has a R formula interface to improve ease-of-use and the MCMC computations are carried out internally in C. R is a free software widely used for statistical analysis that encapsulates a programming language that efficiently represents and executes many numerical computations. As with many higher level languages it can be slow for repetitive task done in a for loop, such as an MCMC, which is used in this analysis to sample from the posterior distribution of the parameters. Using C for the MCMC drastically reduces the time taken and the memory consumption. This increases the size of the network that can be fit.

    A benefit of interfacing C code with R is that available to the programmer are functions which are standard in the R libraries. All the random number generators used are part of R (written in C). The R math library includes `Fortran` functions from `LINPACK`, and R allows users to call these functions from C. This has aided in the computation of the Procrustean transformation which is defined as: $Z^* = \text{argmin}_{TZ}\{tr(Z_0 - TZ)'(Z_0 - TZ)\}$, where T ranges over all of the sets of rotations, reflections and translations (Sibson, 1979).

If both Z and Z_0 are centered around the origin Z^* is computed with the following formula: $Z^* = Z_0Z'(ZZ'_0Z_0Z')^{-1/2}Z$, where Z is a k by n matrix, k being the dimension of the space. The square root of a matrix, B is defined to be the matrix A such that $A^TA = B$. Using matrix decomposition techniques the matrix A can by found by determining the eigenvalues and eigenvector of B. Let U be a matrix of B's eigenvectors and D a diagonal matrix with

the square root of the eigenvalues on the diagonals, then $A = UDU^T$. The inverse of this matrix A is equivalent to using the formula above but replacing the diagonals of the matrix D with the inverse of the square root of the eigenvalues instead. This saves a significant amount of computation because it is no longer necessary to invert a matrix, which in high dimensions can be quite costly.

The coefficients β_j have a spherical Gaussian prior with user specified standard deviation and mean. Note this is a deviation from the original model presented by HRH since the intercept in this model had a gamma prior. The coordinates of the position matrix have a spherical Gaussian prior with a user specified mean and standard deviation. The proposal distribution used in the MCMC for the β_j's is Gaussian, centered around the last value with standard deviation, δ_{β_j}, for each β_j. The proposal distribution used in the MCMC for the coordinates of the positions is Gaussian centered around the last value of the coordinate, with the standard deviation, δ_Z. A more detailed discussion of the use of the `latentnet` package, as well as example code is contained in the Appendix.

4.3 Point Estimation of the Positions in Latent Space

In many situations point estimates for β and the positions are desired in addition to their posterior distributions. There are three natural candidates for point estimates for the parameters in the model: the maximum likelihood estimate (MLE), the posterior mean, and the posterior mode. However, as the model specifies the distances between actors only and not their locations, these estimates may be very poor representations of the positions. That is, even when we expect the distance between actors to be accurately determined, the locations themselves are not. As we shall see in the next section, the above three estimators can give poor estimates of the locations. To resolve this, we propose a fourth estimator being the positions in $\Re^k$ and the corresponding coefficient values which minimize the Kullback-Leibler (KL) divergence of those values to the mean posterior distance model. This will be referred to as the MKL estimate.

The Kullback-Leibler divergence of a distribution with probability mass function p from the distribution with probability mass function q is
$$E_q[\log(q) - \log(p)]$$

Define $\phi_{ij} = \beta^T X_{ij} - d_{ij}$, then it follows that $P_{\phi}(Y = y) = \frac{\exp(\phi^T y)}{c(\phi)}$. The Kullback-Leibler divergence, $KL(\phi, \eta)$, of the latent space model with parameter η from the latent space model with parameter ϕ is:

$$E_{\phi}\left[\log\left(\frac{P_{\phi}(Y = y)}{P_{\eta}(Y = y)}\right)\right] = \sum_{y \in \mathcal{Y}} \log\left(\frac{P_{\phi}(Y = y)}{P_{\eta}(Y = y)}\right) P_{\phi}(Y = y)$$

$$= \sum_{y \in \mathcal{Y}} (\boldsymbol{\phi}^T - \boldsymbol{\eta}^T) y P_{\boldsymbol{\phi}}(Y = y) + \log\left(\frac{c(\boldsymbol{\eta})}{c(\boldsymbol{\phi})}\right)$$

$$= (\boldsymbol{\phi} - \boldsymbol{\eta})^T E_{\boldsymbol{\phi}}[Y] + \log\left(\frac{c(\boldsymbol{\eta})}{c(\boldsymbol{\phi})}\right)$$

