

Estimation in ERGMs

David Hunter

Department of Statistics
Penn State University

U. Washington network modeling group
Research supported by NIDA Grant DA012831 and NICHD Grant HD041877

Sunbelt 2006

Exponential Random Graph Model (ERGM)

$$P_{\theta}(Y = y) \propto \exp\{\theta^t g(y)\}$$

or

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)},$$

where

- Y is a random network on n nodes (a matrix of 0's and 1's)
- θ is a vector of parameters
- $g(y)$ is a known vector of graph statistics on y
- $\kappa(\theta)$ makes all the probabilities sum to 1

How does it run?



http://www.penguin3d.com/gallery/car_cartoon1.htm

- This talk goes a bit “under the hood”.
- However, it is NOT necessary to understand how the motor works in order to drive the car.

- 1 Maximum likelihood estimation
- 2 Approximating the MLE
- 3 Simulating random networks via MCMC
- 4 Maximum pseudolikelihood
- 5 Goodness of fit

- 1 Maximum likelihood estimation
- 2 Approximating the MLE
- 3 Simulating random networks via MCMC
- 4 Maximum pseudolikelihood
- 5 Goodness of fit

The goal of estimation

Exponential Random Graph Model (ERGM)

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}$$

If θ is not known, the above equation defines a model *class*, not a model.

Goal:

Use observed data (a network y^{obs}) to determine the “best” model from the model class.

In other words, find the “best” θ .

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}$$

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}$$

- It follows that $\kappa(\theta)$ is a normalizing “constant”:

$$\kappa(\theta) = \sum_{\text{all possible graphs } z} \exp\{\theta^t g(z)\}.$$

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}, \text{ where } g(y^{\text{obs}}) = 0$$

- It follows that $\kappa(\theta)$ is a normalizing “constant”:

$$\kappa(\theta) = \sum_{\text{all possible graphs } z} \exp\{\theta^t g(z)\}.$$

- Replacing $g(y)$ by $g(y) - g(y^{\text{obs}})$ leaves $P_{\theta}(Y = y)$ unchanged; thus, we “recenter” $g(y)$ so that $g(y^{\text{obs}}) = 0$.

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}, \text{ where } g(y^{\text{obs}}) = 0.$$

- The idea behind maximum likelihood is this: Define $\hat{\theta}$ to be the value of θ such that $P_{\theta}(Y = y^{\text{obs}})$ is as large as possible.

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}, \text{ where } g(y^{\text{obs}}) = 0.$$

- The idea behind maximum likelihood is this: Define $\hat{\theta}$ to be the value of θ such that $P_{\theta}(Y = y^{\text{obs}})$ is as large as possible.
- In practice, we maximize the natural logarithm of $P_{\theta}(Y = y^{\text{obs}})$, which equals $-\log \kappa(\theta)$.

Fact:

The method of maximum likelihood is one of the most well-studied topics in the entire field of statistics, and maximum likelihood estimators generally are known to have a lot of nice properties.

Fact:

The method of maximum likelihood is one of the most well-studied topics in the entire field of statistics, and maximum likelihood estimators generally are known to have a lot of nice properties.

Warning: The fact that $P_{\hat{\theta}}(Y = y^{\text{obs}})$ is as large as possible in this model class does NOT mean that y^{obs} is particularly likely relative to other networks!

(The model class itself might be inappropriate.)

A nifty fact regarding the MLE $\hat{\theta}$

- Because we're dealing with an exponential family of models,

$$E_{\hat{\theta}} g(Y) = g(y^{\text{obs}})$$

and no other value of θ has this property.

- In words:

The MLE gives the unique model in the model class under which the mean value of the vector of statistics equals its observed value.

A nifty fact regarding the MLE $\hat{\theta}$

- Because we're dealing with an exponential family of models,

$$E_{\hat{\theta}} g(Y) = g(y^{\text{obs}})$$

and no other value of θ has this property.

