Analyzing the Supreme Court Citation Network

September 20, 2017

Abstract

Introduction

Coming...

The Exponential Random Configuration Model

Let $c(t) \in \{0,1\}^N$ be a vector indicating which Supreme Court case has been cited at time t, where $c_i(t) = 1, i \in \{1,\dots,N\}$ indicates that the ith case has been cited at time t and $c_i(t) = 0$ indicates that the ith case has not been cited at time t. Furthermore, let

$$\mathcal{C}_t(N) = \{c(t) \in \{0,1\}^N : c_i(t) \in \{0,1\}\}$$

be the set of all possible citation combinations at time t. Note that the cardinality of $\mathcal{C}_t(N)$ increases exponentially for every newly added case, which results in 2^N elements.
The probability function of the ERCM is defined as

$$P_\theta(c(t) \mid c(t-)) = \frac{\exp\left(\theta^T \cdot h(c(t) \mid c(t-))\right)}{\sum_{c(t)^* \in \mathcal{C}} \exp\left(\theta^T \cdot h(c(t) \mid c(t-))\right)} \tag{1}$$

where $c(t-) \in \{0,1\}^{N \times (t-1)}$ is a matrix that indicates which cases have been citing in each other before time t, $\theta \in \mathbb{R}^q$ is a q-dimensional vector of parameters, $h : \mathcal{C}_t(N) \to \mathbb{R}^q$, $(t) \to (h_1(c(t)),\dots,h_q(c(t)))^T$ is a q-dimensional vector of different statistics and $\kappa(\theta) := \sum_{c(t)^* \in \mathcal{C}} \exp(\theta^T \cdot h(c(t)|c(t-)))$ is a normalization constant that ensures that (1) defines a probability function on $\mathcal{C}_t$.

The generative process of a model are informed by the decision regarding which network statistics $h(\cdot)$ are incorporated. We include the following statistics for the Supreme Court citation network:

$$h_{edges} : \mathcal{C}(N) \to \mathbb{R} \quad , \quad c(t) \to \sum_{i=1}^{N} c_i(t)$$

the number of citations made at time t.

$$h_{outstar} : \mathcal{C}(N) \to \mathbb{R} \quad , \quad c(t) \to \sum_{j<i}^{N} c_i(t) \cdot c_j(t) \cdot \sqrt{\frac{(t-a)(t-a-b)}{t^2}}$$

the number of weighted outstars occuring at time t. We argue that it should be more likely to cite more recent cases than cases that have been decided further in the past. For the weight

$$w(a,b) := \sqrt{\frac{(t-a)(t-a-b)}{t^2}}$$

we define a and b as the elapsed time since case i and j have been introduced to the network.

$$h_{triangle} : \mathcal{C}(N) \to \mathbb{R} \quad , \quad c(t) \to \sum_{j<i}^{N} c_i(t) \cdot c_j(t) \cdot c_j(t_{-i}) \cdot w(a,b)$$

where $c_j(t_{-i})$ indicates whether case j was cited at the time case i was introduced into the network. Just as for the outstar statistic, we include a weighting factor to favor more recent cases.

The individual entries $c_i(t)$ can be taken as a manifestation of single Bernoulli variables $C_i(t)$. This interpretation allows the following calculation regarding the

conditional distribution of $C_i(t)$:

$$\begin{aligned}\frac{P_\theta(C_i(t)=1 \mid C_i(t)^c = c_i(t)^c)}{P_\theta(C_i(t)=0 \mid C_i(t)^c = c_i(t)^c)} &= \frac{P_\theta(C_i(t)=1\ ,\ C_i(t)^c = c_i(t)^c)}{P_\theta(C_i(t)=0\ ,\ C_i(t)^c = c_i(t)^c)} \\ &= \frac{P_\theta(C(t)=c_i^+(t))}{P_\theta(C(t)=c_i^-(t))} \\ &= \frac{\exp(\theta^T \cdot h(c_i^+(t) \mid c(t-)))}{\exp(\theta^T \cdot h(c_i^-(t) \mid c(t-)))} \\ &= \exp(\theta^T \cdot (h(c_i^+(t) \mid c(t-)) - h(c_i^-(t) \mid c(t-)))\end{aligned}$$

