COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Estimation of Exponential Ranomd Graph Models for Large Social Networks via Graph Limits

DIMACS Workshop on Statistical Analysis of Network Dynamics and Interactions | Nov. 7-8, 2013

Tian Zheng (tzheng@stat.columbia.edu)
Department of Statistics, Columbia University

Acknowledgement

- This research is supported by NSF.
- Joint work with Ran He

Outline

Introduction

Method

Results

Discussion

$$
\begin{aligned}
p_{\boldsymbol{\beta}}(G) &= \exp\left\{\sum_{i=1}^{k} \beta_i T_i(G) - \psi(\boldsymbol{\beta})\right\} \\
&= \exp\left\{\boldsymbol{\beta}' \boldsymbol{T}(G) - \psi(\boldsymbol{\beta})\right\},
\end{aligned}
$$

where $\boldsymbol{T}(G) = (T_1(G), \cdots, T_k(G))$ and

$$
\psi(\boldsymbol{\beta}) = \log \sum_{G \in \mathcal{G}_n} \exp\left(\boldsymbol{\beta}' \boldsymbol{T}(G)\right).
$$

Existing methods for estimation

- Pseudolikelihood approach
- Markov chain Monte Carlo based approach

Our approach

- is based on graph limits and a theoretical framework proposed by Chatterjee and Diaconis.
- uses techniques such as two dimensional simple function approximation to generalize the method.
- is an iterative algorithm to approximate the MLE.

Our approach

- is based on graph limits and a theoretical framework proposed by Chatterjee and Diaconis.
- uses techniques such as two dimensional simple function approximation to generalize the method.
- is an iterative algorithm to approximate the MLE.

Our approach

- is based on graph limits and a theoretical framework proposed by Chatterjee and Diaconis.
- uses techniques such as two dimensional simple function approximation to generalize the method.
- is an iterative algorithm to approximate the MLE.

Graph limits based approach

- Lovasz, Szegedy, Borgs and their coauthors develop a unifying theory of graph limits.
- Convergent graph sequences have a limit object, which can be represented as symmetric *measurable* functions, i.e, $w : [0,1]^2 \to [0,1]$ that satisfy $w(x,y) = w(y,x)$ for all $x, y \in [0,1]$.
- w-random graph of size n can be generated by
  - first assigning x_i, $i = 1, \ldots, n \sim \text{unif}[0,1]$ to the n nodes,
  - and $e_{ij} \sim \text{Bernoulli}(w(x_i, x_j))$.
- Every finite simple graph G can also be represented as a graph limit w^G in a natural way. Split the interval $[0,1]$ into n equal intervals $J_1, \cdots, J_n$, where $n = |V(G)|$. For $x \in J_i$, $y \in J_j$, define

$$w^G(x,y) = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

- Lovasz, Szegedy, Borgs and their coauthors develop a unifying theory of graph limits.
- Convergent graph sequences have a limit object, which can be represented as symmetric *measurable* functions, i.e, $w : [0,1]^2 \to [0,1]$ that satisfy $w(x,y) = w(y,x)$ for all $x, y \in [0,1]$.
- w-random graph of size n can be generated by
  - first assigning $x_i, i = 1, \ldots, n \sim \text{unif}[0,1]$ to the n nodes,
  - and $e_{ij} \sim \text{Bernoulli}(w(x_i, x_j))$.
- Every finite simple graph G can also be represented as a graph limit w^G in a natural way. Split the interval $[0,1]$ into n equal intervals $J_1, \cdots, J_n$, where $n = |V(G)|$. For $x \in J_i$, $y \in J_j$, define

$$w^G(x,y) = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

Graph limits based approach

- Lovasz, Szegedy, Borgs and their coauthors develop a unifying theory of graph limits.
- Convergent graph sequences have a limit object, which can be represented as symmetric *measurable* functions, i.e, $w : [0,1]^2 \to [0,1]$ that satisfy $w(x,y) = w(y,x)$ for all $x, y \in [0,1]$.
- w-random graph of size n can be generated by
  - first assigning $x_i, i = 1, \dots, n \sim \text{unif}[0,1]$ to the n nodes,
  - and $e_{ij} \sim \text{Bernoulli}(w(x_i, x_j))$.
- Every finite simple graph G can also be represented as a graph limit w^G in a natural way. Split the interval $[0,1]$ into n equal intervals $J_1, \cdots, J_n$, where $n = |V(G)|$. For $x \in J_i$, $y \in J_j$, define

$$w^G(x,y) = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

Graph limits based approach

- Lovasz, Szegedy, Borgs and their coauthors develop a unifying theory of graph limits.
- Convergent graph sequences have a limit object, which can be represented as symmetric *measurable* functions, i.e, $w : [0,1]^2 \to [0,1]$ that satisfy $w(x,y) = w(y,x)$ for all $x, y \in [0,1]$.
- w-random graph of size n can be generated by
  - first assigning $x_i, i = 1, \ldots, n \sim \text{unif}[0,1]$ to the n nodes,
  - and $e_{ij} \sim \text{Bernoulli}(w(x_i, x_j))$.
- Every finite simple graph G can also be represented as a graph limit w^G in a natural way. Split the interval $[0,1]$ into n equal intervals $J_1, \cdots, J_n$, where $n = |V(G)|$. For $x \in J_i$, $y \in J_j$, define