- In words:

The MLE gives the unique model in the model class under which the mean value of the vector of statistics equals its observed value.

- This fact may even be exploited to compute $\hat{\theta}$.

- 1 Maximum likelihood estimation
- 2 Approximating the MLE**
- 3 Simulating random networks via MCMC
- 4 Maximum pseudolikelihood
- 5 Goodness of fit

It is difficult to find the MLE

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}, \text{ where } g(y^{\text{obs}}) = 0$$

- Recall: We wish to maximize

$$\ell(\theta) = -\log \kappa(\theta) = \log \sum_{\substack{\text{all possible} \\ \text{graphs } z}} \exp\{\theta^t s(z)\},$$

a function that is called the *loglikelihood*.

It is difficult to find the MLE

The model class:

$$P_{\theta}(Y = y) = \frac{\exp\{\theta^t g(y)\}}{\kappa(\theta)}, \text{ where } g(y^{\text{obs}}) = 0$$

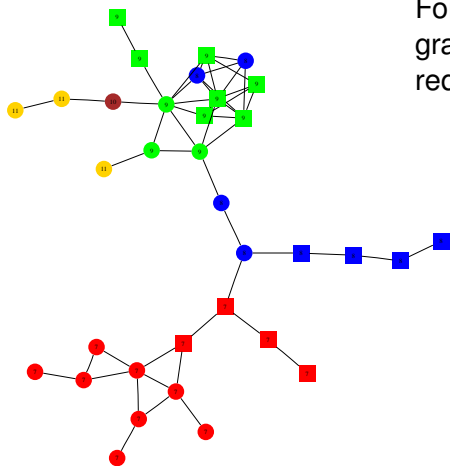
- Recall: We wish to maximize

$$\ell(\theta) = -\log \kappa(\theta) = \log \sum_{\substack{\text{all possible} \\ \text{graphs } z}} \exp\{\theta^t s(z)\},$$

a function that is called the *loglikelihood*.

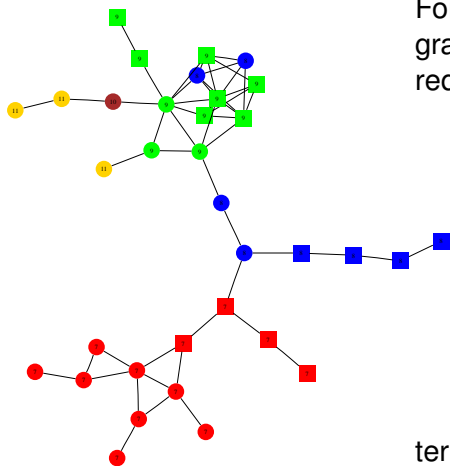
- As Martina pointed out, $\kappa(\theta)$ is a nasty beast: Merely evaluating (let alone maximizing) $\ell(\theta)$ is somewhat computationally burdensome. . .

How burdensome?



For this undirected, 34-node graph, computing $\ell(\theta)$ directly requires summation of

How burdensome?



For this undirected, 34-node graph, computing $\ell(\theta)$ directly requires summation of

7,547,924,849,643,082,704,483,
109,161,976,537,781,833,842,
440,832,880,856,752,412,600,
491,248,324,784,297,704,172,
253,450,355,317,535,082,936,
750,061,527,689,799,541,169,
259,849,585,265,122,868,502,
865,392,087,298,790,653,952

terms.

