Supplementary Information for 'Stochastic gradient descent-based inference for dynamic network models with attractors'

Hancong Pan
Department of Mathematics and Statistics, Boston University
and
Xiaojing Zhu
Department of Mathematics and Statistics, Boston University
and
Cantay Caliskan
Goergen Institute for Data Science, University of Rochester
and
Dino P. Christenson
Department of Political Science, Washington University in St. Louis
and
Konstantinos Spiliopoulos
Department of Mathematics and Statistics, Boston University
and
Dylan Walker
Argyros School of Business and Economics, Chapman University
and
Eric D. Kolaczyk

Department of Mathematics and Statistics, McGill University
E-mail: eric.kolaczyk@mcgill.ca

December 13, 2024

This supplementary document is organized as follows. In Section 1 we present the proofs of the theoretical results that appear in the main body of the paper. In Section 2 we present some additional statistical tests and diagnostics to back up the validity of the numerical results reported in the main body of the paper. In Section 3, we provide implementation details for the algorithm. In Section 4, we compare computational times among MCMC and the newly proposed SGD-based algorithm (in both CPU and GPU settings). We also comment upon the impact of hyper-parameters choices on convergence times. In Section 5 we present a simulation study to illustrate the algorithm's robustness when nodes exit the network at different turnaround levels.

1 Some Useful Results of Multivariate Normal Distribution

Our variance estimation method exploits the properties of the conditional distribution of a multivariate Gaussian distribution. Let x follow a multivariate normal distribution $x \sim \mathcal{N}(\mu, \Sigma)$, and hence the conditional distribution of a subset vector x_1, given its complement vector x_2, is also a multivariate normal distribution $x_1|x_2 \sim \mathcal{N}(\mu_{1|2}, \Sigma_{1|2})$ with the conditional mean and covariance given by $\mu_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$ and $\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$, respectively. The block-wise mean and covariance matrices are defined as $\mu = \left[\begin{smallmatrix}\mu_1 \\ \mu_2\end{smallmatrix}\right]$ and $\Sigma = \left[\begin{smallmatrix}\Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22}\end{smallmatrix}\right]$.

More specifically, the covariance of the conditional distribution $\text{Cov}(x_1|x_2)$ is a constant as a function of x_2, indicating that the shape of the conditional distribution is independent of the specific value of x_2. Additionally, writing $p(x_1|x_2)$ as the p.d.f of the conditional distribution of $x_1|x_2$, the mean $\mu_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$ is the median, and mode of the distribution. The maximum of the p.d.f. function $p(x_1 = \mu_{1|2}|x_2)$ is a constant as a function of x_2.

Therefore, we can recover the shape of the marginal distribution of x_2 from the joint distribution by varying x_2 and following the line $x_1 = \mu_{1|2}$.

Theorem 1.1. *Given the multivariate Gaussian distribution setting presented above, write $p(x_1, x_2)$ as the p.d.f of the joint distribution of x_1, x_2. $p(x_2)$ as the p.d.f of the marginal distribution of x_2. Then, we have $p(x_2) \propto p(x_1 = \mu_{1|2}, x_2)$.*

In particular, the marginal distribution is proportional to the joint distribution evaluated on the curve of the conditional mean of x_1 given x_2.

Proof. Let us explicitly state the conditional mean and covariance matrix for our given multivariate Gaussian distribution. The conditional mean of x_1 given x_2, denoted $\mu_{1|2}$, is:

$$\mu_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2).$$

Additionally, the conditional covariance matrix, denoted $\Sigma_{1|2}$, is described

as:

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}.$$

The joint distribution, $p(x_1, x_2)$, for the multivariate Gaussian is:

$$p(x_1, x_2) = \frac{1}{(2\pi)^{\frac{k}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

The conditional distributions $p(x_1|x_2)$ is:

$$p(x_1|x_2) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_{1|2}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x_1-\mu_{1|2})^T\Sigma_{1|2}^{-1}(x_1-\mu_{1|2})\right).$$