This implies the following equation:

$$\text{logit}(P_\theta(C_i(t)=1 \mid C_i(t)^c = c_i(t)^c)) = \theta^T \cdot (h(c_i^+(t)\ \ |c(t-)) - h(c_i^-(t) \mid c(t-))) \quad (2)$$

In the equation above the following notations were used:

- $c_i^+(t)$ emerges from $c(t)$, while assuming $c_i(t) = 1$
- $c_i^-(t)$ emerges from $c(t)$, while assuming $c_i(t) = 0$
- The condition $C_i(t)^c = c_i(t)^c$ is short for: $C_j(t) = c_j(t)$ for all $j \in \{1, \ldots, N\}$ with $i \neq j$
- The expression $(\Delta c_i)(t) := h(c_i^+(t) \mid c(t-)) - h(c_i^-(t) \mid c(t-))$ is called the *change statistic*. The kth component of $(\Delta c_i)(t)$ captures the difference between citation networks $c_i^+(t)$ and $c_i^-(t)$ on the kth integrated statistic in the model

Estimation

Maximum Pseudo-Likelihood Estimator

One can assume that the dyads are independent of each other, which means that the random variables $C_i(t)$ inside the random vector $C(t)$ are independent of each other. In this case, the equation (2) reduces to

$$logit(P_\theta(C_i(t)=1)) = \theta^T \cdot (\Delta c_i)(t)$$

This corresponds with the logistic regression approach, where the observations of the dependent variables are simply edge values of the observed citation vector, and the observations of the covariate values are given as the scores of every single change

statistic. Therefore, the resulting likelihood function is of the following form:

$$\text{lik}(\theta) = P_\theta(C(t) = c(t)) = \prod_i \frac{\exp\left(\theta^T \Delta(c_i)(t)\right)}{1 + \exp\left(\theta^T \Delta(c_i)(t)\right)} \tag{3}$$

Maximum Likelihood Estimator

The more rigorous technique is to estimate the parameters directly with the log-likelihood function derived from (1), which has the following form:

$$\text{loglik}(\theta) = \theta^T \cdot h(c(t)|c(t-)) - \log(\kappa(\theta)) \tag{4}$$

The problem resulting from estimating the parameters with (4) is that the term

$$\kappa(\theta) := \sum_{c(t)^* \in \mathcal{C}(N)} \exp(\theta^T \cdot h(c(t)^*|c(t-)))$$

which sums up the weighted statistics of all possible binar vectors of length N, has to be evaluated. However, the cardinality of $\mathcal{C}(N)$ ($\#(\mathcal{C}) = 2^N$) is incredibly large and a direkt calculation of this sum is for already small N not feasible.

An solution for this limitation is based on the following consideration: Fix a vector of parameters $\theta_0 \in \Theta$ from the underlying parameter range Θ and compute for $\theta \in \Theta$ the expected value

$$\begin{aligned}
\mathbb{E}_{\theta_0}\left[\exp\left((\theta-\theta_0)^T \cdot \Gamma(C(t))\right)\right] &= \sum_{c(t)\in\mathcal{C}(N)} \exp\left((\theta-\theta_0)^T \cdot \Gamma(c(t))\right) \cdot \mathbb{P}_{\theta_0}(C(t) = c(t)) \\
&= \sum_{c(t)\in\mathcal{C}(N)} \exp\left((\theta-\theta_0)^T \cdot \Gamma(c(t))\right) \cdot \frac{\exp(\theta_0^T \cdot \Gamma(c(t)))}{\kappa(\theta_0)} \\
&= \frac{1}{\kappa(\theta_0)} \sum_{c(t)\in\mathcal{C}(N)} \exp\left(\theta^T \cdot \Gamma(c(t))\right) \\
&= \frac{\kappa(\theta)}{\kappa(\theta_0)}
\end{aligned}$$

This equation offers the following possibility: If one draws L random vectors $c^{(1)}(t), \ldots, c^{(L)}(t)$ out of a distribution $\mathbb{P}_{\theta_0}$ appropriately, one gets with the law of big numbers and a big enough sample L the following relation:

$$\frac{1}{L} \cdot \sum_{i=1}^{L} \exp\left((\theta-\theta_0)^T \cdot \Gamma(c^{(i)}(t))\right) \approx \mathbb{E}_{\theta_0}\left[\exp\left((\theta-\theta_0)^T \cdot \Gamma(C(t))\right)\right] = \frac{\kappa(\theta)}{\kappa(\theta_0)} \tag{5}$$

This approximate can then be used to approximate the log likelihood function.