The germ of an idea

- Suppose we fix θ_0 . A bit of algebra shows that

$$\log E_{\theta_0} [\exp \{(\theta_0 - \theta)^t g(Y)\}] = \ell(\theta) - \ell(\theta_0).$$

The germ of an idea

- Suppose we fix θ_0 . A bit of algebra shows that

$$\log E_{\theta_0} [\exp \{(\theta_0 - \theta)^t g(Y)\}] = \ell(\theta) - \ell(\theta_0).$$

- More to the point,

$$\log E_{\theta_0} [\textit{blah blah } Y \textit{ blah}] = \ell(\theta) - \ell(\theta_0).$$

The germ of an idea

- Suppose we fix θ_0 . A bit of algebra shows that

$$\log E_{\theta_0} [\exp \{(\theta_0 - \theta)^t g(Y)\}] = \ell(\theta) - \ell(\theta_0).$$

- More to the point,

$$\log E_{\theta_0} [\textit{blah blah } Y \textit{ blah}] = \ell(\theta) - \ell(\theta_0).$$

- Thus, $\ell(\theta) - \ell(\theta_0)$ involves an expected value (a mean).

Law of Large Numbers to the Rescue!

The LLN suggests that we approximate an unknown population mean by a sample mean.

Law of Large Numbers to the Rescue!

The LOLN suggests that we approximate an unknown population mean by a sample mean.

Thus,

$$\begin{aligned}\ell(\theta) - \ell(\theta_0) &= \log E_{\theta_0} (\exp \{(\theta_0 - \theta)^t g(Y)\}) \\ &\approx \log \frac{1}{m} \sum_{i=1}^m \exp \{(\theta_0 - \theta)^t g(Y_i)\},\end{aligned}$$

where Y_1, Y_2, \dots, Y_m is a random sample of networks from the distribution defined by the ERGM with parameter θ_0 .

$$\begin{aligned}\ell(\theta) - \ell(\theta_0) &\approx \log \frac{1}{m} \sum_{i=1}^m \exp \{(\theta_0 - \theta)^t \mathbf{s}(Y_i)\} \\ &= \log \frac{1}{m} \sum_{i=1}^m (\text{blah } \theta \text{ blah } Y_i \text{ blah}).\end{aligned}$$

$$\begin{aligned}\ell(\theta) - \ell(\theta_0) &\approx \log \frac{1}{m} \sum_{i=1}^m \exp \{ (\theta_0 - \theta)^t s(Y_i) \} \\ &= \log \frac{1}{m} \sum_{i=1}^m (\text{blah } \theta \text{ blah } Y_i \text{ blah}).\end{aligned}$$

- Given a random sample of networks from P_{θ_0} , we can thus approximate (and subsequently maximize) the loglikelihood shifted by a constant.

$$\begin{aligned}\ell(\theta) - \ell(\theta_0) &\approx \log \frac{1}{m} \sum_{i=1}^m \exp \{(\theta_0 - \theta)^t s(Y_i)\} \\ &= \log \frac{1}{m} \sum_{i=1}^m (\text{blah } \theta \text{ blah } Y_i \text{ blah}).\end{aligned}$$

- Given a random sample of networks from P_{θ_0} , we can thus approximate (and subsequently maximize) the loglikelihood shifted by a constant.
- We can approximate the MLE if we can simulate random networks!

- 1 Maximum likelihood estimation
- 2 Approximating the MLE
- 3 Simulating random networks via MCMC**
- 4 Maximum pseudolikelihood
- 5 Goodness of fit

The need for simulated random networks

The ability to simulate random networks from an ERGM in which θ_0 is a known fixed parameter vector allows us to:

- Approximate the MLE

The need for simulated random networks

The ability to simulate random networks from an ERGM in which θ_0 is a known fixed parameter vector allows us to:

- Approximate the MLE
- Find out what networks generated by the ERGM “look like”

The need for simulated random networks

The ability to simulate random networks from an ERGM in which θ_0 is a known fixed parameter vector allows us to:

- Approximate the MLE
- Find out what networks generated by the ERGM “look like”
- Observe the dynamic evolution of a network

The need for simulated random networks

The ability to simulate random networks from an ERGM in which θ_0 is a known fixed parameter vector allows us to:

- Approximate the MLE
- Find out what networks generated by the ERGM “look like”
- Observe the dynamic evolution of a network

To discuss algorithms used for simulating random networks, we must review the notion of the conditional log-odds of an edge as determined by an ERGM.