By the definition of conditional probabilities, we have:

$$p(x_2) = \frac{p(x_1, x_2)}{p(x_1|x_2)}.$$

Inserting $x_1 = \mu_{1|2}$ into this relation, we get:

$$p(x_2) = \frac{p(\mu_{1|2}, x_2)}{p(\mu_{1|2}|x_2)}.$$

Since $p(\mu_{1|2}|x_2) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_{1|2}|^{\frac{1}{2}}}$ and $\Sigma_{1|2}$ is constant as a function of x_2, we get that

$$p(x_2) = Cp(\mu_{1|2}, x_2).$$

where $C = (2\pi)^{\frac{n}{2}}|\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}|^{\frac{1}{2}}$, a constant as a function of x_2. □

Theorem 1.1 provides an expression for the shape of the marginal distribu-

tion from the joint distribution, when the joint distribution is a multivariate normal distribution.

Next, we proceed with an important result demonstrating that for a normal distribution the ideas of Theorem 1.1 yield a formula for the variance of x_2 that we can then be turned into a practical SGD-based algorithm for uncertainty quantification. In particular, recall that in a univariate normal distribution, two key parameters - the mean and variance - describe the distribution fully. By considering two distinct points from the distribution, we can deduce these parameters.

Corollary 1.1.1. *Consider the multivariate Gaussian distribution setting presented above, and let x_2 be scalar. Then, for some fixed $\tilde{x}_2 \neq \mu_2$,*

$$Var(x_2) = \frac{1}{2} \frac{(\mu_2 - \tilde{x}_2)^2}{\log p(x_1 = \mu_1, x_2 = \mu_2) - \log p(x_1 = \mu_{1|2}, x_2 = \tilde{x}_2)} \tag{1}$$

Proof. Recall that the p.d.f of a univariate Gaussian distribution $x_2 \sim \mathcal{N}(\mu_2, \mathrm{Var}(x_2))$ is given by:

$$p(x_2) = \frac{1}{\sqrt{2\pi \mathrm{Var}(x_2)}} \exp\left(-\frac{(x_2 - \mu_2)^2}{2\mathrm{Var}(x_2)} \right). \tag{2}$$

Taking the logarithm, we obtain:

$$\log p(x_2) = -\frac{1}{2} \log(2\pi \mathrm{Var}(x_2)) - \frac{(x_2 - \mu_2)^2}{2\mathrm{Var}(x_2)}. \tag{3}$$

From Theorem 1.1, we deduce that $\log p(x_2)$ and $\log p(x_1 = \mu_{1|2}, x_2)$ differ by a constant since $p(x_2) \propto p(x_1 = \mu_{1|2}, x_2)$. Considering two distinct points,

$x_2 = \mu_2$ and $x_2 = \tilde{x}_2$:

For $x_2 = \mu_2$:

$$\log p(x_2 = \mu_2) = -\frac{1}{2}\log(2\pi \mathrm{Var}(x_2)). \tag{4}$$

For $x_2 = \tilde{x}_2$:

$$\log p(x_2 = \tilde{x}_2) = -\frac{1}{2}\log(2\pi \mathrm{Var}(x_2)) - \frac{(\tilde{x}_2 - \mu_2)^2}{2\mathrm{Var}(x_2)}. \tag{5}$$

Define the difference in these logarithmic probabilities as:

$$\Delta = \log p(x_1 = \mu_1, x_2 = \mu_2) - \log p(x_1 = \mu_{1|2}, x_2 = \tilde{x}_2). \tag{6}$$

Using the above, we get:

$$\Delta = \frac{(\mu_2 - \tilde{x}_2)^2}{2\mathrm{Var}(x_2)}. \tag{7}$$

Rearranging gives:

$$\mathrm{Var}(x_2) = \frac{(\mu_2 - \tilde{x}_2)^2}{2(\log p(x_1 = \mu_1, x_2 = \mu_2) - \log p(x_1 = \mu_{1|2}, x_2 = \tilde{x}_2))}, \tag{8}$$

completing the proof of the corollary. □

Beyond just the variance represented by the diagonal elements of the covariance matrix, there's also a formula for the off-diagonal elements. This

allows us to reconstruct a single column of the covariance matrix.

