Big Problems for Small Networks: Statistical Analysis of Small Networks and Team Performance

George G Vega Yon Kayla de la Haye

Department of Preventive Medicine

SONIC Speaker Series
Northwestern University, IL
March 20, 2019

Acknowledgements

This material is based upon work support by, or in part by, the U.S. Army Research Laboratory and the U.S. Army Research Office under grant number W911NF-15-1-0577

Computation for the work described in this paper was supported by the University of Southern California's Center for High-Performance Computing (https://hpcc.usc.edu).

We thank members of our MURI research team, USC's Center for Applied Network Analysis, Andrew Slaughter, and attendees of the NASN 2018 conference for their comments.

Research Problem

In the context of network science of small teams:

Research Problem

In the context of network science of small teams:

What characterizes the social networks that emerge from small teams?

Research Problem

In the context of network science of small teams:

What characterizes the social networks that emerge from small teams?

Is there any association between how team networks are structured and their performance?

Research Problem (cont'd)

We are trying to answer these two questions with the following experimental data:

Research Problem (cont'd)

We are trying to answer these two questions with the following experimental data:

- 42 teams typically made of 4 to 5 mixed-gender teammates.

Research Problem (cont'd)

We are trying to answer these two questions with the following experimental data:

- 42 teams typically made of 4 to 5 mixed-gender teammates.
- Each team completed 1 hour of group tasks, in particular, a set of tasks to measure collective intelligence developed by MIT (Kim et al. 2017).

Research Problem (cont'd)

We are trying to answer these two questions with the following experimental data:

- 42 teams typically made of 4 to 5 mixed-gender teammates.
- Each team completed 1 hour of group tasks, in particular, a set of tasks to measure collective intelligence developed by MIT (Kim et al. 2017).
- We surveyed team members to capture information regarding socio-demographics **and**:

Research Problem (cont'd)

We are trying to answer these two questions with the following experimental data:

- 42 teams typically made of 4 to 5 mixed-gender teammates.
- Each team completed 1 hour of group tasks, in particular, a set of tasks to measure collective intelligence developed by MIT (Kim et al. 2017).
- We surveyed team members to capture information regarding socio-demographics **and**:
  - Social Intelligence (SI domains): Social Perception (measured by RME), Social Accomodation, Social Gregariousness, and Social Awareness

Research Problem (cont'd)

We are trying to answer these two questions with the following experimental data:

- 42 teams typically made of 4 to 5 mixed-gender teammates.
- Each team completed 1 hour of group tasks, in particular, a set of tasks to measure collective intelligence developed by MIT (Kim et al. 2017).
- We surveyed team members to capture information regarding socio-demographics **and**:
  - Social Intelligence (SI domains): Social Perception (measured by RME), Social Accomodation, Social Gregariousness, and Social Awareness
  - Social Networks: Advice Seeking, Leadership, Influence (among others).

Contents

Part I: Network Structure

Exponential Random Graph Models (ERGMs)

Figure 1: Friendship network of a UK university faculty. Source: **igraphdata** R package (Csardi, 2015). Figure drawn using the R package **netplot** (yours truly, https://github.com/usccana/netplot)

Exponential Random Graph Models (ERGMs)

Figure 1: Friendship network of a UK university faculty. Source: **igraphdata** R package (Csardi, 2015). Figure drawn using the R package **netplot** (yours truly, https://github.com/usccana/netplot)

How can we explain what we see here?

ERGMs. . . the *lingua franca* of SNA

- Seeks to answer the question: What local social structures gave origin to a given observed graph?

ERGMs... the *lingua franca* of SNA

- Seeks to answer the question: What local social structures gave origin to a given observed graph?
- The model is centered around a vector of **sufficient statistics** $s()$, and is operationalized as:

$$\Pr(\mathbf{G} = \mathbf{g} \mid \theta, \mathbf{X}) = \frac{\exp\{\theta^{\mathbf{t}} s(\mathbf{g}, \mathbf{X})\}}{\kappa(\theta, \mathbf{X})}, \quad \forall \mathbf{g} \in \mathcal{G} \tag{1}$$

Where $\kappa(\theta, \mathbf{X})$ is the normalizing constant and equals $\sum_{\mathbf{g}' \in \mathcal{G}} \exp\{\theta^{\mathbf{t}} s(\mathbf{g}', \mathbf{X})\}$.

