The Journal of Open Source Software

NetworkChange: Analyzing Network Changes in R

Jong Hee Park[1] **and Yunkyu Sohn**[2]

1 Department of Political Science and International Relations, Seoul National University **2** Department of Sociology, Seoul National University

DOI: 10.21105/joss.02708

Software

- Review
- Repository
- Archive

Editor: Arfon Smith

Reviewers:

- @akbaritabar
- @marcjwilliams1
- @martinmodrak

Submitted: 09 July 2020
Published: 03 February 2022

Summary

NetworkChange is an R package that detects multiple structural changes in longitudinal network data using the latent space approach (Peter D. Hoff et al., 2002). Based on the Bayesian multi-array representation of longitudinal networks (Peter D. Hoff, 2015), **NetworkChange** performs Bayesian hidden Markov analysis to discover changes in structural network features across temporal layers using the hidden Markov model formulation. **NetworkChange** can detect various forms of changes in network structure such as block-splitting, block-merging, and core-periphery changes. **NetworkChange** also provides functions for model diagnostics using WAIC, average loss, and log marginal likelihoods as well as visualization tools for dynamic analysis results of longitudinal networks.

Statement of Need

The package is designed for R users who need to analyze longitudinal network data to discover latent node-level characteristics including cases when there are discrete changes of the underlying states governing the node-level characteristics. We present a hidden Markov network change-point model (HNC) based on the multilinear tensor regression model (Peter D. Hoff, 2015). Our approach formalizes structural changes of networks as shifts in discrete states associated with specific sets of network generating parameters. This is in contrast to an R package for latent space and cluster analysis of networks (Krivitsky & Handcock, 2008) which does not incorporate a state space model (e.g. hidden Markov model) and a Python code for longitudinal network analysis (Peel & Clauset, 2015) under a distinct formulation (hierarchical random graph model) with a changepoint detection function. In addition to functions for the statistical analysis, **NetworkChange** provides visualization functions for summary of the analysis results. Please refer to Figure 1) for the organization of the package functions and their usage. The complete guide for using core functions of the package is presented at https://cran.r-project.org/web/packages/NetworkChange/vignettes/NetworkChange.html as its vignette with a synthetic data set and an empirical data set analysis example. Park & Sohn (2020) provide methodological details of the algorithms implemented in the package.