$$w^G(x,y) = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

Graph limits based approach

- Lovasz, Szegedy, Borgs and their coauthors develop a unifying theory of graph limits.
- Convergent graph sequences have a limit object, which can be represented as symmetric *measurable* functions, i.e, $w : [0,1]^2 \to [0,1]$ that satisfy $w(x,y) = w(y,x)$ for all $x, y \in [0,1]$.
- w-random graph of size n can be generated by
  - first assigning $x_i, i = 1, \ldots, n \sim \text{unif}[0,1]$ to the n nodes,
  - and $e_{ij} \sim \text{Bernoulli}(w(x_i, x_j))$.
- Every finite simple graph G can also be represented as a graph limit w^G in a natural way. Split the interval $[0,1]$ into n equal intervals $J_1, \cdots, J_n$, where $n = |V(G)|$. For $x \in J_i$, $y \in J_j$, define

$$w^G(x,y) = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

Graph limits based approach

- Lovasz, Szegedy, Borgs and their coauthors develop a unifying theory of graph limits.
- Convergent graph sequences have a limit object, which can be represented as symmetric *measurable* functions, i.e, $w : [0,1]^2 \to [0,1]$ that satisfy $w(x,y) = w(y,x)$ for all $x, y \in [0,1]$.
- w-random graph of size n can be generated by
  - first assigning x_i, $i = 1, \ldots, n \sim \text{unif}[0,1]$ to the n nodes,
  - and $e_{ij} \sim \text{Bernoulli}(w(x_i, x_j))$.
- Every finite simple graph G can also be represented as a graph limit w^G in a natural way. Split the interval $[0,1]$ into n equal intervals $J_1, \cdots, J_n$, where $n = |V(G)|$. For $x \in J_i$, $y \in J_j$, define

$$w^G(x,y) = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

- Chatterjee and Diaconis propose a quotient space of w, in which every simple graph G has an equivalence class $\widetilde{G}$, and define a distance $\delta_{\square}$ such that $(\widetilde{\mathcal{W}}, \delta_{\square})$ is a metric space.
- $\delta_{\square}(\widetilde{f}, \widetilde{g}) := \inf_{\sigma} d_{\square}(f, g_{\sigma})$, $g_{\sigma}(x, y) := g(\sigma x, \sigma y)$ and σ is a measure perserving bijection.
- Here $d_{\square} = \sup_{S,T \subseteq [0,1]} \left| \int_{S \times T} [f(x, y) - g(x, y)] dxdy \right|$.
- where S, T are measurable subsets of $[0, 1]$.

- Chatterjee and Diaconis propose a quotient space of w, in which every simple graph G has an equivalence class $\widetilde{G}$, and define a distance $\delta_\square$ such that $(\widetilde{\mathcal{W}}, \delta_\square)$ is a metric space.
- $\delta_\square(\widetilde{f}, \widetilde{g}) := \inf_\sigma d_\square(f, g_\sigma)$, $g_\sigma(x, y) := g(\sigma x, \sigma y)$ and σ is a measure perserving bijection.
- Here $d_\square = \sup_{S,T \subseteq [0,1]} \left| \int_{S \times T} [f(x, y) - g(x, y)] dxdy \right|$.
- where S, T are measurable subsets of $[0, 1]$.

Graph limits based approach (cont'd)

- Chatterjee and Diaconis propose a quotient space of w, in which every simple graph G has an equivalence class $\widetilde{G}$, and define a distance $\delta_\square$ such that $(\widetilde{\mathcal{W}}, \delta_\square)$ is a metric space.
- $\delta_\square(\widetilde{f}, \widetilde{g}) := \inf_\sigma d_\square(f, g_\sigma)$, $g_\sigma(x, y) := g(\sigma x, \sigma y)$ and σ is a measure perserving bijection.
- Here $d_\square = \sup_{S,T \subseteq [0,1]} \left| \int_{S \times T} [f(x, y) - g(x, y)] dxdy \right|$.
- where S, T are measurable subsets of $[0, 1]$.

Graph limits based approach (cont'd)

- Chatterjee and Diaconis propose a quotient space of w, in which every simple graph G has an equivalence class $\widetilde{G}$, and define a distance $\delta_\Box$ such that $(\widetilde{\mathcal{W}}, \delta_\Box)$ is a metric space.
- $\delta_\Box(\widetilde{f}, \widetilde{g}) := \inf_\sigma d_\Box(f, g_\sigma)$, $g_\sigma(x, y) := g(\sigma x, \sigma y)$ and σ is a measure perserving bijection.
- Here $d_\Box = \sup_{S,T \subseteq [0,1]} \left| \int_{S \times T} [f(x, y) - g(x, y)] dxdy \right|$.
- where S, T are measurable subsets of $[0, 1]$.

Graph limits based approach (cont'd)

- ERGM graph can be written as:

$$p_n(G) := e^{n^2(T(\widetilde{G})-\psi_n)},$$

where $T : \widetilde{\mathcal{W}} \to \mathbb{R}$ be a bounded continuous function on the metric space $(\widetilde{\mathcal{W}}, \delta_\square)$.

- Example:

$$\begin{aligned} T(\widetilde{G}) &= \sum_{i=1}^{3} \beta_i t(H_i, \widetilde{G}) \\ &= \frac{2\beta_1(\text{\# edges in G})}{n^2} + \frac{6\beta_2(\text{\# two-stars in G})}{n^3} \\ &\quad + \frac{6(\beta_3 - 2\beta_2)(\text{\# triangles in G})}{n^3}. \end{aligned}$$

Graph limits based approach (cont'd)

- ERGM graph can be written as:

$$p_n(G) := e^{n^2(T(\widetilde{G})-\psi_n)},$$

where $T : \widetilde{\mathcal{W}} \to \mathbb{R}$ be a bounded continuous function on the metric space $(\widetilde{\mathcal{W}}, \delta_\square)$.