ERGMs... the *lingua franca* of SNA

- Seeks to answer the question: What local social structures gave origin to a given observed graph?
- The model is centered around a vector of **sufficient statistics** $s()$, and is operationalized as:

$$\Pr\left(\mathbf{G} = \mathbf{g} \mid \theta, \mathbf{X}\right) = \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}, \mathbf{X}\right)\right\}}{\kappa\left(\theta, \mathbf{X}\right)}, \quad \forall \mathbf{g} \in \mathcal{G} \tag{1}$$

  Where $\kappa\left(\theta, \mathbf{X}\right)$ is the normalizing constant and equals $\sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}', \mathbf{X}\right)\right\}$.
- The set of sufficient statistics reflects social and psychological mechanisms that are hypothesized to drive the network structure. Figure 2 shows some examples of values in $s()$.

ERGMs. . . the *lingua franca* of SNA

- Seeks to answer the question: What local social structures gave origin to a given observed graph?
- The model is centered around a vector of **sufficient statistics** $s\left(\right)$, and is operationalized as:

$$\Pr\left(\mathbf{G} = \mathbf{g} \mid \theta, \mathbf{X}\right) = \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}, \mathbf{X}\right)\right\}}{\kappa\left(\theta, \mathbf{X}\right)}, \quad \forall \mathbf{g} \in \mathcal{G} \tag{1}$$

  Where $\kappa\left(\theta, \mathbf{X}\right)$ is the normalizing constant and equals $\sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}', \mathbf{X}\right)\right\}$.
- The set of sufficient statistics reflects social and psychological mechanisms that are hypothesized to drive the network structure. Figure 2 shows some examples of values in $s\left(\right)$.
- In the case of directed networks, $\mathcal{G}$ has $2^{n(n-1)}$ terms.

ERGMs... the *lingua franca* of SNA

- Seeks to answer the question: What local social structures gave origin to a given observed graph?
- The model is centered around a vector of **sufficient statistics** $s()$, and is operationalized as:

$$\Pr(\mathbf{G} = \mathbf{g} \mid \theta, \mathbf{X}) = \frac{\exp\left\{\theta^{\mathbf{t}} s(\mathbf{g}, \mathbf{X})\right\}}{\kappa(\theta, \mathbf{X})}, \quad \forall \mathbf{g} \in \mathcal{G} \tag{1}$$

  Where $\kappa(\theta, \mathbf{X})$ is the normalizing constant and equals $\sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s(\mathbf{g}', \mathbf{X})\right\}$.
- The set of sufficient statistics reflects social and psychological mechanisms that are hypothesized to drive the network structure. Figure 2 shows some examples of values in $s()$.
- In the case of directed networks, $\mathcal{G}$ has $2^{n(n-1)}$ terms.
- See Wasserman, Pattison, Robins, Snijders, Handcock and others.

Structures

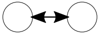
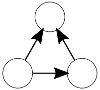
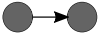
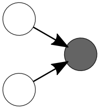
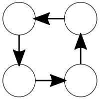
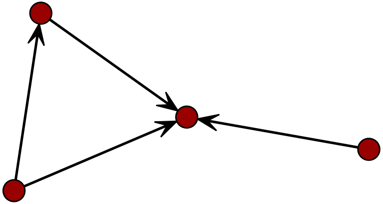
Representation	Description
	Mutual Ties (Reciprocity) $\sum_{i \neq j} y_{ij} y_{ji}$
	Transitive Triad (Balance) $\sum_{i \neq j \neq k} y_{ij} y_{jk} y_{ik}$
	Homophily $\sum_{i \neq j} y_{ij} \mathbf{1}\left(x_i = x_j\right)$
	Covariate Effect for Incoming Ties $\sum_{i \neq j} y_{ij} x_j$
	Four Cycle $\sum_{i \neq j \neq k \neq l} y_{ij} y_{jk} y_{kl} y_{li}$

Figure 2: Besides of the common edge count statistic (number of ties in a graph), ERGMs allow measuring other more complex structures that can be captured as sufficient statistics.