Corollary 1.1.2. *Consider the multivariate Gaussian distribution setting presented above. Then, for some fixed* $\tilde{x}_2 \neq \mu_2$,

$$\Sigma_{12} = \frac{\Sigma_{22}}{(\tilde{x}_2 - \mu_2)}(\mu_{1|2} - \mu_1),$$

where $\mu_{1|2}$ *denotes the conditional mean of* x_1 *given* $x_2 = \tilde{x}_2$.

The methodology established for a generic normal distribution $p(x_1, x_2)$ is adapted to estimate variance in the posterior distribution $\pi(\cdot|Y)$, effectively applying the same theoretical concepts to the Bayesian analysis under a Laplace approximation.

2 Supplementary Results in X Data Analysis

2.1 BIC values

In our analysis, we utilized the Bayesian Information Criterion (BIC) to determine the best change-point among the competing models. For the X platform dataset, we evaluated the BIC values for potential single change-points from 2012 onwards, as presented in Table below. The model suggesting a change-point in 2012 yielded the lowest BIC value.

	2012	2013	2014	2015	2016	2017	2018	2019
BIC	**220138**	220163	220201	220189	220212	220203	220234	220224

Table 1: BIC values for competing models with different change-point for the X platform data.

2.2 Diagnostics

Note that the variance estimation algorithm is similar to a quadratic approximation to the log-posterior density. We performed diagnostics to evaluate the assumption that the log likelihood function can be approximated by a quadratic function in a small neighborhood centered around the mode.

2.2.1 Normality test

We rewrite equation (1) as follows:

$$\max_{\theta_1=\theta_1^*-\eta} \log \pi(Z,\theta|Y) = \max_{\theta_1=\theta_1^*} \log \pi(Z,\theta|Y) - \frac{1}{2}\frac{\eta^2}{\widehat{\mathrm{Var}(\theta_1|Y)}}, \tag{9}$$

and if our assumptions are correct, then $\widehat{\mathrm{Var}(\theta_1|Y)}$ should be a constant no matter the choice of η. We vary η to obtain a plot of $\max_{\theta_1=\theta_1^*-\eta} \log \pi(Z,\theta|Y)$ as a function of η to test whether this assumption is valid. If this assumption holds, $\max_{\theta_1=\theta_1^*-\eta} \log \pi(Z,\theta|Y)$ should be approximately a quadratic function as a function of η.

We choose θ_1 to be γ_R^w, and set η to be -0.03, -0.02, -0.01, 0.01, 0.02, 0.03. In Figure 1 we plot $\max_{\theta_1=\theta_1^*-\eta} \log \pi(Z,\theta|Y)$ as a function of η (left) and η^2 (right). We can see that $\max_{\theta_1=\theta_1^*-\eta} \log \pi(Z,\theta|Y)$ is approximately quadratic

as a function of η and linear as a function of η^2. The diagnostic plot does not show evidence of violating the underlying assumptions of the variance estimation method.