- Example:

$$\begin{aligned} T(\widetilde{G}) &= \sum_{i=1}^{3} \beta_i t(H_i, \widetilde{G}) \\ &= \frac{2\beta_1(\#\text{ edges in G})}{n^2} + \frac{6\beta_2(\#\text{ two-stars in G})}{n^3} \\ &\quad + \frac{6(\beta_3 - 2\beta_2)(\#\text{ triangles in G})}{n^3}. \end{aligned}$$

Graph limits based approach (cont'd)

- Assume $\widetilde{w_0}$ is the graph limit of $\widetilde{G_n}$ as $n \to \infty$.
- For a graph G_n of size n, assuming w_0, we have

$$\begin{aligned} \log p_n(G_n) &= T(G_n) - \psi_n \\ &= \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left[e_{ij} \log w_0(x_i, x_j) + (1 - e_{ij}) \log(1 - w_0(x_i, x_j)) \right], \end{aligned}$$

- Here x_i and x_j are random draws from the uniform distribution on $[0, 1]$.
- As $n \to \infty$, we then have

$$\lim_{n \to \infty} \psi_n = \sup_{\tilde{w} \in \widetilde{W}} \left(T(\tilde{w}) - I(\tilde{w}) \right),$$

where

$$\begin{aligned} I(\tilde{w}) &= \iint_{[0,1]^2} I(w(x, y)) dx dy \\ I(u) &= \frac{1}{2} u \log u + \frac{1}{2} (1 - u) \log(1 - u) \end{aligned}$$

- Assume $\widetilde{w_0}$ is the graph limit of $\widetilde{G_n}$ as $n \to \infty$.
- For a graph G_n of size n, assuming w_0, we have

$$\begin{aligned} \log p_n(G_n) &= T(G_n) - \psi_n \\ &= \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left[e_{ij} \log w_0(x_i, x_j) + (1 - e_{ij}) \log(1 - w_0(x_i, x_j)) \right], \end{aligned}$$

- Here x_i and x_j are random draws from the uniform distribution on $[0, 1]$.
- As $n \to \infty$, we then have

$$\lim_{n \to \infty} \psi_n = \sup_{\tilde{w} \in \widetilde{W}} \left(T(\tilde{w}) - I(\tilde{w}) \right),$$

where

$$\begin{aligned} I(\tilde{w}) &= \iint_{[0,1]^2} I(w(x, y)) dxdy \\ I(u) &= \frac{1}{2} u \log u + \frac{1}{2}(1 - u) \log(1 - u) \end{aligned}$$

Graph limits based approach (cont'd)

- Assume $\widetilde{w_0}$ is the graph limit of $\widetilde{G_n}$ as $n \to \infty$.
- For a graph G_n of size n, assuming w_0, we have

$$\begin{aligned}\log p_n(G_n) &= T(G_n) - \psi_n \\ &= \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left[e_{ij} \log w_0(x_i, x_j) + (1 - e_{ij}) \log(1 - w_0(x_i, x_j))\right],\end{aligned}$$

- Here x_i and x_j are random draws from the uniform distribution on $[0, 1]$.
- As $n \to \infty$, we then have

$$\lim_{n\to\infty} \psi_n = \sup_{\tilde{w} \in \widetilde{W}} \left(T(\tilde{w}) - I(\tilde{w})\right),$$

where

$$\begin{aligned} I(\tilde{w}) &= \iint_{[0,1]^2} I(w(x, y))dxdy \\ I(u) &= \frac{1}{2} u \log u + \frac{1}{2}(1 - u) \log(1 - u) \end{aligned}$$

Graph limits based approach (cont'd)

- Assume $\widetilde{w_0}$ is the graph limit of $\widetilde{G_n}$ as $n \to \infty$.
- For a graph G_n of size n, assuming w_0, we have

$$\begin{aligned} \log p_n(G_n) &= T(G_n) - \psi_n \\ &= \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left[e_{ij} \log w_0(x_i, x_j) + (1 - e_{ij}) \log(1 - w_0(x_i, x_j)) \right], \end{aligned}$$

- Here x_i and x_j are random draws from the uniform distribution on $[0, 1]$.
- As $n \to \infty$, we then have

$$\lim_{n \to \infty} \psi_n = \sup_{\tilde{w} \in \widetilde{W}} \left(T(\tilde{w}) - I(\tilde{w}) \right),$$

where

$$\begin{aligned} I(\tilde{w}) &= \iint_{[0,1]^2} I(w(x, y)) dxdy \\ I(u) &= \frac{1}{2} u \log u + \frac{1}{2} (1 - u) \log(1 - u) \end{aligned}$$

Graph limits based approach (cont'd)

- When n is large, almost all random graphs G_n drawn from ERGM induced by T are close to w random-graphs F when $T(\widetilde{F}) - I(\widetilde{F})$ is maximized.
- Based on these findings, Chatterjee and Diaconis remarked that one can approximate MLE, by evaluating $\psi(\beta)$ on a fine grid in β space and then carrying out the maximization by classical methods such as a grid search.

Graph limits based approach (cont'd)

- When n is large, almost all random graphs G_n drawn from ERGM induced by T are close to w random-graphs F when $T(\widetilde{F}) - I(\widetilde{F})$ is maximized.
- Based on these findings, Chatterjee and Diaconis remarked that one can approximate MLE, by evaluating $\psi(\beta)$ on a fine grid in β space and then carrying out the maximization by classical methods such as a grid search.