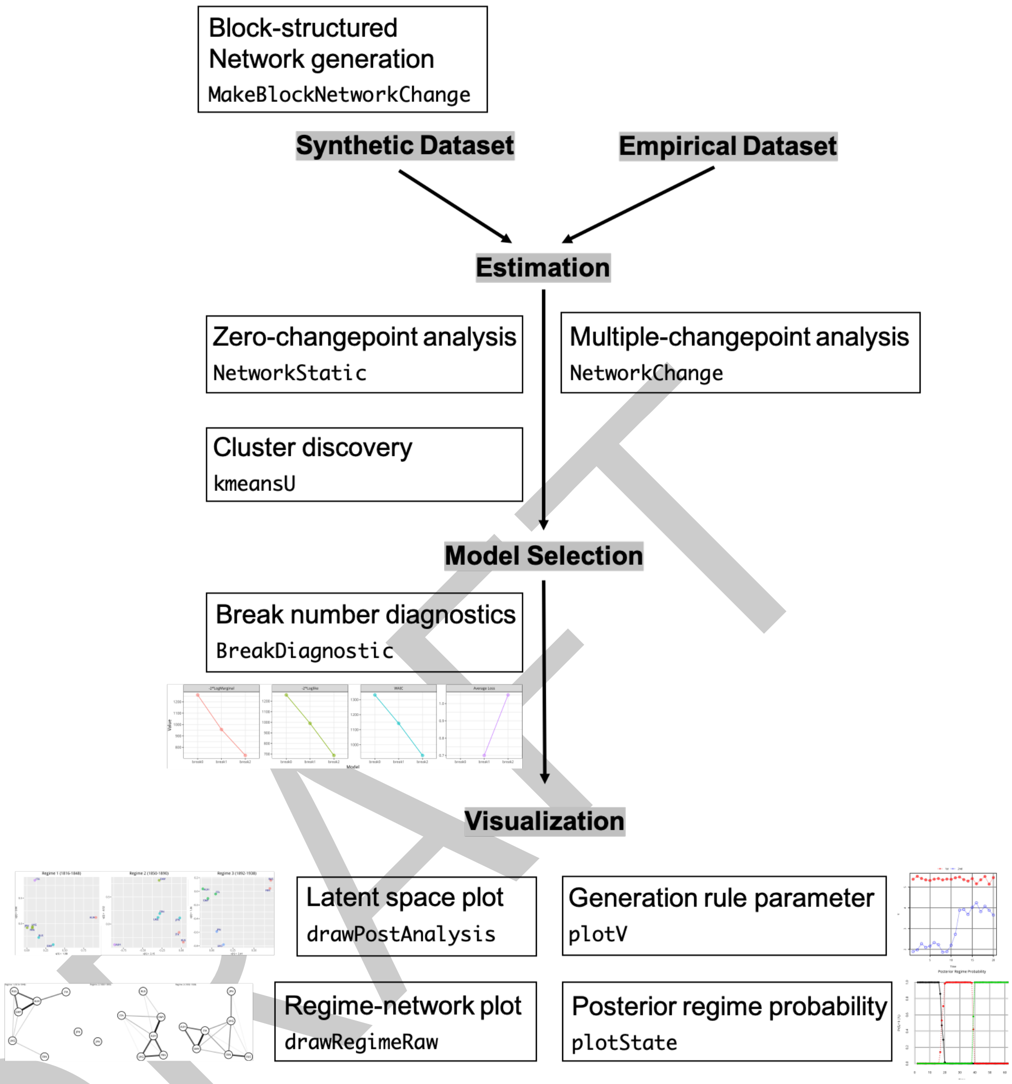

Figure 1: Summary of selected features and functions of the package.

Empirical Data Analysis Example

In this section, we analyze changes in the international military alliance network among major powers. The data set is originally from (Gibler, 2008) and users can call this data set by `data(MajorAlly)`.

Our goal in this section is to detect structural changes in the longitudinal alliance network among major powers using HNC. We follow the COW dataset's coding of "major powers" (the United Kingdom, Germany, Austria-Hungary, France, Italy, Russia, the United States, Japan, and China) in the analysis. China and the United States were dropped as they did not have an alliance relationship with the analyzed countries during the selected period. We aggregated every 2 year network from the original annual binary networks to increase the density of each layer.

```
library(NetworkChange)
data(MajorAlly)
Y <- MajorAlly
```

```
time <- dim(Y)[3]
drop.state <- c(which(colnames(Y) == "USA"), which(colnames(Y) == "CHN"))
newY <- Y[-drop.state, -drop.state, 1:62]
```

First, we fit a pilot model to elicit reasonable inverse gamma prior values for $\mathbf{v}_t$ (v_0 and v_1).

```
G <- 100
set.seed(1990)
test.run <- NetworkStatic(newY, R=2, mcmc=G, burnin=G, verbose=0,
                          v0=10, v1=time*2)
V <- attr(test.run, "V")
sigma.mu = abs(mean(apply(V, 2, mean)))
sigma.var = 10*mean(apply(V, 2, var))
v0 <- 4 + 2 * (sigma.mu^2/sigma.var)
v1 <- 2 * sigma.mu * (v0/2 - 1)
```

Then, we diagnose the break number by comparing model-fits of several models with a varying number of breaks.

```
set.seed(11223);
detect2 <- BreakDiagnostic(newY, R=2, break.upper=2,
                           mcmc=G, burnin=G, verbose=0,
                           v0=v0, v1=v1)
detect2[[1]]
```

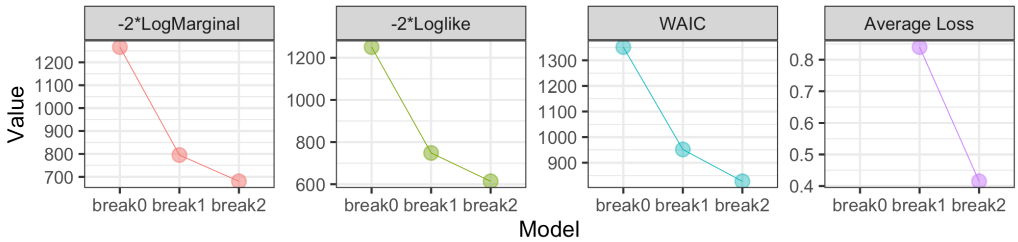

Figure 2: Break number detection.