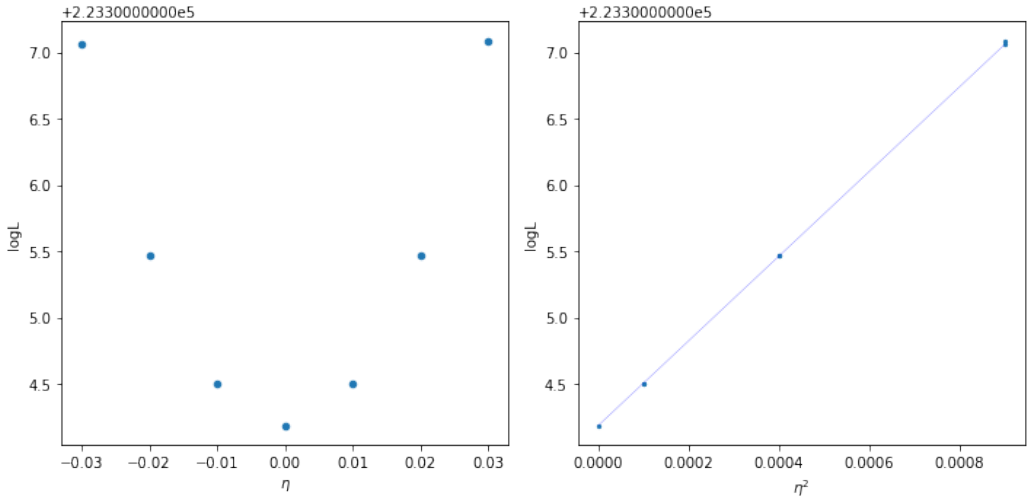

Figure 1: We choose θ_1 to be γ_R^w, and set η to be -0.03, -0.02, -0.01, 0.01, 0.02, 0.03. We plot $\max_{\theta_1=\theta_1^*-\eta} \log \pi(Z, \theta|Y)$ as a function of η (left) and η^2 (right).

2.2.2 Linearity test

From Theorem 1.1, we see that the conditional mean of $X_{2:n}|X_1$ is linear as a function of X_1 under the Gaussian assumption, i.e.,

$$\mu(X_{2:n}|X_1 = \mu_1 + \eta) = \mu_{2:n} + \Sigma_{21}\Sigma_{11}^{-1}\eta. \tag{10}$$

We vary η to obtain the slope $\frac{\mu(X_j|X_1=\mu_1+\eta)-\mu_j}{\eta}$ for each X_j with different η to test whether this assumption is valid. If this assumption holds,

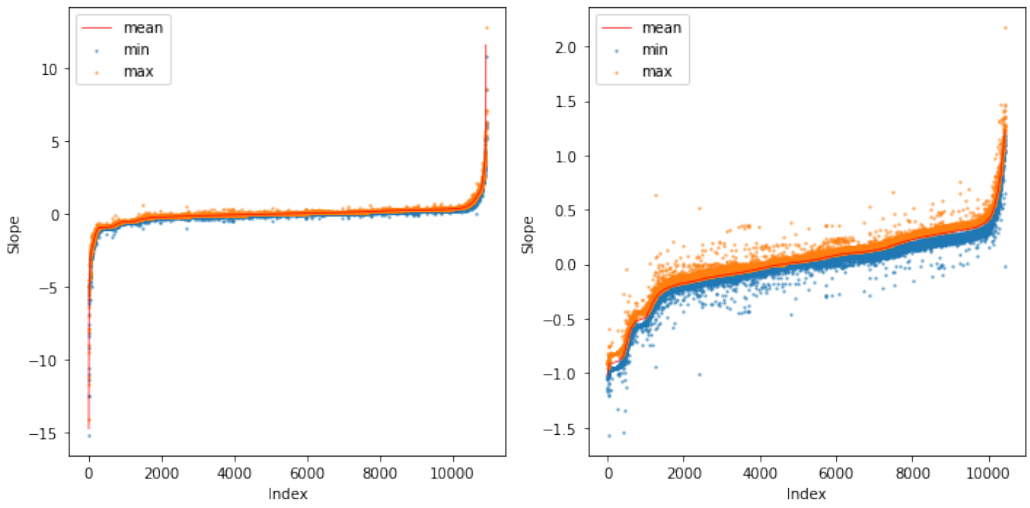

Figure 2: We choose θ_1 to be γ_R^w, and set η to be -0.03, -0.02, -0.01, 0.01, 0.02, 0.03. We calculate $\frac{\mu(X_j|X_1=\mu_1+\eta)-\mu_j}{\eta}$ for each latent position parameters under each η. The latent position parameters are ordered by the mean of its slopes and plot the mean maximum and minimum of its slopes (left) and the the slopes of 95% of the latent position parameters after removing the tail 5% that have a very large or very small slopes.