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $y_{ij} = 0$ or 1 , depending on whether there is an edge
- y_{ij}^c denotes the status of all pairs in y other than (i, j)
- y_{ij}^+ denotes the same network as y but with $y_{ij} = 1$
- y_{ij}^- denotes the same network as y but with $y_{ij} = 0$

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $y_{ij} = 0$ or 1 , depending on whether there is an edge
- y_{ij}^c denotes the status of all pairs in y other than (i, j)
- y_{ij}^+ denotes the same network as y but with $y_{ij} = 1$
- y_{ij}^- denotes the same network as y but with $y_{ij} = 0$

Conditional on $Y_{ij}^c = y_{ij}^c$, Y has only two possible states, depending on whether $Y_{ij} = 0$ or $Y_{ij} = 1$.

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $y_{ij} = 0$ or 1 , depending on whether there is an edge
- y_{ij}^c denotes the status of all pairs in y other than (i, j)
- y_{ij}^+ denotes the same network as y but with $y_{ij} = 1$
- y_{ij}^- denotes the same network as y but with $y_{ij} = 0$

Conditional on $Y_{ij}^c = y_{ij}^c$, Y has only two possible states, depending on whether $Y_{ij} = 0$ or $Y_{ij} = 1$.
Let's calculate the ratio of the two respective probabilities.

[We'll use $P_\theta(Y = y) = \exp\{\theta^t g(y)\} / \kappa(\theta)$.]

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $y_{ij} = 0$ or 1 , depending on whether there is an edge
- y_{ij}^c denotes the status of all pairs in y other than (i, j)
- y_{ij}^+ denotes the same network as y but with $y_{ij} = 1$
- y_{ij}^- denotes the same network as y but with $y_{ij} = 0$

$$\frac{P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} = 0 | Y_{ij}^c = y_{ij}^c)} = \frac{\exp\{\theta^t g(y_{ij}^+)\}}{\exp\{\theta^t g(y_{ij}^-)\}}$$

A lot of cancellation happened on the right hand side!

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $y_{ij} = 0$ or 1 , depending on whether there is an edge
- y_{ij}^c denotes the status of all pairs in y other than (i, j)
- y_{ij}^+ denotes the same network as y but with $y_{ij} = 1$
- y_{ij}^- denotes the same network as y but with $y_{ij} = 0$

$$\frac{P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} = 0 | Y_{ij}^c = y_{ij}^c)} = \exp\{\theta^t [g(y_{ij}^+) - g(y_{ij}^-)]\}$$

A lot of cancellation happened on the right hand side!

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $y_{ij} = 0$ or 1 , depending on whether there is an edge
- y_{ij}^c denotes the status of all pairs in y other than (i, j)
- y_{ij}^+ denotes the same network as y but with $y_{ij} = 1$
- y_{ij}^- denotes the same network as y but with $y_{ij} = 0$

$$\log \frac{P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} = 0 | Y_{ij}^c = y_{ij}^c)} = \theta^t [g(y_{ij}^+) - g(y_{ij}^-)]$$

Conditional log-odds of an edge

Notation: For a network y and a pair (i, j) of nodes,

- $\Delta(g(y))_{ij}$ denotes the vector of change statistics,

$$\Delta(g(y))_{ij} = g(y_{ij}^+) - g(y_{ij}^-).$$

So $\Delta(g(y))_{ij}$ is the conditional log-odds of edge (i, j) .

$$\log \frac{P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} = 0 | Y_{ij}^c = y_{ij}^c)} = \theta^t \Delta(g(y))_{ij}$$

Simulating random networks

Goal:

Simulate random network(s) Y from an ERGM.

Note: There is no model, only a model class, unless we have a specific parameter vector θ ; we'll need to fix an θ somehow.

Simulating random networks

Goal:

Simulate random network(s) Y from an ERGM.

Note: There is no model, only a model class, unless we have a specific parameter vector θ ; we'll need to fix an θ somehow.