The test results from WAIC, log marginal likelihood, and average loss indicate that HNC with two breaks is most reasonable.

Based on the test result, we fit the HNC with two breaks to the major power alliance network and save the result in *R* object `fit`.

```
G <- 100
K <- dim(newY)
m <- 2
initial.s <- sort(rep(1:(m+1), length=K[[3]]))
set.seed(11223);
fit <- NetworkChange(newY, R=2, m=m, mcmc=G, initial.s = initial.s,
                     burnin=G, verbose=0, v0=v0, v1=v1)
```

First, we can examine transitions of hidden regimes by looking at posterior state probabilities ($p(\mathbf{S}|\mathcal{Y}, \Theta)$) over time. `plotState()` in **MCMCpack** pacakge provides a function to draw the posterior state probabilities from changepoint analysis results. Since our input data is an array, we need to change the input data as a vector.

```r
attr(fit, "y") <- 1:K[[3]]
plotState(fit, start=1)
```

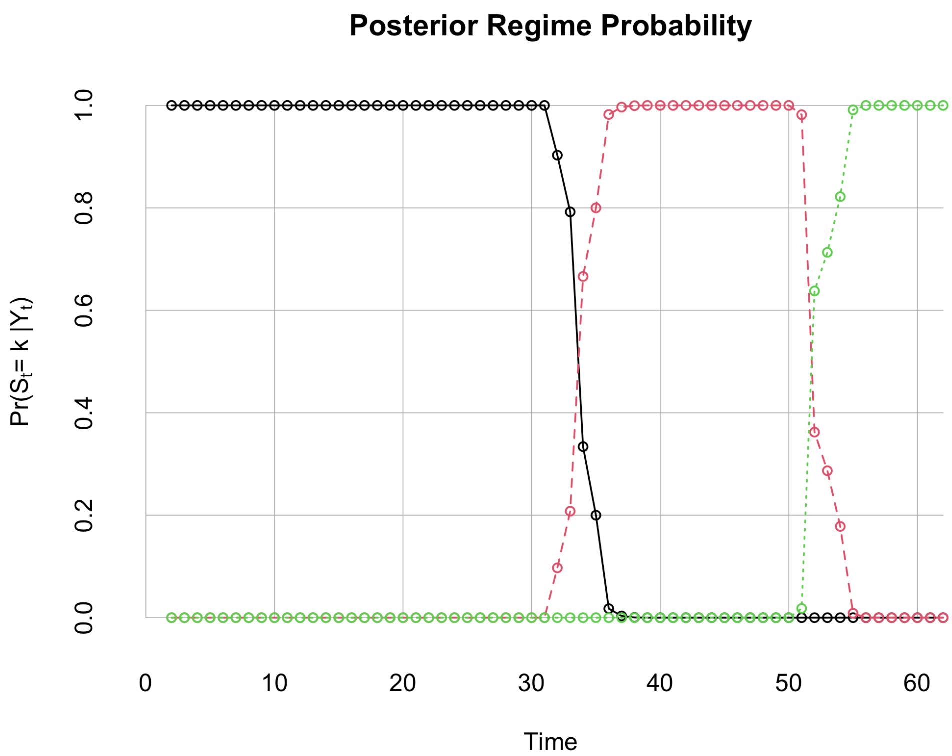

Figure 3: Regime probability.