$\frac{\mu(X_j|X_1=\mu_1+\eta)-\mu_j}{\eta}$ should be a constant regardless the choice of η for each X_j.

We use the results of the previous test, set θ_1 to be γ_R^w, and η to take values in the set $\{-0.03, -0.02, -0.01, 0.01, 0.02, 0.03\}$. We calculate $\frac{\mu(X_j|X_1=\mu_1+\eta)-\mu_j}{\eta}$ for each latent position parameters and each η. In Figure 2 we order the latent position parameters by the mean of its slopes and plot the mean maximum and minimum of its slopes (left) and the the slopes of 95% of the latent position parameters after removing the tail 5% that have a very large or very small slopes. We can see that the slopes, or the correlated changes for each latent position parameters when we change γ_R^w are close under different choice of η. Therefore the diagnostic plot does not show substantial evidence

of violating the underlying assumptions of the variance estimation method.

3 Implementation Details

We use gradient descent with momentum and choose different learning hyperparameters for the latent position parameters and the global parameters. With gradient descent, we use all the terms in the log posterior function instead of taking a sampled posterior function. We stop running gradient descent when the parameter updates drop below a threshold of $\epsilon = 10^{-4}$, a condition we check every 100 gradient steps. Priors for α and δ were chosen to be $\mathcal{N}(0, 100)$ to keep it flat and uninformative. We chose the priors for γ^w and γ^b to be $\mathcal{N}(0.5, 100)$ and $\mathcal{N}(-0.5, 100)$ to reflect the prior belief of polarization, however these are also quite uninformative given the large variance. We fix τ^2 at 10, σ^2 at 1 and ϕ^2 at 10. We choose $p = 2$, following Zhu et al. (2023). The choice of two dimensions aligns with DW-NOMINATE, a widely recognized model of congressional ideology. This model demonstrates that two dimensions can account for up to 90% of the variation in roll call voting (McCarty et al., 2016).

For the step size λ, we started by conducting a search to identify the optimal initial step size for our model by observing the loss curves for various candidates. The goal was to identify a step size that avoided slow convergence and prevented the loss from exploding. Once the best initial step size was determined, we proceeded with training the model. During training, if the

loss plateaued, we halved the step size to promote further reduction in the loss. After one such adjustment, further halving did not yield significant improvements, so we concluded the training process at that point. To choose η_α, we take α as an example. We start by selecting an initial guess for η_α such that $\log \pi(\alpha^*) - \log \pi(\alpha^* + \eta_\alpha)$ is neither too large nor too small, aiming for a value between 10 and 50. Here, $\log \pi(\alpha^*)$ denotes the log probability density function π evaluated at the mode with α set to be α^*, while $\log \pi(\alpha^* + \eta_\alpha)$ represents the same function evaluated at the mode with α set to be $\alpha^* + \eta_\alpha$. This range ensures that the perturbation is sizable enough to influence the parameters meaningfully but not so large that they deviate excessively from the optimum. In our case, we opted for a uniform η_α for all parameters for which we wanted to estimate variance, and this approach worked well.

4 Algorithm Comparison and Analysis

Figure 3 illustrates the convergence times for SGD-based algorithms executed on a CPU and an Nvidia GPU A40, alongside the convergence times for the MCMC algorithm. The x-axis denotes the number of nodes in the network, while the y-axis represents the convergence time in minutes. The blue curve corresponds to the convergence times of the SGD algorithm when executed on a CPU, the red curve represents the convergence times of the same algorithm when executed on a GPU, and the green curve shows the convergence times for the MCMC algorithm. The SGD algorithm provides approximately a 30-