The MKL estimates are the parameter values, $\eta_{ij} = \boldsymbol{\beta}^T X_{ij} - (\sum_{l=1}^{k}(Z_{il} - Z_{jl})^2)^{1/2}$, minimizing the posterior mean of the Kullback-Leibler divergence:

$$
\begin{aligned}
E_{\boldsymbol{\phi}|Y_{obs}}[KL(\boldsymbol{\phi}, \boldsymbol{\eta})] &= \boldsymbol{\phi}^T E_{\boldsymbol{\phi}|Y_{obs}}[E_{\boldsymbol{\phi}}[Y]] - \boldsymbol{\eta}^T E_{\boldsymbol{\phi}|Y_{obs}}[E_{\boldsymbol{\phi}}[Y]] \\
&\quad - E_{\boldsymbol{\phi}|Y_{obs}}[\log(c(\boldsymbol{\phi}))] + E_{\boldsymbol{\phi}|Y_{obs}}[\log(c(\boldsymbol{\eta}))] \\
&= \boldsymbol{\phi}^T \; E_{\boldsymbol{\phi}|Y_{obs}}[E_{\boldsymbol{\phi}}[Y]] - \boldsymbol{\eta}^T E_{\boldsymbol{\phi}|Y_{obs}}[E_{\boldsymbol{\phi}}[Y]] - \log(c(\boldsymbol{\phi})) + \log(c(\boldsymbol{\eta}))
\end{aligned}
$$

Since $E_{\boldsymbol{\phi}|Y_{obs}}[E_{\boldsymbol{\phi}}[Y]] = E[Y|Y_{obs}]$ is the mean posterior probability of a tie under the model with parameter $\boldsymbol{\phi}$ and the first and third terms do not depend on $\boldsymbol{\eta}$, the optimization problem is simplified to finding the parameter values which maximize:

$$\frac{\exp(\boldsymbol{\eta}^T E[Y|Y_{obs}])}{c(\boldsymbol{\eta})}$$

This optimization is easily implemented using the likelihood routines already used in the algorithm. The posterior mean $E[Y|Y_{obs}]$ can be accurately estimated from the MCMC samples and does not require the positions, but the distances.

For each of the four estimators of the positions and β_0, the estimates are free to move around the parameter space without restriction. Thus, the positional estimates are similar but are stretched or contracted in comparison to one another. This lack of restrictions on the parameter space when maximizing the likelihood, usually results in the nodal positions being pushed far away from one another with β_0 increasing to compensate for the large distances. The prior distributions in the MKL method force more information to be present from the data to drastically increase β_0, thus the absolute distances are kept reasonably small. In ordered to compare these estimates, each position was normalized by setting the coordinates equal to $Z_{ij}^{norm} = Z_{ij} / \sqrt{\sum_{j=1}^{k} Z_{ij}^2}$

5   Application of the Latent Space Model

This section describes the analysis of two data sets using the latent space model and method described above. In each of the analyses, several values of the proposal δ's were tried, and the values which appeared to allow the parameters to move around the space without compromising the acceptance rates too greatly were chosen. Each data set is fitted with the latent space model, and the parameter estimates are investigated

5.1   Florentine Marriage Data

The first data set discussed is known as the Florentine Marriage data. This data was compiled by Padgett and Ansell (Padgett and Ansell, 1993), and focuses on 16 prominent Florentine families and their business and marriage relations. The data collected by Padgett covers a large time span as well as types of networks, but this analysis will concentrate on the marriage data from the 15th century. Each actor in the data set is a family and a tie is present if there existed at least one marriage between the families. For easier visualization the isolates were removed from the data before analysis, thus this analysis will be on 15 of the original 16 families. There are no covariates, so estimation includes β_0 and the position of each of the families.

The Markov Chain takes a sample of size 700 from the posterior distribution of the parameters, allowing 10,000 iterations for burnin and 1,000 iterations between samples. The proposal parameters for the chain were $\delta_{\beta_0} = 0.5$ and $\delta_Z = 0.3$. The prior distribution on β_0 is $N(0, 10^2)$ and the prior distribution on each of the coordinates Z_{ij} is also $N(0, 10^2)$. The plots of the log-likelihood values and β_0 at each sampled interval in the chain, seen in Figure 1, indicate convergence. The bottom two plots of Figure 1 are the marginal posterior distribution of β_0 and the positions of the actors.

Figure 2 displays the four different normalized estimates for the $\Re^2$ positions described in Section 4.3.

The basic structure of the graph is similar in all four estimates, the main differences is located on the left hand side of the graphs: actors 1, 10 and 13. These actors do not have many ties to other actors thus there is limited information contained in the network about their position in latent space. While the posterior distances are stable the positions are not; the posterior samples sometimes place actor 1 on top and actor 13 on the bottom, sometimes vice versa. This results in a bimodal posterior distribution for the positions of these actors. In Figure 1, it can be seen that the green points corresponding to actor 1, the pink points corresponding to actor 13 and the blue points corresponding to actor 10 form two distinct clusters each. It is well known that the mean and modes of distributions are poor point estimates for multi-modal distributions. This implies the MLE and MKL estimates better summarize the estimated positions of the actors, when there is multi-modality. While the normalized graphs are comparable the raw graphs are quite different the square root of the sum of squared distances in the MLE graphs is 68.63 and the MLE for β_0 is 14.45. The same quantities for the MKL method are 20.94 and 3.96. This dramatic difference between estimates demonstrates that the interpretation of β_0 is meaningless without the distances.