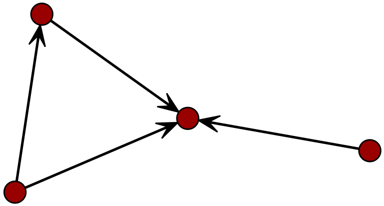
Example of model

In this network



Example of model

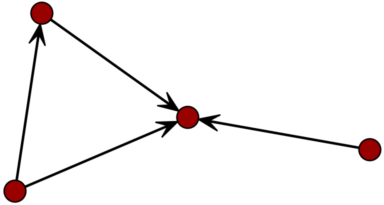
In this network



We see 4 **edges**, 1 **transitive triad** and **no mutual ties**.

Example of model

In this network



We see 4 **edges**, 1 **transitive triad** and **no mutual ties**.

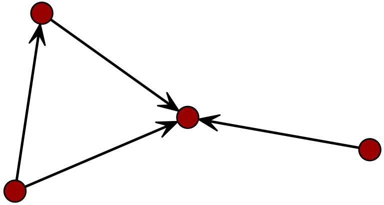
The probability function of this model would be

$$\Pr\left(\mathbf{G} = \mathbf{g} \mid \theta\right) = \frac{\exp\left\{4\theta_{edges} + \theta_{ttriads} + 0\theta_{mutual}\right\}}{\sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}'\right)\right\}}$$

with $\theta = \left[\theta_{edges} \quad \theta_{ttriads} \quad \theta_{mutual}\right]^{\mathbf{t}}$

Example of model

In this network



We see 4 **edges**, 1 **transitive triad** and **no mutual ties**.

The probability function of this model would be

$$\Pr\left(\mathbf{G}=\mathbf{g}\mid\theta\right)=\frac{\exp\left\{4\theta_{edges}+\theta_{ttriads}+0\theta_{mutual}\right\}}{\sum_{\mathbf{g}'\in\mathcal{G}}\exp\left\{\theta^{\mathbf{t}}s\left(\mathbf{g}'\right)\right\}}$$

with $\theta=\left[\theta_{edges}\quad\theta_{ttriads}\quad\theta_{mutual}\right]^{\mathbf{t}}$

This model has **MLE parameter estimates** of -0.19 (low density), 0.27 (high chance of ttriads), and -9.75 (low chance of mutuality) for the parameters `edges`, `ttriads`, and `mutual` respectively.

Estimation of ERGMs

- Calculating of the normalizing constant in (1), $\kappa = \sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}', \mathbf{X}\right)\right\}$, makes ERGMs difficult to estimate.

Estimation of ERGMs

- Calculating of the normalizing constant in (1), $\kappa = \sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}', \mathbf{X}\right)\right\}$, makes ERGMs difficult to estimate.
- For this reason, statistical methods have focused on avoiding the direct calculation of κ; most modern methods for estimating ERGMs rely on MCMC.

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Generate a large sample of graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \;\middle|\; \hat{\theta}_0, \mathbf{X}\right)$ using MCMC

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Generate a large sample of graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \,\middle|\, \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sample of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat\theta_0$
2. While the algorithm doesn't converge, do:
   a. Generate a large sample of graphs from $\Pr\left(\mathbf{G}=\mathbf{g}\,\middle|\,\hat\theta_0,\mathbf{X}\right)$ using MCMC
   b. Use the simulated sample of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Generate a large sample of graphs from $\Pr\left(\mathbf{G}=\mathbf{g} \mid \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sample of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

Once it converges, the last value of $\hat{\theta}_0$ is the MCMC-MLE estimates.

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Generate a large sample of graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \;\middle|\; \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sample of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

Once it converges, the last value of $\hat{\theta}_0$ is the MCMC-MLE estimates.The variance approximation is another story.