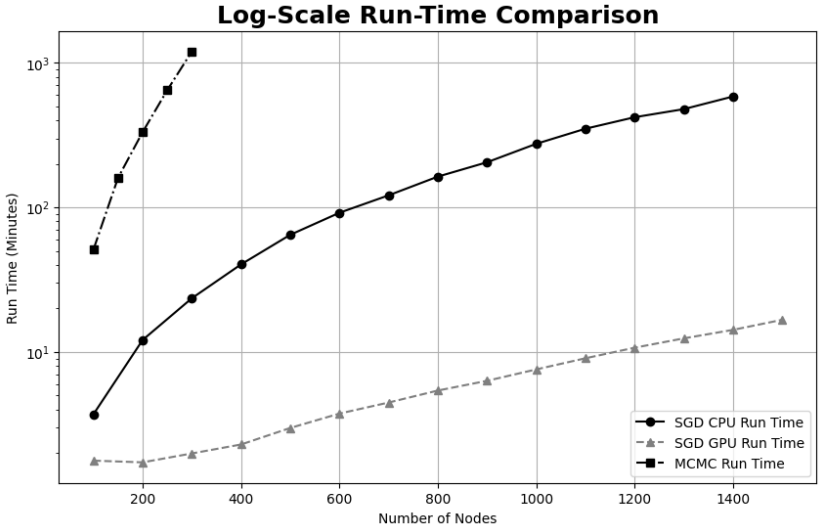

Figure 3: Comparison of Convergence Times for SGD (CPU and GPU) and MCMC Algorithms Across Different Network Sizes.

fold reduction in convergence time compared to the MCMC algorithm. Additionally, parallelization further improves the scalability of the SGD method: executing the SGD algorithm on the Nvidia GPU A40 yields substantial efficiency gains, resulting in around a 20-fold reduction in convergence time compared to its execution on the CPU. Note that the reported convergence times for the SGD algorithm include both the point estimation step and the variance estimation step.

4.1 Impact of Hyperparameters on Convergence Time

It is important to understand how hyperparameters, such as step size (λ), affect the convergence speed. Choosing the correct step size (λ) and momentum in SGD is crucial, as it significantly affects the convergence behavior.

Specifically, if the step size is too large, the algorithm may diverge, causing the loss to explode. If the step size is too small, the algorithm will converge very slowly. To address this, we fixed the momentum parameter at 0.99 and focused on tuning the step size. For the step size λ, we started by conducting an empirical search to identify an optimal initial value. We observed the loss curves for various step size candidates, aiming to find a step size that avoids both slow convergence and loss explosion. After determining the best initial step size, we began training the model. During training, if we noticed that the loss plateaued, we halved the step size. This halving process was repeated until the log posterior function convergences. Once the correct magnitude is established, fine-tuning the step size only results in marginal gains in convergence speed. For the computational time plot presented above, we used the step size of 0.001 for all latent position parameters and 0.001 for all global parameters, while using the sign of the gradient (+1/-1) for updating the global parameters instead of the actual gradient values.

4.2 Computational Complexity and Factors Contributing to Speedup

Each iteration of the proposed SGD-based algorithm has a computational complexity of $O(T \times N^2)$ for both time and space, where T is the number of time points in the network time-series and N represents the number of nodes in each network. This complexity comes from the requirement to

process all node pairs across each time point. Each iteration of the MCMC algorithm also has the same computational complexity of $O(T \times N^2)$. The significant speedup of the SGD-based algorithm predominantly stems from two key factors:

1. **Parallelization on GPU**: The ability to leverage the parallel processing capabilities of GPUs significantly reduces the time spent per iteration. This factor provides around a 20-fold increase in performance compared to the CPU execution.

2. **Fewer Iterations to Converge**: The SGD-based algorithm requires far fewer iterations to achieve convergence compared to the MCMC algorithm. This results in approximately a 30-fold increase in performance.