Outline

Introduction

Method

Results

Discussion

Two-dimensional simple functions approximation

- For any m, split $[0,1]^2$ into m^2 lattices with equal area,

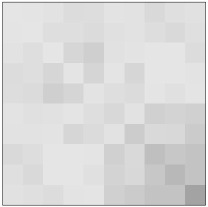
$$A_{ij} = \left\{(x,y) : x \in \left[\frac{i-1}{m}, \frac{i}{m}\right) \text{ and } y \in \left[\frac{j-1}{m}, \frac{j}{m}\right)\right\},$$

 where $i, j = 1, \cdots, m$. And let $\{c_{ij}\}$ be a sequence of real numbers between 0 and 1.

-

$$\hat{w}_m = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x,y),$$

where $\{\hat{c}_{ij}; i,j = 1, \ldots m\} = \underset{\{c_{ij}; i,j=1,\ldots m\}}{\operatorname{argmax}} [T(w_m) - I(w_m)].$



Two-dimensional simple functions approximation

- For any m, split $[0,1]^2$ into m^2 lattices with equal area,

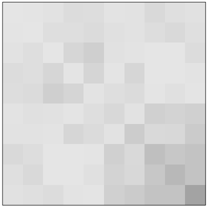
$$A_{ij} = \left\{(x,y) : x \in \left[\frac{i-1}{m}, \frac{i}{m}\right) \text{ and } y \in \left[\frac{j-1}{m}, \frac{j}{m}\right)\right\},$$

  where $i, j = 1, \cdots, m$. And let $\{c_{ij}\}$ be a sequence of real numbers between 0 and 1.

-

$$\hat{w}_m = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x,y),$$

  where $\{\hat{c}_{ij}; i,j = 1, \ldots m\} = \underset{\{c_{ij}; i,j=1,\ldots m\}}{\operatorname{argmax}} [T(w_m) - I(w_m)]$.



- For example, we can easily derive (for an ERGM model using *egdes*, *two-stars* and *triangles*.)

$$
\begin{aligned}
& T(w_m) - I(w_m) \\
= \quad & \frac{\beta_1}{m^2} \sum_{ij} c_{ij} + \frac{\beta_2}{m^3} \sum_{ijk} c_{ij} c_{jk} + \frac{\beta_3}{m^3} \sum_{ijk} c_{ij} c_{jk} c_{ik} \\
& - \frac{1}{2m^2} \sum_{ij} \left[c_{ij} \log c_{ij} + (1 - c_{ij}) \log(1 - c_{ij}) \right].
\end{aligned}
$$

Estimating parameters of ERGM via graph limits

- Give an initial value of β, $\beta^{(0)}$.
- For each t,
  - Given $\beta^{(t)}$, use simple function approximation to estimate $\tilde{w}^{(t)}$ by maximizing $T_{\beta^{(t)}}(\tilde{w}) - I(\tilde{w})$.
    The corresponding simple function is
    $$\hat{w}_m^{(t)} = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x, y)$$
    and $\hat{\psi}^{(t)} = T_{\beta^{(t)}}\left(\widetilde{\hat{w}_m^{(t)}}\right) - I\left(\widetilde{\hat{w}_m^{(t)}}\right)$.
  - set $\beta^{(t+1)} = \underset{\beta}{\operatorname{argmax}} \log \hat{p}_n(\beta; G, \hat{w}_m^{(t)})$.
- Stop once $||\beta^{(t+1)} - \beta^{(t)}|| < \varepsilon$ for some fixed ε. And the corresponding $\beta^{(t+1)}$ is the GLMLE.

- Give an initial value of β, $\beta^{(0)}$.
- For each t,
  - Given $\beta^{(t)}$, use simple function approximation to estimate $\tilde{w}^{(t)}$ by maximizing $T_{\beta^{(t)}}(\tilde{w}) - I(\tilde{w})$.
    The corresponding simple function is
    $\hat{w}_m^{(t)} = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x, y)$
    and $\hat{\psi}^{(t)} = T_{\beta^{(t)}}\left(\widetilde{\hat{w}_m^{(t)}}\right) - I\left(\widetilde{\hat{w}_m^{(t)}}\right)$.
  - set $\beta^{(t+1)} = \operatorname{argmax}_{\beta} \log \hat{p}_n(\beta; G, \hat{w}_m^{(t)})$.
- Stop once $||\beta^{(t+1)} - \beta^{(t)}|| < \varepsilon$ for some fixed ε. And the corresponding $\beta^{(t+1)}$ is the GLMLE.

Estimating parameters of ERGM via graph limits

- Give an initial value of β, $\beta^{(0)}$.
- For each t,
  - Given $\beta^{(t)}$, use simple function approximation to estimate $\tilde{w}^{(t)}$ by maximizing $T_{\beta^{(t)}}(\tilde{w}) - I(\tilde{w})$.
    The corresponding simple function is
    $$\hat{w}_m^{(t)} = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x, y)$$
    and $\hat{\psi}^{(t)} = T_{\beta^{(t)}}\left(\widetilde{\hat{w}_m^{(t)}}\right) - I\left(\widetilde{\hat{w}_m^{(t)}}\right)$.
  - set $\beta^{(t+1)} = \operatorname*{argmax}_{\beta} \log \hat{p}_n(\beta; G, \hat{w}_m^{(t)})$.
- Stop once $||\beta^{(t+1)} - \beta^{(t)}|| < \varepsilon$ for some fixed ε. And the corresponding $\beta^{(t+1)}$ is the GLMLE.