Next, we draw regime-specific latent node positions of major powers using `drawPostAnalysis`. Users can choose the number of clusters in each regime by `n.cluster`.

```r
p.list <- drawPostAnalysis(fit, newY, n.cluster=c(4, 4, 3))
multiplot(plotlist = p.list, cols=3)
```

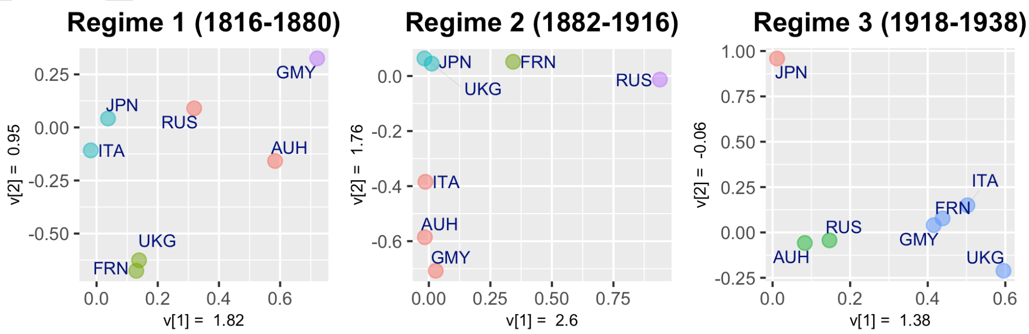

Figure 4: Regime-specific latent node positions.

Then, using `drawRegimeRaw()`, we can visualize original network connections for each regime by collapsing network data within each regime.

```
drawRegimeRaw(fit, newY)
```

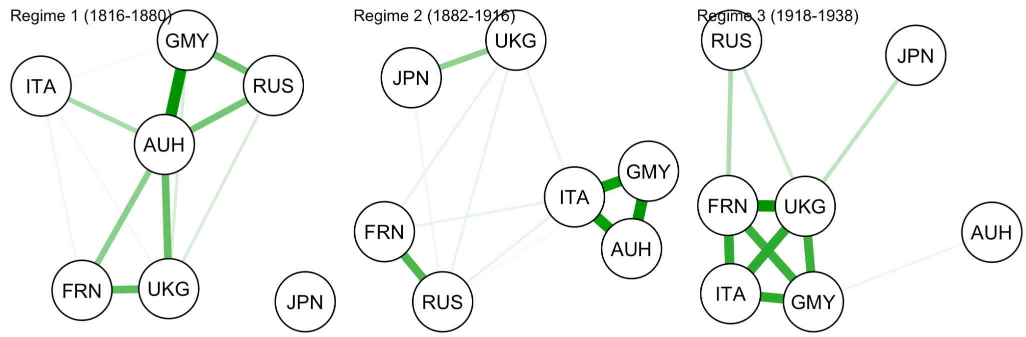

Figure 5: Regime-specific networks.

Identifying hidden regimes of the military alliance network makes it clear the central role of Austria-Hungary during the first two regimes in the military alliance network among major powers.

Potential Presence of Multi-modal Posterior Density

It is not surprising that real-world data generating processes are more likely to be the result of multiple heterogeneous factors that produce a multi-modal posterior landscape in both temporal and state-specific group structural dimensions.

While it is possible to implement conventional MCMC mixing diagnostics to verify the quality of mixing (e.g. diagnostic options included in the **posterior** package (Bürkner et al., 2020)), we should note that a single MCMC chain may not be sufficient to recover the multimodal posterior density of real world longitudinal networks formed by nontrivial data generating processes. Running multiple MCMC chains can be helpful to recover the multi-modal landscape of the high dimensional parameter space.[1]

Using Network Data in Other Formats

The input network data type for **NetworkChange** is a full three-dimensional array with dimensions N by N by T, where N is the number of nodes and T is the number of observed time points, and each slice represents an adjacency matrix for a specific time point. Most other network packages include a function for converting an edge list or other sparse representation of network data into an adjacency matrix format. For example, the **igraph** package includes the `as.matrix.igraph` function, and the **sna** package includes the `as.sociomatrix.sna` function.

The proposed method provides estimation of a one mode (i.e. unipartite) symmetric (i.e. undirected) longitudinal network data. In order to analyze a bipartite network data set, users may apply one-mode projection of each cross-sectional network to obtain a symmetric matrix for a node set which belongs to one of the two modes (see chapter 6.6 of M. Newman (2018) for an explanation).