- While significant advances have been made in the area, simulation based models can suffer from **model degeneracy** (Handcock 2003).

- While significant advances have been made in the area, simulation based models can suffer from **model degeneracy** (Handcock 2003).
- This occurs when the model parameters *live in a degenerate region*. This impacts directly the MCMC process yielding poor mixing

- While significant advances have been made in the area, simulation based models can suffer from **model degeneracy** (Handcock 2003).
- This occurs when the model parameters *live in a degenerate region*. This impacts directly the MCMC process yielding poor mixing

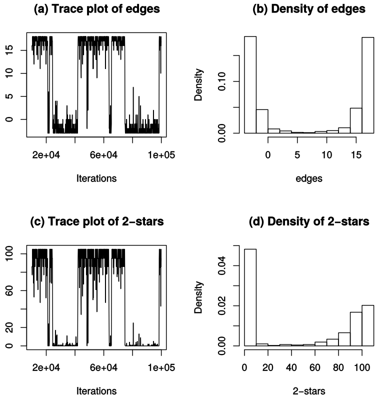


Figure 3: Model degeneracy. Figure 5 in Handcock (2003)

- While significant advances have been made in the area, simulation based models can suffer from **model degeneracy** (Handcock 2003).
- This occurs when the model parameters *live in a degenerate region*. This impacts directly the MCMC process yielding poor mixing

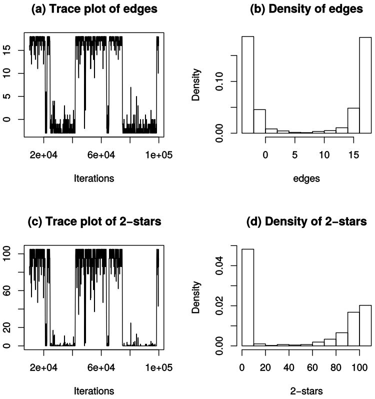
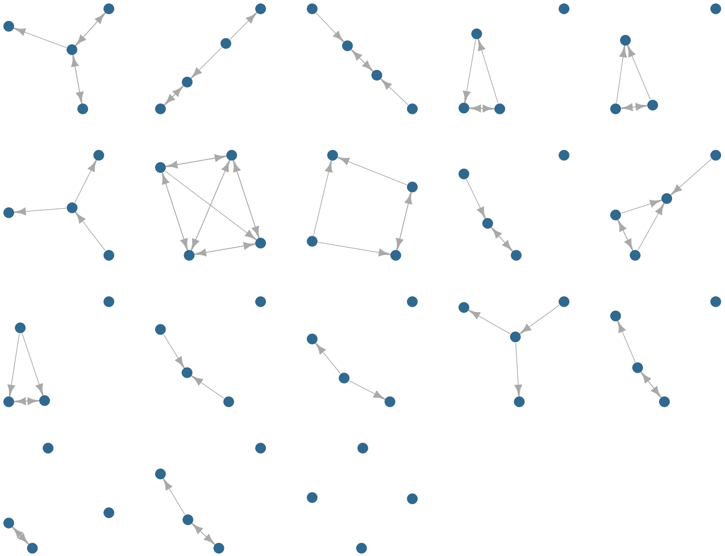


Figure 3: Model degeneracy. Figure 5 in Handcock (2003)

- Model degeneracy is particularly problematic with small networks. . . (says anyone who has tried to fit one).

ERGMs for Small Networks



ERGMs for Small Networks (cont'd)

- In the case of small networks (e.g. at most 6 nodes), the calculation of κ becomes computationally feasible.

ERGMs for Small Networks (cont'd)

- In the case of small networks (e.g. at most 6 nodes), the calculation of κ becomes computationally feasible.
- This allows direct calculation of (1), **avoiding the need for simulations** and allowing us to obtain Maximum Likelihood Estimates using *standard* optimization techniques.