Next, we will discuss how a sufficient number of suitable drawings $c^{(1)}(t), \ldots, c^{(L)}(t)$ can be sampled from the distribution $\mathbb{P}_{\theta_0}$.
For this purpose, the Markov Chain Monte Carlo (MCMC) methods can be used.

Gibbs sampling for the ERCM

To be able to compute the approximate likelihood function one needs a sufficiently large number of random vectors from the distribution $\mathbb{P}_{\theta_0}$. Snijders [**?**] introduces an approach to sample random networks for the ERGM framework by using *MCMC methods*. We adapt this approach for sampling appropriate binary vectors for the ERCM.

Gibbs sampling

Choose any vector $c^{(0)}(t) \in \mathcal{C}(N)$ (e.g. observed vector). Afterwards, the length L of the respective sub-sequence is determined. For $k \in \{0, ..., L-1\}$ execute the following steps recursively (here the vector in its kth iteration is denoted as $c^{(k)}(t)$):

1. Randomly choose a number $i \in \{1, \ldots, N\}$
2. Compute using the likelihood the value

$$\pi := \mathbb{P}_\theta(C_i(t) = 1 | C_i^c(t) = (c_i^{(k)}(t))^c) = \frac{\exp(\theta^T \cdot \Delta(c_i)(t))}{1 + \exp(\theta^T \cdot \Delta(c_i)(t))}$$

3. Draw a random number Z from $\mathrm{Bin}(1, \pi)$. If
   - $Z = 0$, define $c^{(k+1)}(t)$ via

$$c_p^{(k+1)}(t) = \begin{cases} 0 & \text{if } p = i \\ c_p^{(k)}(t) & \text{if } p \neq i \end{cases}$$

   - $Z = 1$, define $c^{(k+1)}(t)$ via

$$c_p^{(k+1)}(t) = \begin{cases} 1 & \text{if } p = i \\ c_p^{(k)}(t) & \text{if } p \neq i \end{cases}$$

4. Start at step 1 with $c^{(k+1)}(t)$.

The depicted algorithm provides a sequence of random vectors $c^{(0)}(t), ..., c^{(L)}(t)$. Since the orignial vector was chosen randomly and the first simulated vectors are very dependent on the chosen mvector (only one entry is changed per iteration!), usually the first B networks, where $N \ll B \ll L$, are discarded as the so called *Burn-In*.

Metropolis Hastings for the ERCM

Choose any vector $c^{(0)}(t) \in \mathcal{C}(N)$ to start with (e.g., the observed vector). For $k \in \{0, ..., L-1\}$ recursively proceed as follows:

1. Randomly choose a number $i \in \{1, \ldots, N\}$
2. Compute, using the equation (2) the value

$$\pi := \frac{\mathbb{P}_\theta(C_i(t) \neq c_i^{(k)}(t) \mid C_i(t)^c = c_i(t)^c)}{\mathbb{P}_\theta(C_i(t) = c_i^{(k)}(t) \mid C_i(t)^c = c_i(t)^c)}$$

3. Define $\delta := \min\{1, \pi\}$ and draw a random number Z from $\text{Bin}(1, \delta)$. If
   - $Z = 0$, let $c^{(k+1)}(t) := c^{(k)}(t)$
   - $Z = 1$, define $c^{(k+1)}(t)$ as

$$c_p^{(k+1)}(t) = \begin{cases} 1 - c_p^{(k)}(t) & \text{if } p = i \\ c_p^{(k)}(t) & \text{if } p \neq i \end{cases}$$

4. Start at step 1 with $c^{(k+1)}(t)$.

The first $B \ll L$ vectors are discarded as Burn-In.

Results/Discussion