- Give an initial value of $\boldsymbol{\beta}$, $\boldsymbol{\beta}^{(0)}$.
- For each t,
  - Given $\beta^{(t)}$, use simple function approximation to estimate $\tilde{w}^{(t)}$ by maximizing $T_{\boldsymbol{\beta}^{(t)}}(\tilde{w}) - I(\tilde{w})$.
    The corresponding simple function is
    $$\hat{w}_m^{(t)} = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x, y)$$
    and $\hat{\psi}^{(t)} = T_{\boldsymbol{\beta}^{(t)}}\left(\widetilde{\hat{w}_m^{(t)}}\right) - I\left(\widetilde{\hat{w}_m^{(t)}}\right)$.
  - set $\boldsymbol{\beta}^{(t+1)} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \log \hat{p}_n(\boldsymbol{\beta}; G, \hat{w}_m^{(t)})$.
- Stop once $||\beta^{(t+1)} - \beta^{(t)}|| < \varepsilon$ for some fixed ε. And the corresponding $\beta^{(t+1)}$ is the GLMLE.

- Give an initial value of $\boldsymbol{\beta}$, $\boldsymbol{\beta}^{(0)}$.
- For each t,
  - Given $\beta^{(t)}$, use simple function approximation to estimate $\tilde{w}^{(t)}$ by maximizing $T_{\boldsymbol{\beta}^{(t)}}(\tilde{w}) - I(\tilde{w})$.
    The corresponding simple function is
    $$\hat{w}_m^{(t)} = \sum_{i,j=1}^{m} \hat{c}_{ij} \mathbf{1}_{A_{ij}}(x, y)$$
    and $\hat{\psi}^{(t)} = T_{\boldsymbol{\beta}^{(t)}}\left(\widetilde{\hat{w}_m^{(t)}}\right) - I\left(\widetilde{\hat{w}_m^{(t)}}\right)$.
  - set $\boldsymbol{\beta}^{(t+1)} = \operatorname*{argmax}_{\boldsymbol{\beta}} \log \hat{p}_n(\boldsymbol{\beta}; G, \hat{w}_m^{(t)})$.
- Stop once $||\boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)}|| < \varepsilon$ for some fixed ε. And the corresponding $\boldsymbol{\beta}^{(t+1)}$ is the GLMLE.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating β
  - For exponential family,
    $$E_\beta\left[\boldsymbol{T}(G)\right] = \nabla\psi(\beta)$$
    .
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
    $$\begin{aligned}\nabla \log p_n(\beta; G) &= n^2\left\{\boldsymbol{T}(G) - \nabla\psi(\beta)\right\}\\ &= n^2\left\{\boldsymbol{T}(G) - E_\beta\left[\boldsymbol{T}(G)\right]\right\}.\end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating β
  - For exponential family,
    $$E_\beta\left[\boldsymbol{T}(G)\right] = \nabla\psi(\beta)$$
    .
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
    $$\begin{aligned}\nabla \log p_n(\beta; G) &= n^2\left\{\boldsymbol{T}(G) - \nabla\psi(\beta)\right\}\\ &= n^2\left\{\boldsymbol{T}(G) - E_\beta\left[\boldsymbol{T}(G)\right]\right\}.\end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating $\boldsymbol{\beta}$
  - For exponential family,

$$E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right] = \nabla\psi(\boldsymbol{\beta})$$

.

  - Thus the first derivative of the log-likelihood function for an ERGM graph G is

$$\begin{aligned}\nabla \log p_n(\boldsymbol{\beta}; G) &= n^2\left\{\boldsymbol{T}(G) - \nabla\psi(\boldsymbol{\beta})\right\}\\ &= n^2\left\{\boldsymbol{T}(G) - E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right]\right\}.\end{aligned}$$

- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating $\boldsymbol{\beta}$
  - For exponential family,
$$E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right] = \nabla\psi(\boldsymbol{\beta})$$
.
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
$$\begin{aligned}\nabla \log p_n(\boldsymbol{\beta}; G) &= n^2 \left\{\boldsymbol{T}(G) - \nabla\psi(\boldsymbol{\beta})\right\} \\ &= n^2 \left\{\boldsymbol{T}(G) - E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right]\right\}.\end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating $\boldsymbol{\beta}$
  - For exponential family,
    $$E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right] = \nabla\psi(\boldsymbol{\beta})$$
    .
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
    $$\begin{aligned} \nabla \log p_n(\boldsymbol{\beta}; G) &= n^2 \left\{\boldsymbol{T}(G) - \nabla\psi(\boldsymbol{\beta})\right\} \\ &= n^2 \left\{\boldsymbol{T}(G) - E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right]\right\}. \end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating $\boldsymbol{\beta}$
  - For exponential family,
  $$E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right] = \nabla\psi(\boldsymbol{\beta})$$
  .
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
  $$\begin{aligned}\nabla \log p_n(\boldsymbol{\beta}; G) &= n^2\left\{\boldsymbol{T}(G) - \nabla\psi(\boldsymbol{\beta})\right\} \\ &= n^2\left\{\boldsymbol{T}(G) - E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right]\right\}.\end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating $\boldsymbol{\beta}$
  - For exponential family,
$$E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right] = \nabla\psi(\boldsymbol{\beta})$$
.
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
$$\begin{aligned}\nabla \log p_n(\boldsymbol{\beta}; G) &= n^2\left\{\boldsymbol{T}(G) - \nabla\psi(\boldsymbol{\beta})\right\} \\ &= n^2\left\{\boldsymbol{T}(G) - E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right]\right\}.\end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Practical remarks