[1]Given that we use highly non-informative priors, multiple chain swapping approaches, such as parallel tempering, may be helpful for a more efficient recovery of the multi-modal structure in the high dimensional parameter space (see chapter 6 of M. E. Newman & Barkema (1999) for a summary). We leave this possibility for future research.

Computation Time

The algorithm's computational cost is heavily influenced by the size of the input network data set as well as the number of dimensions and breaks to be estimated. In most cases, two-dimensional estimation will provide a reasonable description of the latent factors. When using a personal computer and the current version of the package, two-dimensional estimation for a single break results the following execution time for the corresponding longitudinal network size.

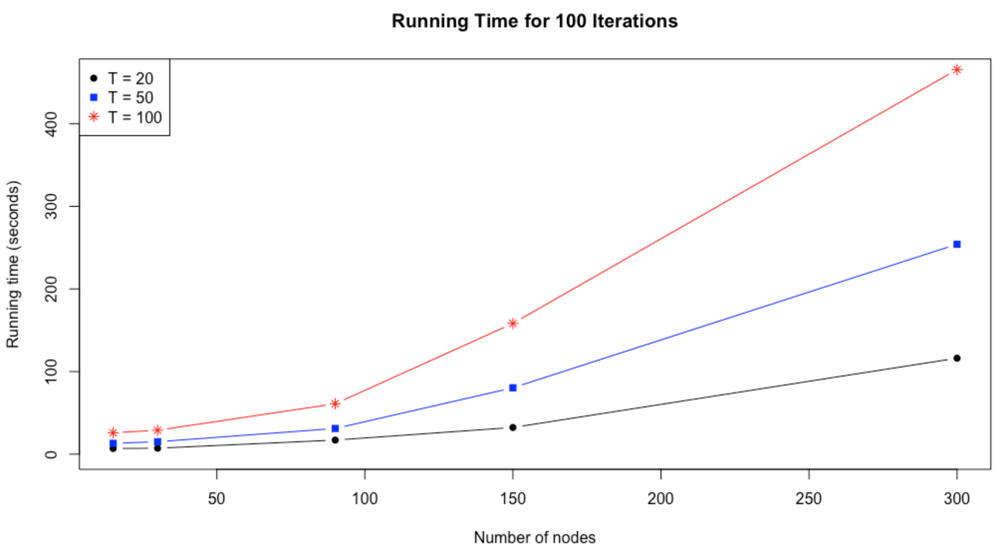

Figure 6: Execution time.

Because the number of breaks increases the number of states and state-specific latent factors to be estimated, increasing the number of states has a greater impact on `NetworkChange` function's execution time than the other factors. Please note that `BreakDiagnostic`, which runs `NetworkChange` over different numbers of breaks, spends time according to the length of time spent by `NetworkChange` given the aforementioned conditions.

Acknowledgements

We thank the reviewers and the editors of JOSS for their helpful comments. This work was supported by the New Faculty Startup Fund from Seoul National University [200-20210116] and the Japan Society for the Promotion of Science Early-Career Scientists Grant [JP19K13606] to Y.S.

References

Bürkner, P.-C., Gabry, J., Kay, M., & Vehtari, A. (2020). Posterior: Tools for working with posterior distributions. *Earthquake Spectra, R Package Version 0.1*, *3*.

Gibler, D. M. (2008). *International military alliances, 1648-2008*. CQ Press. https://doi.org/10.4135/9781604265781

Hoff, Peter D. (2015). Multilinear tensor regression for longitudinal relational data. *The Annals of Applied Statistics*, *9*(3), 1169–1193.

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

Krivitsky, P. N., & Handcock, M. S. (2008). Fitting position latent cluster models for social networks with latentnet. *Journal of Statistical Software*, *24*(5).

Newman, M. (2018). *Networks*. Oxford University Press.

Newman, M. E., & Barkema, G. T. (1999). *Monte carlo methods in statistical physics*. Oxford University Press.

Park, J. H., & Sohn, Y. (2020). Detecting structural changes in longitudinal network data. *Bayesian Analysis*, *15*(1), 133–157.

Peel, L., & Clauset, A. (2015). Detecting change points in the large-scale structure of evolving networks. *Twenty-Ninth AAAI Conference on Artificial Intelligence*.