The number of ties in a graph is a basic network summary statistics and a good parameter estimate should produces graphs in which the observed number of ties (20 undirected ties
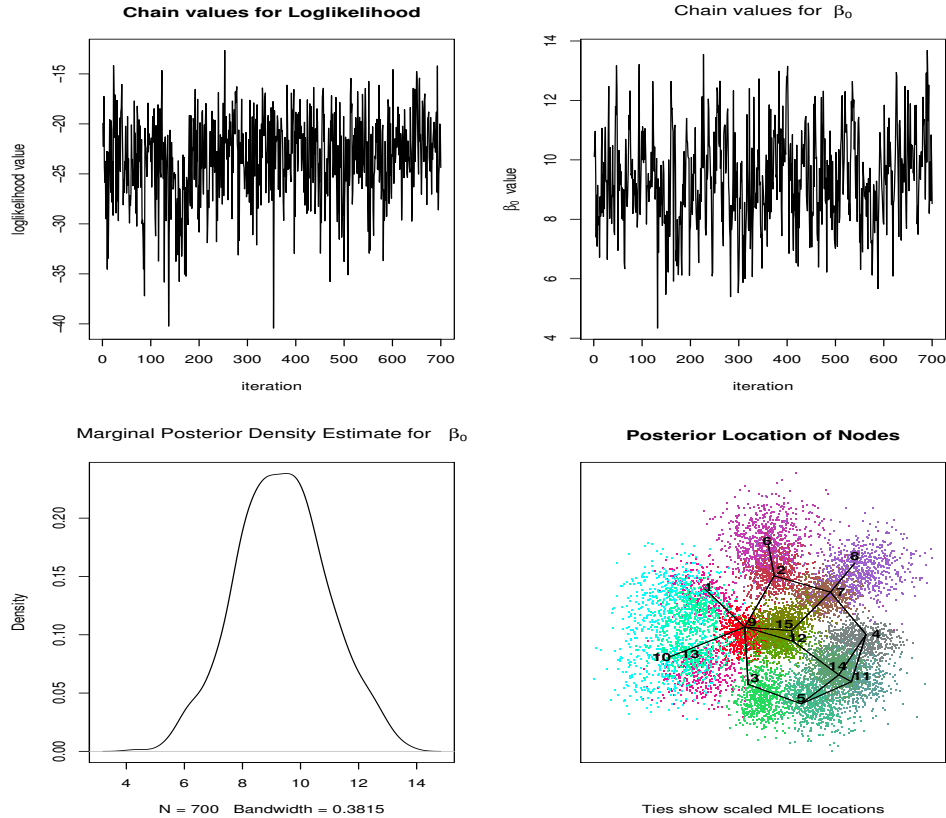
Figure 1: Labeled clockwise starting from upper left. (a) Chain values for the log-likelihood. (b) Chain values for β_0. (c)Marginal Posterior for β_0. (d) Marginal posterior for positions.

in the Florentine data) have a high probability. Figure 3 shows the density estimates of the number of ties produced from a probability distribution corresponding to the latent space model with each of the four parameter estimates. The MLE and MKL estimates appear to produce reasonable distributions on the number of ties, while the posterior mean and mode tend to generate graphs which have "too many" ties.