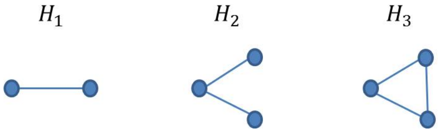
- Initial values: use w corresponding to the observed graph to find initial value of β.
- Updating w_m
- Updating $\boldsymbol{\beta}$
  - For exponential family,
$$E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right] = \nabla\psi(\boldsymbol{\beta})$$
.
  - Thus the first derivative of the log-likelihood function for an ERGM graph G is
$$\begin{aligned} \nabla \log p_n(\boldsymbol{\beta}; G) &= n^2 \left\{\boldsymbol{T}(G) - \nabla\psi(\boldsymbol{\beta})\right\} \\ &= n^2 \left\{\boldsymbol{T}(G) - E_{\boldsymbol{\beta}}\left[\boldsymbol{T}(G)\right]\right\}. \end{aligned}$$
- Computational complexity
  - Obtaining w_m^G in the initial step takes $O(n^2)$.
  - In each iteration, the computational complexity is $O(m^3)$.

Outline

Results

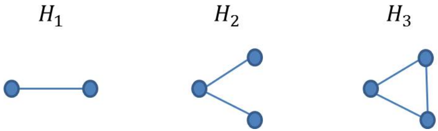
- Can be used on large network.
- Outperform MCMC-based algorithm, especially when the network is large.
- Run faster then MCMC-based algorithm.

- Consider an ERGM using homomorphism densities $t(H_i, \cdot)$ as sufficient statistics, where H_1 is edge, H_2 is two-star and H_3 is triangle.



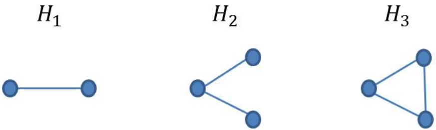
- The true value of the parameters β is $\beta = (-2, -1, 1)$.
- Using the R function `simulate.ergm` from the `ergm` package, we generate ERGM graphs of different sizes ($n = 100, 200, 500, 1000, 2000, 4000$) for this model.
- In each case, we simulate 100 graphs and apply our algorithm as well as MCMC algorithm (R function `ergm`) to model these data.
- We set $m = 10$

- Consider an ERGM using homomorphism densities $t(H_i, \cdot)$ as sufficient statistics, where H_1 is edge, H_2 is two-star and H_3 is triangle.



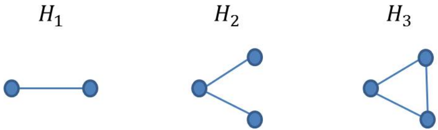
- The true value of the parameters β is $\beta = (-2, -1, 1)$.
- Using the R function `simulate.ergm` from the `ergm` package, we generate ERGM graphs of different sizes ($n = 100, 200, 500, 1000, 2000, 4000$) for this model.
- In each case, we simulate 100 graphs and apply our algorithm as well as MCMC algorithm (R function `ergm`) to model these data.
- We set $m = 10$

- Consider an ERGM using homomorphism densities $t(H_i, \cdot)$ as sufficient statistics, where H_1 is edge, H_2 is two-star and H_3 is triangle.



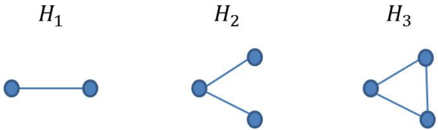
- The true value of the parameters β is $\beta = (-2, -1, 1)$.
- Using the R function `simulate.ergm` from the `ergm` package, we generate ERGM graphs of different sizes ($n = 100, 200, 500, 1000, 2000, 4000$) for this model.
- In each case, we simulate 100 graphs and apply our algorithm as well as MCMC algorithm (R function `ergm`) to model these data.
- We set $m = 10$

- Consider an ERGM using homomorphism densities $t(H_i, \cdot)$ as sufficient statistics, where H_1 is edge, H_2 is two-star and H_3 is triangle.



- The true value of the parameters β is $\beta = (-2, -1, 1)$.
- Using the R function `simulate.ergm` from the `ergm` package, we generate ERGM graphs of different sizes ($n = 100, 200, 500, 1000, 2000, 4000$) for this model.
- In each case, we simulate 100 graphs and apply our algorithm as well as MCMC algorithm (R function `ergm`) to model these data.
- We set $m = 10$

- Consider an ERGM using homomorphism densities $t(H_i, \cdot)$ as sufficient statistics, where H_1 is edge, H_2 is two-star and H_3 is triangle.



- The true value of the parameters β is $\beta = (-2, -1, 1)$.
- Using the R function `simulate.ergm` from the `ergm` package, we generate ERGM graphs of different sizes ($n = 100, 200, 500, 1000, 2000, 4000$) for this model.
- In each case, we simulate 100 graphs and apply our algorithm as well as MCMC algorithm (R function `ergm`) to model these data.
- We set $m = 10$