We'll discuss one way to achieve the goal, called a Metropolis algorithm.

The Metropolis algorithm is one of a broad class of algorithms called Markov Chain Monte Carlo (MCMC) algorithms.

Obtaining samples via Markov Chain Monte Carlo

Whence the name MCMC?

- Markov Chain: A sequence Y_1, Y_2, \dots where the $(i + 1)$ th network is randomly generated based on the i th network.
- Monte Carlo: The computational implementation of the "randomly generated" part.

MCMC Idea for simulating networks:

Simulate a carefully designed Markov chain on the sample space of networks for a while. When we stop it, we'll have our random network.

Metropolis algorithm

- First, select a pair of nodes at random, say (i, j) .

Metropolis algorithm

- First, select a pair of nodes at random, say (i, j) .
- Calculate the ratio

$$\begin{aligned}\pi &= \frac{P(Y_{ij} \text{ changes} | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} \text{ does not change} | Y_{ij}^c = y_{ij}^c)} \\ &= \exp\{\pm\theta_0^t \Delta(g(y))_{ij}\}\end{aligned}$$

Metropolis algorithm

- First, select a pair of nodes at random, say (i, j) .
- Calculate the ratio

$$\begin{aligned}\pi &= \frac{P(Y_{ij} \text{ changes} | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} \text{ does not change} | Y_{ij}^c = y_{ij}^c)} \\ &= \exp\{\pm\theta_0^t \Delta(g(y))_{ij}\}\end{aligned}$$

- Accept the change of Y_{ij} with probability $\min\{1, \pi\}$ (i.e., always make the change if $\pi \geq 1$; otherwise, only make the change sometimes).

Metropolis algorithm

- First, select a pair of nodes at random, say (i, j) .
- Calculate the ratio

$$\begin{aligned}\pi &= \frac{P(Y_{ij} \text{ changes} | Y_{ij}^c = y_{ij}^c)}{P(Y_{ij} \text{ does not change} | Y_{ij}^c = y_{ij}^c)} \\ &= \exp\{\pm\theta_0^t \Delta(g(y))_{ij}\}\end{aligned}$$

- Accept the change of Y_{ij} with probability $\min\{1, \pi\}$ (i.e., always make the change if $\pi \geq 1$; otherwise, only make the change sometimes).

Note: The values of $g(y_{ij}^+)$ and $g(y_{ij}^-)$ are never needed; only the difference $\Delta(g(y))_{ij}$ matters.

How should θ_0 be chosen?

- Theoretically, the estimated value of $\ell(\theta) - \ell(\theta_0)$ converges to the true value as the size of the MCMC sample increases, regardless of the value of θ_0 .

How should θ_0 be chosen?

- Theoretically, the estimated value of $\ell(\theta) - \ell(\theta_0)$ converges to the true value as the size of the MCMC sample increases, regardless of the value of θ_0 .
- However, in practice this convergence can be agonizingly slow, especially if θ_0 is not chosen close to the maximizer of the likelihood.

How should θ_0 be chosen?

- Theoretically, the estimated value of $\ell(\theta) - \ell(\theta_0)$ converges to the true value as the size of the MCMC sample increases, regardless of the value of θ_0 .
- However, in practice this convergence can be agonizingly slow, especially if θ_0 is not chosen close to the maximizer of the likelihood.
- A choice that sometimes works is the MPLE (maximum pseudolikelihood estimate)

- 1 Maximum likelihood estimation
- 2 Approximating the MLE
- 3 Simulating random networks via MCMC
- 4 Maximum pseudolikelihood**
- 5 Goodness of fit

Maximum Pseudolikelihood: Intuition

- What if we assume that there is no dependence (or very weak dependence) among the Y_{ij} ?

Maximum Pseudolikelihood: Intuition

- What if we assume that there is no dependence (or very weak dependence) among the Y_{ij} ?
- In other words, what if we approximate the marginal $P(Y_{ij} = 1)$ by the conditional $P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)$?