The better performance of the MKL could be due to the bi-modality of the posterior distribution of some actors. In Figure 2, it is noted that actors 1,10 and 13 are closer in the posterior mean and mode representation then in the MLE and MKL representation, thus they have a smaller distance when these first two positional estimates are used. Focusing attention on the troublesome actors (1,10,13) and the ties between them, shows some differences in the graphs generated by each of the estimates. The MLE estimates produces graphs which have on average 1.25 ties between the three actors, the posterior mean graphs have on average 2.98 ties, the posterior mode graphs have an average 2.98 ties and finally the MKL graphs have on average of 1.48 ties. In the observed graph there is one tie between these three actors
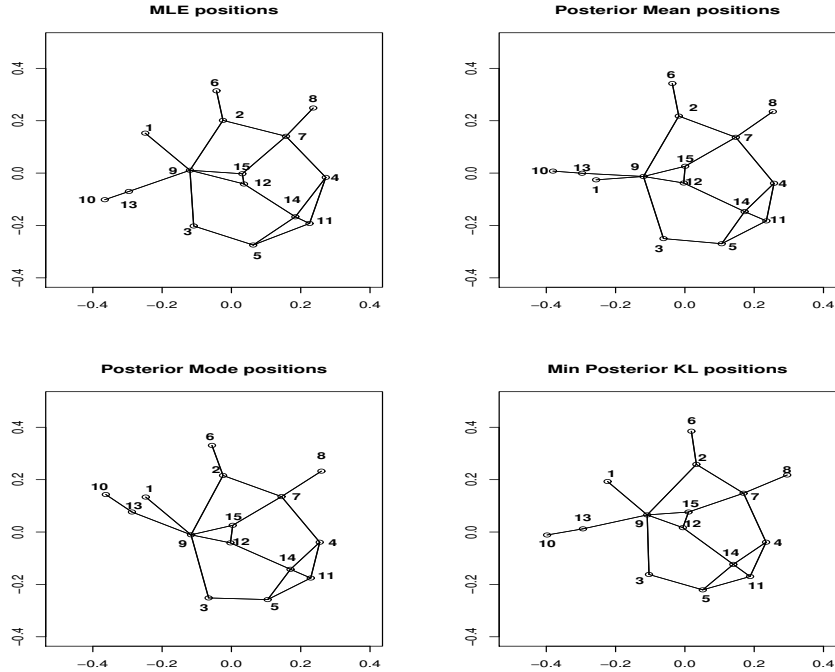
9

Figure 2: While the basic structure is similar it appear that there is some discrepancy in the location of actors 1,10,12

(1-13), so the MLE and MKL estimates produce graphs with ties between this trio most like the observed graph. For this analysis it appears that the MLE and MKL estimates are the best point estimates for the coefficients and positions of the actors.

5.2   Monk Data

The second analysis discussed is of another standard data set in the social network literature. In 1968, Sampson collected data on 18 monks and their interpersonal relations (Sampson, 1968). Each monk was asked about positive relations with the other monks and reciprocity was not required, thus the graph is directed. The data contain 56 directed ties between the 18 monks. There are no covariates in this data set, thus estimation is of β_0 and the positions of the actors. Similar to the Florentine analysis the chain samples 700 points from the posterior, allowing 10,000 iterations for burnin and 1,000 iteration between samples. In the Monk analysis the parameters for the chain were $\delta_{\beta_0} = 0.3$ and $\delta_Z = 0.2$. The prior distribution on β_0 was $N(0, 5^2)$ and the prior distribution on the positions coordinates, Z_{ij} is $N(0, 10^2)$ Figure 4 contains the summary plots of the Markov Chain. The chain values for the log-likelihood and β_0 indicate convergence. The marginal density of β_0 appears not to be skewed and to be unimodal. The marginal posterior for the positions appears to
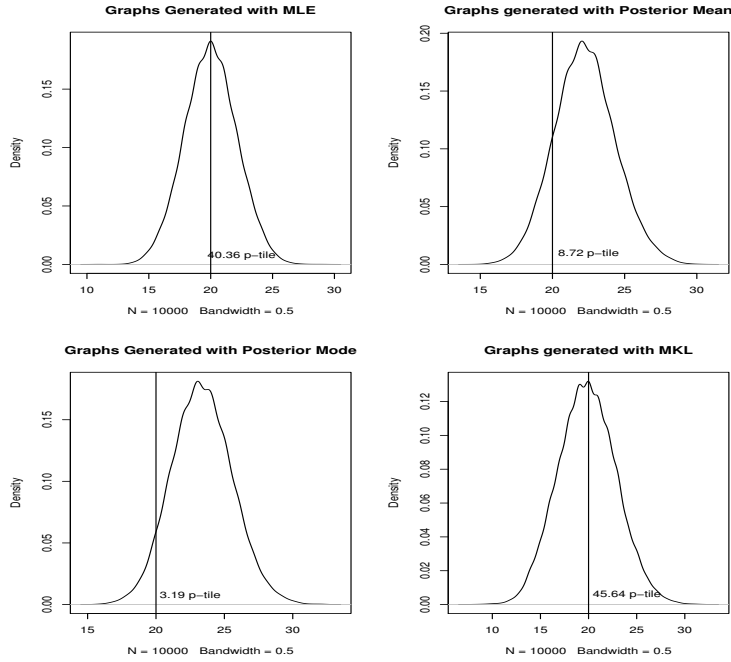
10

Figure 3: Density estimates for the number of ties in a graph generated by the latent space model with the specified estimates as parameters

show clustering of the position values, although some of the cluster appear to be elongated suggesting uncertainty in the locations.