ERGMs for Small Networks (cont'd)

- In the case of small networks (e.g. at most 6 nodes), the calculation of κ becomes computationally feasible.
- This allows direct calculation of (1), **avoiding the need for simulations** and allowing us to obtain Maximum Likelihood Estimates using *standard* optimization techniques.
- In addition, most of the time samples of small networks include multiple of them, e.g.: Families, Small teams (like our data), Ego-nets, etc.

ERGMs for Small Networks (cont'd)

- In the case of small networks (e.g. at most 6 nodes), the calculation of κ becomes computationally feasible.
- This allows direct calculation of (1), **avoiding the need for simulations** and allowing us to obtain Maximum Likelihood Estimates using *standard* optimization techniques.
- In addition, most of the time samples of small networks include multiple of them, e.g.: Families, Small teams (like our data), Ego-nets, etc.
- This makes pooled ERGM estimates a natural way of modeling the data:/pause

$$\Pr\left(\mathbf{G}_1 = \mathbf{g}_1, \dots, \mathbf{G}_p = \mathbf{g}_p \mid \theta, \mathbf{X}_1, \dots, \mathbf{X}_p\right) = \prod_p \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}_p, \mathbf{X}_p\right)\right\}}{\kappa_p\left(\theta, \mathbf{X}_p\right)}$$

ERGMs for Small Networks (cont'd)

- In the case of small networks (e.g. at most 6 nodes), the calculation of κ becomes computationally feasible.
- This allows direct calculation of (1), **avoiding the need for simulations** and allowing us to obtain Maximum Likelihood Estimates using *standard* optimization techniques.
- In addition, most of the time samples of small networks include multiple of them, e.g.: Families, Small teams (like our data), Ego-nets, etc.
- This makes pooled ERGM estimates a natural way of modeling the data:/pause

$$\mathrm{Pr}\left(\mathbf{G}_1 = \mathbf{g}_1, \dots, \mathbf{G}_p = \mathbf{g}_p \mid \theta, \mathbf{X}_1, \dots, \mathbf{X}_p\right) = \prod_p \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}_p, \mathbf{X}_p\right)\right\}}{\kappa_p\left(\theta, \mathbf{X}_p\right)}$$

How different is this from the “normal” way to fit ERGMs?

Estimation of ERGMs (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \;\middle|\; \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

Estimation of ERGMitos (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:

   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \;\middle|\; \hat{\theta}_0, \mathbf{X}\right)$ using MCMC

   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.

   c. next iteration

Estimation of ERGMitos (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \,\middle|\, \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

By skipping the MCMC part we:

Estimation of ERGMitos (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \;\middle|\; \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

By skipping the MCMC part we:

1. are able to get MLE estimates directly,

Estimation of ERGMitos (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \,\middle|\, \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

By skipping the MCMC part we:

1. are able to get MLE estimates directly,
2. avoid the degeneracy problem latent in MCMC, and

Estimation of ERGMitos (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \,\middle|\, \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

By skipping the MCMC part we:

1. are able to get MLE estimates directly,
2. avoid the degeneracy problem latent in MCMC, and
3. obtain more accurate (exact) estimates faster.

Estimation of ERGMitos (cont'd)

Description of the MCMC-MLE algorithm (Geyer and Thompson 1992)

1. Make an initial guess of the model parameters $\hat{\theta}_0$
2. While the algorithm doesn't converge, do:
   a. Simulate B graphs from $\Pr\left(\mathbf{G} = \mathbf{g} \,\middle|\, \hat{\theta}_0, \mathbf{X}\right)$ using MCMC
   b. Use the simulated sequence of graphs to approximate the likelihood function. And with this approximation update the parameter θ_0 using a Newton-Raphson step.
   c. next iteration

By skipping the MCMC part we:

1. are able to get MLE estimates directly,
2. avoid the degeneracy problem latent in MCMC, and
3. obtain more accurate (exact) estimates faster.