Simulation study 1 (cont'd)

size n	GLMLE			MCMCMLE		
	Bias($\hat{\beta}_1$) se($\hat{\beta}_1$)	Bias($\hat{\beta}_2$) se($\hat{\beta}_2$)	Bias($\hat{\beta}_3$) se($\hat{\beta}_3$)	Bias($\hat{\beta}_1$) se($\hat{\beta}_1$)	Bias($\hat{\beta}_2$) se($\hat{\beta}_2$)	Bias($\hat{\beta}_3$) se($\hat{\beta}_3$)
100	−0.017 (0.206)	−0.429 (5.055)	0.929 (7.161)	0.042 (0.163)	−0.496 (1.738)	9.800 (7.638)
200	−0.022 (0.100)	0.137 (1.369)	0.075 (1.667)	0.033 (0.188)	−1.757 (3.968)	23.780 (18.074)
500	−0.490 (0.019)	0.285 (0.491)	0.079 (2.433)	−0.481 (0.069)	0.598 (1.725)	−9.748 (43.559)
1000	−0.922 (0.013)	0.045 (0.381)	0.154 (0.330)	−0.917 (0.048)	0.483 (2.660)	−27.233 (102.808)
2000	−1.347 (0.009)	−0.209 (0.347)	0.355 (0.255)	−1.346 (0.029)	0.458 (3.787)	−20.266 (188.530)
4000	−1.741 (0.007)	−0.417 (0.307)	0.547 (0.127)	−1.742 (0.023)	0.588 (6.431)	18.510 (379.371)

Simulation study 2

- W-random graph is a method to generate random graph using a given graph limit w.
  - Generate n independent numbers $X_1, \cdots, X_n$ from the uniform distribution $U(0, 1)$.
  - Connect nodes i and j by an edge with probability $w(X_i, X_j)$, independently for every pair.
- All other settings are the same as simulation study 1.

Simulation study 2

- W-random graph is a method to generate random graph using a given graph limit w.
  - Generate n independent numbers $X_1, \cdots, X_n$ from the uniform distribution $U(0, 1)$.
  - Connect nodes i and j by an edge with probability $w(X_i, X_j)$, independently for every pair.
- All other settings are the same as simulation study 1.

Simulation study 2

- W-random graph is a method to generate random graph using a given graph limit w.
  - Generate n independent numbers $X_1, \cdots, X_n$ from the uniform distribution $U(0, 1)$.
  - Connect nodes i and j by an edge with probability $w(X_i, X_j)$, independently for every pair.
- All other settings are the same as simulation study 1.

- W-random graph is a method to generate random graph using a given graph limit w.
  - Generate n independent numbers $X_1, \cdots, X_n$ from the uniform distribution $U(0, 1)$.
  - Connect nodes i and j by an edge with probability $w(X_i, X_j)$, independently for every pair.
- All other settings are the same as simulation study 1.

size n	GLMLE			MCMCMLE		
	Bias($\hat{\beta}_1$) se($\hat{\beta}_1$)	Bias($\hat{\beta}_2$) se($\hat{\beta}_2$)	Bias($\hat{\beta}_3$) se($\hat{\beta}_3$)	Bias($\hat{\beta}_1$) se($\hat{\beta}_1$)	Bias($\hat{\beta}_2$) se($\hat{\beta}_2$)	Bias($\hat{\beta}_3$) se($\hat{\beta}_3$)
100	0.110 (0.694)	−2.412 (16.639)	0.182 (10.243)	0.004 (0.150)	0.487 (1.546)	7.164 (8.593)
200	−0.018 (0.045)	0.357 (0.661)	−0.098 (2.275)	−0.015 (0.114)	0.803 (1.125)	−6.063 (17.025)
500	−0.009 (0.012)	0.223 (0.064)	−0.103 (0.127)	−0.031 (0.068)	0.979 (0.661)	−1.681 (8.269)
1000	−0.009 (0.006)	0.225 (0.021)	−0.125 (0.040)	−0.031 (0.051)	0.962 (0.520)	−0.557 (5.283)
2000	−0.007 (0.003)	0.219 (0.021)	−0.110 (0.045)	−0.031 (0.030)	0.982 (0.307)	−1.263 (4.180)
4000	−0.007 (0.002)	0.212 (0.017)	−0.094 (0.029)	−0.035 (0.024)	1.029 (0.240)	−1.452 (2.960)

Real data analysis

- We apply our method to two real large social networks from Slashdot, a technology-related news website that has a large specific user community.

	nodes	edges	two-stars	triangles	transtivity ratio
Slashdot0811	77,360	469,180	68,516,301	551,724	0.02416
Slashdot0902	82,168	504,230	74,983,589	602,592	0.02411

- Although MCMC-based approach works in theory for large networks, it fails in practice, primarily because these two networks are too large to be coerced to objects to which the `ergm` function can be applied. Our GLMLE algorithm still works.
  - *Slashdot0811*: $(-4.5109, -1.5863, 1.6871)$, running time for obtaining w^G is 392 seconds, while that of estimation is 153 seconds.
  - *Slashdot0902*: $(-4.6502, -1.8122, 1.9430)$, running time for obtaining w^G is 436 seconds, while that of estimation is 124 seconds.

Real data analysis

- We apply our method to two real large social networks from Slashdot, a technology-related news website that has a large specific user community.

	nodes	edges	two-stars	triangles	transtivity ratio
Slashdot0811	77,360	469,180	68,516,301	551,724	0.02416
Slashdot0902	82,168	504,230	74,983,589	602,592	0.02411

- Although MCMC-based approach works in theory for large networks, it fails in practice, primarily because these two networks are too large to be coerced to objects to which the `ergm` function can be applied. Our GLMLE algorithm still works.
  - *Slashdot0811*: $(-4.5109, -1.5863, 1.6871)$, running time for obtaining w^G is 392 seconds, while that of estimation is 153 seconds.
  - *Slashdot0902*: $(-4.6502, -1.8122, 1.9430)$, running time for obtaining w^G is 436 seconds, while that of estimation is 124 seconds.