The four types of positional estimates are give in Figure 5 with the same normalization technique used earlier for easier comparison. The differences between these estimates are more complex than in the Florentine analysis. Some of the actors seem to have stable positions regardless of the method used, actors 1,2,4,5,6,9,11 and 18. Actors 8 and 12, seem to have some uncertainty that may result in a bimodal distribution since they have collapsed onto one another in the posterior mean and mode estimates yet the MLE and MKL keep them close yet distinct. Actor 3 is closer to the center of the graph in the MLE positions, but in the other three it has moved up and taken actor 7 with it. Actors 13 and 14 also appear to be a pair that move together, in the MLE estimate they are at the right side of the graph in the center height wise yet in the other posterior estimates they are both up near the top of the graph. It appears that actor 10 travels with these actors into the center of the graph. Actor 17's position also shifts from the center top of the graph in the MLE to the upper right hand corner. These plots as well as the marginal posterior position plots seems to imply that the positional distribution is not very well behaved. Once again the MLE and MKL estimates would appear to be better estimates than the posterior mean and mode because of this complex behavior. This analysis produced MLE and MKL estimates
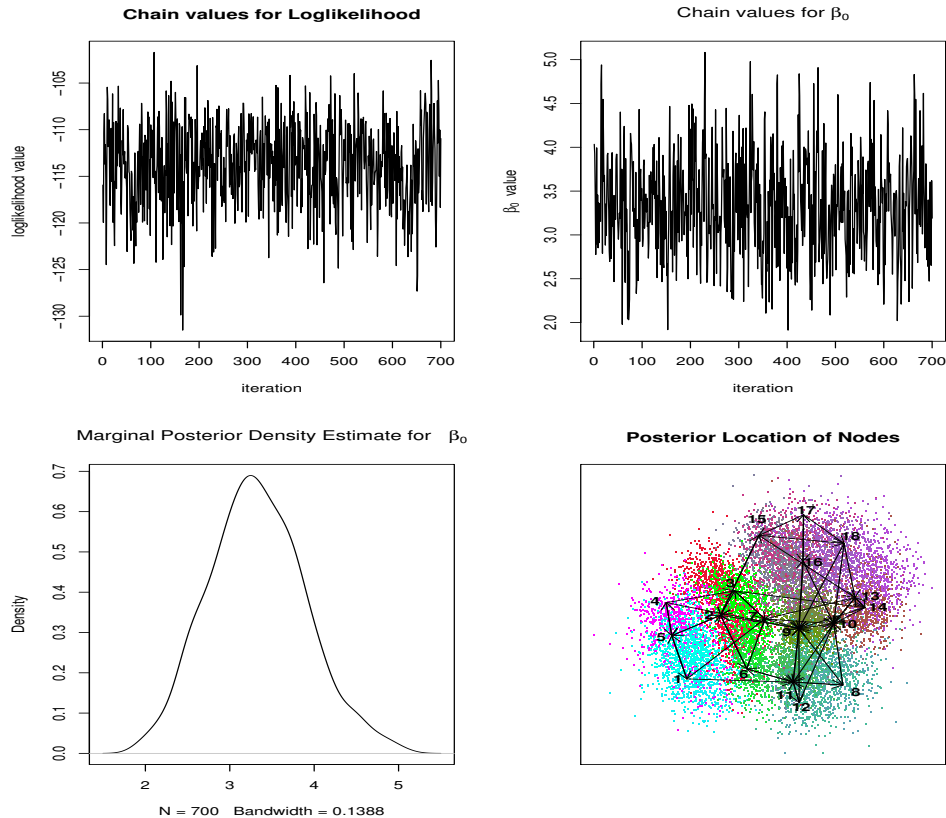
Figure 4: Labeled clockwise starting from upper left: (a) Chain values for the log-likelihood, (b) Chain values for β_0, (c) Marginal posterior for β_0, (d) Marginal posterior for positions

that are more similar the MLE estimate for β_0 is 2.38, with a graph with the square root of sum of squared distances of 15.77 while the MKL estimated the same quantities with 1.62 and 12.54.

The superiority of the MLE and MKL can be seen once again in these density plots for the number of ties in graphs generated by each of the estimates. The MLE and MKL seem to generate graphs which tend to have close to the observed value of 56 ties, while the posterior mean and mode estimates generate graphs with many more ties.