5 Varying percentage leaving

	$\alpha = 1$		$\delta = 2$		$\gamma^w = 0.25$		$\gamma^b = 0.5$	
	$\hat{\alpha}$	$\widehat{\text{Var}(\alpha)}$	$\hat{\delta}$	$\widehat{\text{Var}(\delta)}$	$\hat{\gamma^w}$	$\widehat{\text{Var}(\gamma^w)}$	$\hat{\gamma^b}$	$\widehat{\text{Var}(\gamma^b)}$
Turnover=0%	0.94 (0.004)	0.005 (<0.001)	1.98 (0.005)	0.005 (<0.001)	0.24 (0.025)	0.028 (0.007)	0.50 (0.025)	0.026 (0.008)
Turnover=10%	0.94 (0.006)	0.005 (<0.001)	1.98 (0.007)	0.005 (<0.001)	0.24 (0.032)	0.030 (0.007)	0.50 (0.030)	0.027 (0.008)
Turnover=20%	0.94 (0.006)	0.005 (<0.001)	1.98 (0.007)	0.006 (<0.001)	0.25 (0.041)	0.032 (0.007)	0.49 (0.034)	0.028 (0.008)
Turnover=30%	0.94 (0.008)	0.005 (<0.001)	1.98 (0.005)	0.007 (<0.001)	0.24 (0.044)	0.034 (0.008)	0.50 (0.034)	0.030 (0.010)
Turnover=40%	0.94 (0.007)	0.005 (<0.001)	1.98 (0.008)	0.008 (<0.001)	0.26 (0.042)	0.036 (0.008)	0.48 (0.029)	0.030 (0.009)
Turnover=50%	0.94 (0.008)	0.005 (<0.001)	1.98 (0.008)	0.009 (<0.001)	0.25 (0.042)	0.038 (0.008)	0.49 (0.031)	0.032 (0.010)
Turnover=60%	0.94 (0.007)	0.005 (<0.001)	1.98 (0.010)	0.011 (<0.001)	0.27 (0.036)	0.039 (0.008)	0.48 (0.027)	0.031 (0.009)
Turnover=70%	0.94 (0.007)	0.005 (<0.001)	1.98 (0.014)	0.015 (<0.001)	0.26 (0.030)	0.043 (0.008)	0.48 (0.025)	0.033 (0.011)
Turnover=80%	0.95 (0.005)	0.005 (<0.001)	1.99 (0.024)	0.022 (<0.001)	0.26 (0.056)	0.049 (0.008)	0.49 (0.038)	0.034 (0.012)

Table 2: Posterior-based mean (empirical standard deviation) for point estimation and variance estimation for parameters α, δ, γ^w, and γ^b, based on $n = 500$ nodes with $T = 10$ time points. The table presents the robustness of the estimation method across varying proportions of node turnover, ranging from 0% to 80%. Despite increasing turnover, the point estimates remain consistent. The variance of the estimates for δ, γ^w, and γ^b increases due to the reduction in sample size, which is also captured by the variance estimates appropriately.

Table 2 presents the posterior-based mean (empirical standard deviation) for point estimation and variance estimation, based on a CLSNA model with $n = 500$ nodes, similar to the size of a US Congress, across varying proportions of nodes entering and leaving the network to demonstrate the robustness of the method. Similar to Congress, at each time step, a random subset of nodes are dropped out, while an equal number of new nodes are randomly initialized according to the model's prior distribution. The results indicate that both the point estimates and variance estimates are robust across different turnover levels. As the turnover proportion increases, the variance of the estimates for δ, γ^w, and γ^b also increases, which can be attributed to the smaller sample sizes available for estimating these parameters. This shows

the method's capability to provide robust point estimates with varying proportions of nodes entering and leaving while effectively capturing the increase in uncertainty as the turnover proportion rises.

References

McCarty, N., Poole, K. T. and Rosenthal, H. (2016) *Polarized America: The dance of ideology and unequal riches*. mit Press.

Zhu, X., Caliskan, C., Christenson, D. P., Spiliopoulos, K., Walker, D. and Kolaczyk, E. D. (2023) Disentangling positive and negative partisanship in social media interactions using a coevolving latent space network with attractors model. *Journal of the Royal Statistical Society Series A: Statistics in Society*. URL: `https://doi.org/10.1093/jrsssa/qnad008`. Qnad008.