- We apply our method to two real large social networks from Slashdot, a technology-related news website that has a large specific user community.

	nodes	edges	two-stars	triangles	transtivity ratio
Slashdot0811	77,360	469,180	68,516,301	551,724	0.02416
Slashdot0902	82,168	504,230	74,983,589	602,592	0.02411

- Although MCMC-based approach works in theory for large networks, it fails in practice, primarily because these two networks are too large to be coerced to objects to which the `ergm` function can be applied. Our GLMLE algorithm still works.
  - *Slashdot0811*: $(-4.5109, -1.5863, 1.6871)$, running time for obtaining w^G is 392 seconds, while that of estimation is 153 seconds.
  - *Slashdot0902*: $(-4.6502, -1.8122, 1.9430)$, running time for obtaining w^G is 436 seconds, while that of estimation is 124 seconds.

- We apply our method to two real large social networks from Slashdot, a technology-related news website that has a large specific user community.

	nodes	edges	two-stars	triangles	transtivity ratio
Slashdot0811	77,360	469,180	68,516,301	551,724	0.02416
Slashdot0902	82,168	504,230	74,983,589	602,592	0.02411

- Although MCMC-based approach works in theory for large networks, it fails in practice, primarily because these two networks are too large to be coerced to objects to which the `ergm` function can be applied. Our GLMLE algorithm still works.
  - *Slashdot0811*: $(-4.5109, -1.5863, 1.6871)$, running time for obtaining w^G is 392 seconds, while that of estimation is 153 seconds.
  - *Slashdot0902*: $(-4.6502, -1.8122, 1.9430)$, running time for obtaining w^G is 436 seconds, while that of estimation is 124 seconds.

- In order to compare our method with MCMC-based approach, we obtain a random subnetwork G_{sub} from the *Slashdot0902* network via link-tracing-based sampling method. It contains 376 nodes, $1,609$ edges, $48,915$ two-stars and $1,661$ triangles.
- Besides the above model, we consider another model:

$$
\begin{aligned}
T(\widetilde{G}) &= \beta_1(\text{edges density}) + \beta_2(\text{triangle percent}) \\
&= \frac{2\beta_1(\text{\# edges in G})}{n^2} \\
&\quad + \frac{\beta_2(\text{\# triangles in G})}{(\text{\# two-stars in G}) - 2 \times (\text{\# triangles in G})}.
\end{aligned}
$$

- In order to compare our method with MCMC-based approach, we obtain a random subnetwork G_{sub} from the *Slashdot0902* network via link-tracing-based sampling method. It contains 376 nodes, $1,609$ edges, $48,915$ two-stars and $1,661$ triangles.
- Besides the above model, we consider another model:

$$
\begin{aligned}
T(\widetilde{G}) &= \beta_1(\text{edges density}) + \beta_2(\text{triangle percent}) \\
&= \frac{2\beta_1(\text{\# edges in G})}{n^2} \\
&\quad + \frac{\beta_2(\text{\# triangles in G})}{(\text{\# two-stars in G}) - 2 \times (\text{\# triangles in G})}.
\end{aligned}
$$

Real data analysis (cont'd)

Method	$\hat{\beta}$	corresponding w	$\frac{1}{n^2}\log(p_n)$
Model 1			
MCMCMLE	$(-2.5161, 3.3917, 43.2382)$	w_1	-44.1442
GLMLE	$(-1.8415, -0.7689, 0.7705)$	w_2	-0.0558
Model 2			
MCMCMLE	$(-1.6072, 0.1206)$	w_3	-0.1408
GLMLE	$(-2.1921, 0.0714)$	w_4	-0.0518

Visualization of graph limit objects used

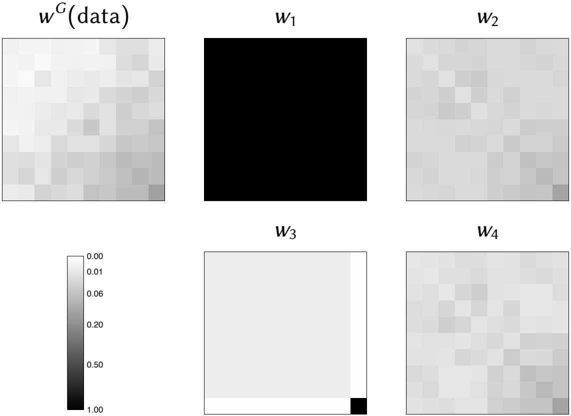


Figure : Heat map of graph limits w_1,w_2,w_3,w_4 and the graph limit representation of G_{sub}, w^G, as in above table. The different shades of gray represent the values of $w(x, y) \in [0, 1]$, with black being 1 and white 0.

Hypothesis testing

- We conduct a likelihood ratio test based on the approximate likelihood values for a number of models to test whether the values of each parameter in GLMLE is statistically significant.

Model	log-likelihood	Deviance	Deviance d.f.	p-value
Model 1				
NULL	-48997.19	—	—	—
T_1 only	-8085.31	40911.88	1	$< 1 \times 10^{-16}$
T_1 and T_2	-8019.34	65.97	1	4.44×10^{-16}
model 1	-7887.76	131.58	1	$< 1 \times 10^{-16}$
Model 2				
NULL	-48997.19	—	—	—
T_1 only	-8085.31	40911.88	1	$< 1 \times 10^{-16}$
model 2	-7321.27	764.04	1	$< 1 \times 10^{-16}$

Future work

- Choosing m.
- Examine the numerical stability.
- Apply our algorithm to more general exponential random graph models.

Thank you!