6 Discussion

This paper has provided a brief introduction to the latent space model for modeling network data as well as code which allows users to fit these models in the statistical freeware R. The code provided in the package `latentnet` is slightly different than the model originally presented by HRH in that all coefficients have a Gaussian prior, including the intercept. This
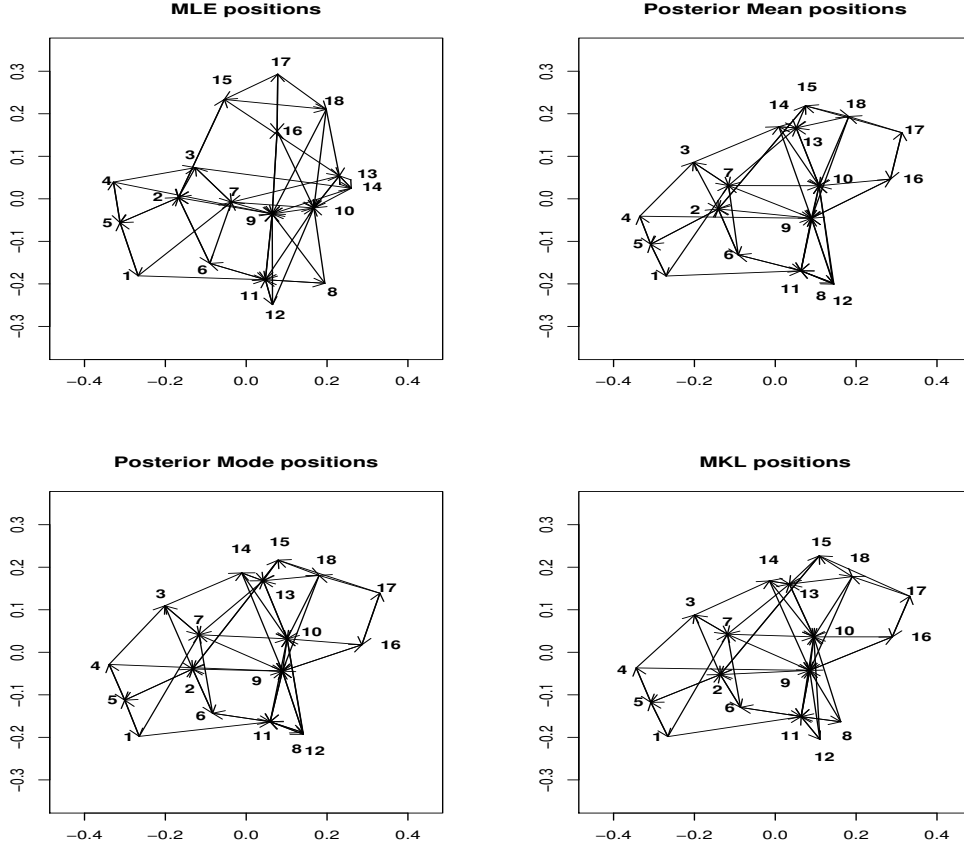
code also expands on the points estimates for the coefficients and positions discussed in HRH, by providing all four of the estimators described in Section 4.3: maximum likelihood estimate, posterior mean, posterior mode and the estimator which minimizes Kullback-Leibler divergence from the posterior.

Of the four explored here, the estimates that seem to produce graphs most consistent with the observed network are the maximum likelihood estimates and the minimum Kullback-Leibler divergence estimates. It was shown in Section 4.3 that by definition the MKL is designed to minimize the posterior mean of the KL divergence from the true model. The MLE of the distances can be shown to minimize the KL divergence from the data (Barndorff-Nielsen, 1978). It follows that the MLE of the positions are the values which minimized the KL divergence from the MLE distances. That is

$$\hat{\boldsymbol{\eta}}_{MLE} = argmax_{\boldsymbol{\eta}}\{P_{\boldsymbol{\eta}}(Y = y)\} = argmin_{\boldsymbol{\eta}}\{KL(\hat{\boldsymbol{\phi}}, \boldsymbol{\eta})\}$$