Maximum Pseudolikelihood: Intuition

- What if we assume that there is no dependence (or very weak dependence) among the Y_{ij} ?
- In other words, what if we approximate the marginal $P(Y_{ij} = 1)$ by the conditional $P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)$?
- Then the Y_{ij} are independent with

$$\log \frac{P(Y_{ij} = 1)}{P(Y_{ij} = 0)} = \theta^t \Delta(g(y^{\text{obs}}))_{ij},$$

so we obtain an estimate of θ using straightforward logistic regression.

Maximum Pseudolikelihood: Intuition

- What if we assume that there is no dependence (or very weak dependence) among the Y_{ij} ?
- In other words, what if we approximate the marginal $P(Y_{ij} = 1)$ by the conditional $P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)$?
- Then the Y_{ij} are independent with

$$\log \frac{P(Y_{ij} = 1)}{P(Y_{ij} = 0)} = \theta^t \Delta(g(y^{\text{obs}}))_{ij},$$

so we obtain an estimate of θ using straightforward logistic regression.

- Result: The **maximum pseudolikelihood estimate**.

Maximum Pseudolikelihood: Intuition

- What if we assume that there is no dependence (or very weak dependence) among the Y_{ij} ?
- In other words, what if we approximate the marginal $P(Y_{ij} = 1)$ by the conditional $P(Y_{ij} = 1 | Y_{ij}^c = y_{ij}^c)$?
- Then the Y_{ij} are independent with

$$\log \frac{P(Y_{ij} = 1)}{P(Y_{ij} = 0)} = \theta^t \Delta(g(y^{\text{obs}}))_{ij},$$

so we obtain an estimate of θ using straightforward logistic regression.

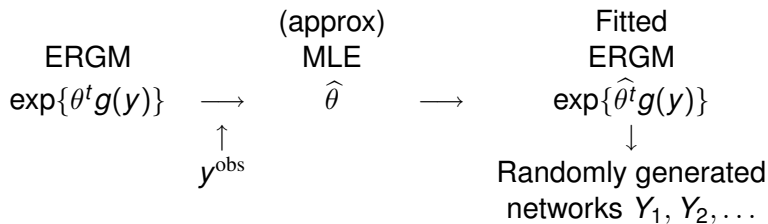
- Result: The **maximum pseudolikelihood estimate**.
- For independence models, MPLE = MLE!

Warnings about MPLE

Unfortunately, little is known about the quality of MPL estimates in general, but we do know some ways in which they can be misleading.

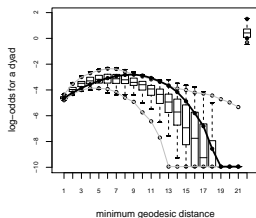
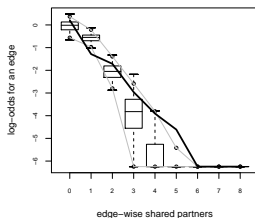
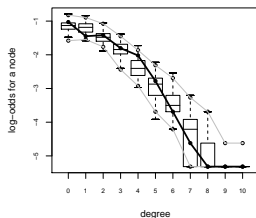
- If the model is bad, you'll get nice-looking MPLE results that do not reveal the problem.
- If the model is good, in many cases the MPLE looks “close” to the MLE; however, “close” can be deceiving, since small changes in θ can sometimes lead to large differences in the behavior of randomly generated networks.

- 1 Maximum likelihood estimation
- 2 Approximating the MLE
- 3 Simulating random networks via MCMC
- 4 Maximum pseudolikelihood
- 5 Goodness of fit**



Question: How does the observed network “look” as a representative of the sample of generated networks?

Goodness of fit check: An example



- Garry Robbins will next discuss goodness of fit (among other topics) in more detail.
- Steve Goodreau will provide several examples in tomorrow's `statnet` workshop.