where $\hat{\boldsymbol{\phi}}$ is the MLE of the data. The MKL minimizes the KL divergence from the model with parameter given by the posterior expectation of the graph under the mean-value parameterization of the exponential family model (Handcock, 2003). Similarly the MLE minimizes
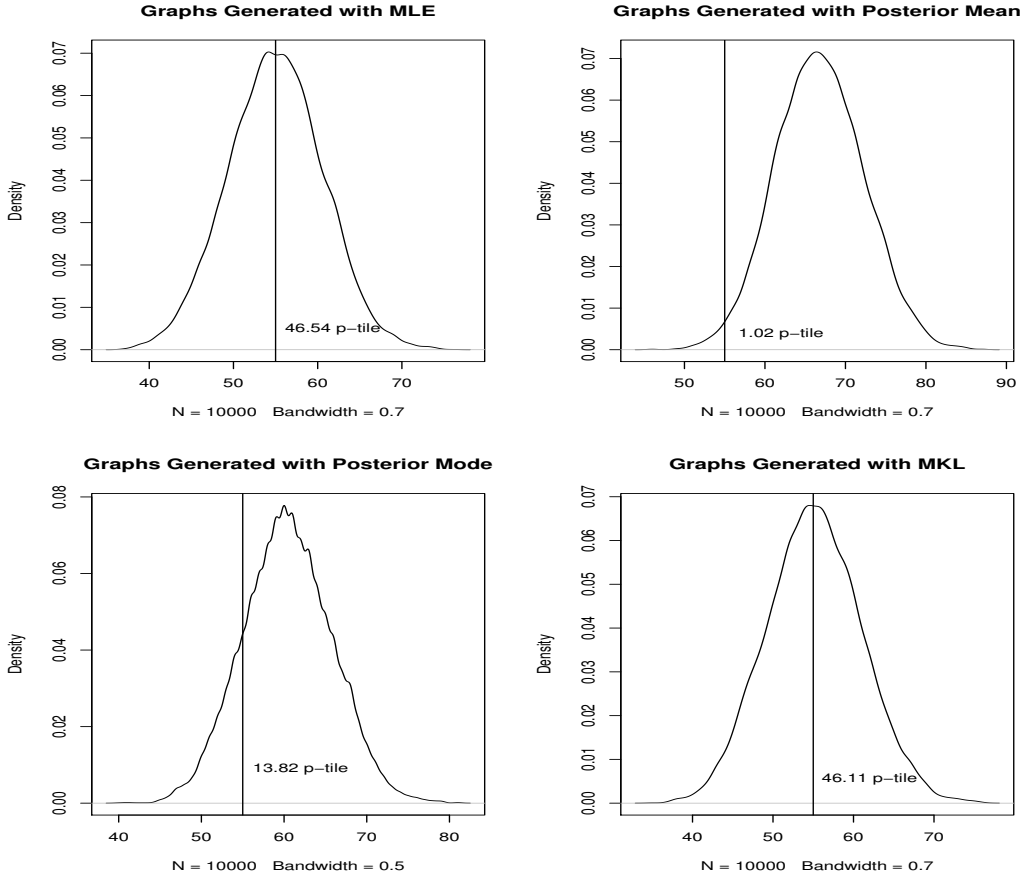
13

Figure 5: Density estimates for the number of ties in a graph generated by the latent space model with the specified estimates as parameters

the KL divergence from the model with mean-value parameter given by the observed graph. In this sense, the MKL minimizes the KL divergence from the posterior expectation of the graph, while the MLE minimizes the KL divergence from the observed graph. Heuristically, if the model is approximately correct, one expects that the posterior mean graph will be closer to the true parameter than the observed graph due to statistical averaging and use of prior information. Hence for most analyses the MKL estimate will be superior to the MLE. However both should be considered as they represent contrasting assumptions about the model (Handcock, 2003).

In network data, the purpose is to model the joint probability of all the ties in the network as well as the probability a tie between any two actors. While a graph has n actors and g ties, it is only one sample of the whole network. In many situations, the behavior of the MLE is understood under both large and finite sample sizes, but rarely can a sample of size one produce estimates which are reliable. Working with small sample sizes force more emphasis

to be placed on the model than the observations. Since there is limited information in the sample, the observed values are used more as a guide to update the assumed model. In both of the Padgett Florentine and Sampson Monk data sets the MLE and MKL estimates appear to behave similarly. As the size of the graph grows and ties are sparse it is reasonable to believe that the unrestrictiveness of the MLE could lead to instability in the estimates. While the MKL method may introduce bias into the estimates by putting more trust in the model, it may also produce estimates with less variability.

This paper proposes a new estimator and examines some of properties when applied to two data sets. More research is needed to truly understand the behavior of each of the point estimates (especially the variance) as well as to develop meaningful error bounds. It is unclear how the estimators will perform as the number of actors grows, particularly since the dimension of the parameter space grows with each additional actor. In order to understand the loss of flexibility and the gain in modeling ability of using the MKL estimates, it is necessary to understand the impact of the prior on the posterior distribution of the parameters.

References

Barndorff-Nielsen, O. E., 1978. Information and Exponential Families in Statistical Theory. New York: Wiley.

Faust, K., 1988. Comparison of methods for positional analysis: Structural and general equivalence. Social Networks 10, 313–341.

Frank, O., Strauss, D., 1986. Markov graphs. Journal of the American Statistical Association 81 (395), 832–842.

Gelman, A., Carlin, J. B., Stern, H. S., Rubin, D. B., 1995. Bayesian Data Analysis. London: Chapman and Hall.

Handcock, M. S., 2003. Working paper, Center for Statistics and the Social Sciences, University of Washington.

Hoff, P. D., Raftery, A. E., Handcock, M. S., December 2002. Latent space approaches to social network analysis. Journal of the American Statistical Association 97 (460), 1090–1098.

McFarland, D. D., Brown, D. J., 1973. Social distance as a metric: a systematic introduction to smallest space analysis. In: Laumann, E. O. (Ed.), Bonds of Pluralism: The Form and Substance of Urban Social Networks. New York: Wiley, pp. 213–253.

Padgett, J. F., Ansell, C. K., 1993. Robust action and the rise of the medici. The Amercan Journal of Sociology 98, 1259–1319.

Sampson, S. F., 1968. A novitiate in a period of change: An experimental and case study of relationships. Unpublished ph.d. dissertation, Department of Sociology, Cornell University.

Sibson, R., 1979. Studies in the robustness of multidimensional scaling: Perturbational analysis of classical scaling. Journal of the Royal Statistical Society, Series B, Methodological 41 (2), 217–229.

Appendix A

The `latentnet` package is designed to fit latent space models in one or more dimensions. The package is written in a combination of (the open-source statistical language) `R` and (ANSI standard) `C`, and runs in both Linux and Windows environments; it is called from the `R` command line.

R is a statistical computing package that runs on a variety of platforms, including both Windows and Linux based systems. `R` is a GNU-licensed freeware software package, considered by some to be simply an implementation of the well known statistical language S (albeit with some syntactic changes). `R` is widely available on the internet; installation of `R` is a fairly straightforward process and explained in great detail at the main `R` web resource-site, `www.r-project.org`, under the section "Download."

Installing `latentnet` and `graph` packages

Installation of the `latentnet` library (and the `graph` library that it requires) is likewise a straightforward matter. They can be installed over the web from within `R` and on both LINUX and Windows platforms. Both versions are identical.

```
> install.packages("latentnet",
      contriburl="http://www.csde.washington.edu/~handcock")
```

Once the package has been installed, it can be used by typing the call

```
> library(latentnet)
```

The functions are documented within `R`. To get help type in `R`:

```
> help(package="latentnet")
> help(latentnet)
```

This process should be repeated for the `graph` package.

16

Other useful functions in the package

Details and examples of these functions are given in the help pages of the functions.

Plotting of graphs: `plot.graph()`

`plot.graph()` plots directly to R graphics devices, and has an interactive feature. It is a variant of Carter Butts's routine in the `sna` package.

Printing and Summary of graphs and graph fits:

`latentnet` uses R generic methods so that calls to `summary()`, `anova` and `print()`, etc, lead to appropriate summaries and printing of the objects.

Model diagnostics: `ergm.mcmc.diagnostics()`

Model diagnostics are an important pert of most projects; without them, we will not be able to assess how well we are doing. We have implemented standard MCMC diagnostics and the `latentnet` fits can be directly run within the `CODA` package which is available for R as package coda. For more information see
http://www.mrc-bsu.cam.ac.uk/bugs/classic/coda04/readme.shtml

Using `latentnet`

The `ergm` function takes in arguments including a formula and several numeric specifications and returns a list with the model fit information. First, the user must enter a formula (according to the `ergm` requirements) containing the network to be fit as a `graph` object and the dyadic covariate information to be used. For more information on the `graph` object, see the `graph` package, which is required for use of `latentnet`. The dimension of the Euclidean representation of the latent space needs to be specified (default $k = 2$) and the number of covariates used in the model (default $p = 0$). The user can enter the desired sample size to be taken from the posterior distribution in the MCMC chain, the number of iterations allowed for burn-in, and the interval length between samples from the chain. The other arguments are the parameters for prior and proposal distributions over β and the position matrix.

The return object provides all of the information used in the analyses in Section 5. The chain information is returned to the user: burn-in length, thinning interval and posterior sample size in addition to the posterior samples. The posterior samples includes the coefficient values, the position matrices and the log-likelihood evaluated at the sample values. The coefficient values is returned in a n by q matrix, with n being the posterior sample size and $q = p + 1$, the number of covariates plus a intercept. The position values are returned

in a three dimensional array of size N by k by n, where N is the number of actors in the network. The posterior mean position matrix and the posterior mode positional estimates are also returned. The MLE estimates for both the coefficients, contained in a vector, and the positions, in a matrix, are supplied with the value of the log-likelihood at those values. The other three estimates of the actors positions discussed in this paper are also returned (maximum *a priori*, mean *a priori*, and minimum KL. scheme.

Example code for an analysis of the Florentine Marriage data is as follows:

```
#
# load the library
#
> library(latentnet)
#
# load the Florentine data
#
> data(gflo)
#
# Fit a latent space model in two-dimensions to the Florentine data
#
> flo.l2 <- ergm(gflo~latent(2,p=0,z.delta=0.3,b.delta=0.5,
                             b.prior.mu=0,b.prior.sd=10,
                             z.prior.mu=0,z.prior.sd=10),
               MCMCsamplesize=100, burnin=10000, interval=1000)
#
# Summarize the fit
#
> summary(flo.l2)
#
# Give an Analysis-of-variance decomposition
#
> anova(flo.l2)
#
# Plot the results
#
> plot(flo.l2)
```