We have implemented this and more in the `ergmito` R package

Sidetrack...

ito, ita: From the latin -*īttus*. suffix in Spanish used to denote small or affection.
e.g.:

*¡Que lindo ese perr**ito**! / What a beautiful little dog!*
*¿Me darias una tac**ita** de azúcar? / Would you give me a small cup of sugar?*

Sidetrack...

ito, ita: From the latin -*īttus*. suffix in Spanish used to denote small or affection. e.g.:

*¡Que lindo ese perr**ito**! / What a beautiful little dog!*
*¿Me darias una tac**ita** de azúcar? / Would you give me a small cup of sugar?*

Special thanks to George Barnett who proposed the name during the 2018 NASN!

Features of `ergmito`

This (lifecycle experimental) R package has the following features

Features of `ergmito`

This (lifecycle experimental) R package has the following features

- Built on top of **statnet**'s `ergm` R package (Hunter et al. 2008; Handcock et al. 2018).

[1] A directed graph of size 6 has a support set with $2^{6\times(6-1)} = 1,073,741,824$ elements.

Features of `ergmito`

This (lifecycle experimental) R package has the following features

- Built on top of **statnet**'s `ergm` R package (Hunter et al. 2008; Handcock et al. 2018).
- Allows estimating ERGMs for small networks (less than 7 and perhaps 6)[1] via MLE.

[1] A directed graph of size 6 has a support set with $2^{6\times(6-1)} = 1,073,741,824$ elements.

Features of `ergmito`

This (lifecycle experimental) R package has the following features

- Built on top of **statnet**'s `ergm` R package (Hunter et al. 2008; Handcock et al. 2018).
- Allows estimating ERGMs for small networks (less than 7 and perhaps 6)[1] via MLE.
- Implements pooled ERGM models.

[1] A directed graph of size 6 has a support set with $2^{6\times(6-1)} = 1,073,741,824$ elements.

Features of `ergmito`

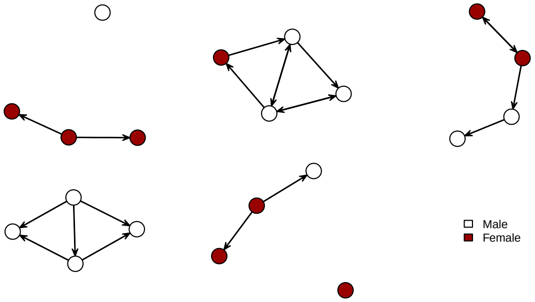
This (lifecycle experimental) R package has the following features

- Built on top of **statnet**'s `ergm` R package (Hunter et al. 2008; Handcock et al. 2018).
- Allows estimating ERGMs for small networks (less than 7 and perhaps 6)[1] via MLE.
- Implements pooled ERGM models.
- Includes a simulation function for efficiently drawing samples of small networks, and by **effiently** we mean **fast**.

[1] A directed graph of size 6 has a support set with $2^{6\times(6-1)} = 1,073,741,824$ elements.

`ergmito` example

```r
library(ergmito)
data(fivenets, package = "ergmito")
```



```r
# Looking at one of the five networks
fivenets[[1]]
```

```
##  Network attributes:
##   vertices = 4
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 2
##     missing edges= 0
##     non-missing edges= 2
##
##  Vertex attribute names:
##     female name
##
## No edge attributes
```

```r
# Looking at one of the five networks
fivenets[[1]]
```

```
##  Network attributes:
##   vertices = 4
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 2
##     missing edges= 0
##     non-missing edges= 2
##
##  Vertex attribute names:
##     female name
##
## No edge attributes
```

So how can we fit this model?

ergmito example (cont'd)

The same as you would do with the ergm package:

```
(model1 <- ergmito(fivenets ~ edges + nodematch("female")))
```

```
##
## ERGMito estimates
##
##  Coefficients:
##            edges  nodematch.female
##           -1.705             1.587
```

	Model 1
edges	-1.70^{**}
	(0.54)
nodematch.female	1.59^{*}
	(0.64)
AIC	73.34
BIC	77.53
Log Likelihood	-34.67
Num. networks	5

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 1: Statistical models

Simulation Study

We conducted a simulation study to explore the properties of MLE for small networks (a.k.a. ERGMito). To generate each sample of teams:

Simulation Study

We conducted a simulation study to explore the properties of MLE for small networks (a.k.a. ERGMito). To generate each sample of teams:

1. Draw the **population parameters** from a piecewise Uniform with values in $[-4, -.1] \cup [.1, 4]$

Simulation Study

We conducted a simulation study to explore the properties of MLE for small networks (a.k.a. ERGMito). To generate each sample of teams:

1. Draw the **population parameters** from a piecewise Uniform with values in $[-4, -.1] \cup [.1, 4]$
2. We will draw **groups of sizes 3 to 5**. The number of networks per group size are drawn from a Poisson distribution with parameter 10 (hence, an expected size of 30 networks per sample).

Simulation Study

We conducted a simulation study to explore the properties of MLE for small networks (a.k.a. ERGMito). To generate each sample of teams:

1. Draw the **population parameters** from a piecewise Uniform with values in $[-4, -.1] \cup [.1, 4]$
2. We will draw **groups of sizes 3 to 5**. The number of networks per group size are drawn from a Poisson distribution with parameter 10 (hence, an expected size of 30 networks per sample).
3. Use the **drawn parameters** and **group sizes** to **generate random graphs** using an ERGM data generating process.

Simulation Study

We conducted a simulation study to explore the properties of MLE for small networks (a.k.a. ERGMito). To generate each sample of teams:

1. Draw the **population parameters** from a piecewise Uniform with values in $[-4, -.1] \cup [.1, 4]$
2. We will draw **groups of sizes 3 to 5**. The number of networks per group size are drawn from a Poisson distribution with parameter 10 (hence, an expected size of 30 networks per sample).
3. Use the **drawn parameters** and **group sizes** to **generate random graphs** using an ERGM data generating process.

We simulated 100,000 samples, each one composed of an average of 30 networks.

Simulation Study (cont'd)

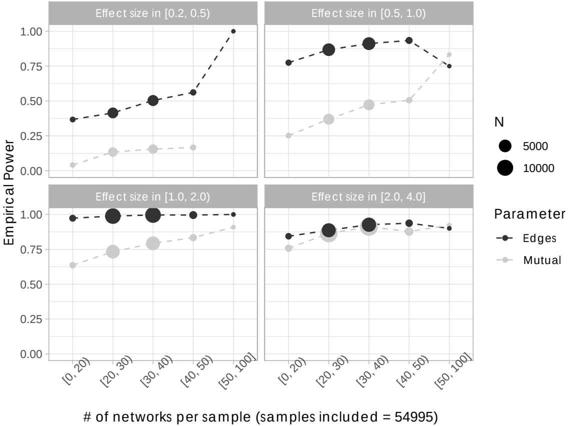


Figure 4: Empirical power of Pooled-ERGM estimates at various levels of effect size. As expected, power increases significantly with sample size (# of networks per sample). Interestingly, the discovery rate of an effect size within $[1, 2)$ is very high even with a sample size of 20-30 networks. More extreme points have higher volatility due to small number of samples included.

So now that we can estimate ERGMs for small networks (cool!)...

So now that we can estimate ERGMs for small networks (cool!)…

… what can this tell us about our 42 teams?

Preliminary results

	Advice	Influence	Leadership
Edges	-1.87^{***}	-0.78^{***}	-0.57^{***}
	(0.30)	(0.13)	(0.14)
Transitive Triads	0.24^{***}	0.21^{**}	
	(0.06)	(0.08)	
Indeg. RME	0.35^{***}		
	(0.08)		
Outdeg. Female	0.43^{*}		
	(0.19)		
Outdeg. Social Accomodation	0.11		
	(0.08)		
Indeg. Female			-0.38^{*}
			(0.19)
AIC	693.18	760.40	655.78
BIC	714.50	769.12	664.32
Log Likelihood	-341.59	-378.20	-325.89
Num. networks	38	41	38

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 2: The two statistics that showed to be the most robust were **Indeg. RME** and **Outdeg. Female**. These two effects can be described as (1) individuals with high levels of RME receive more ties, and (2) female subjects were more likely of seeking advice than male. Other statistics such as GPA, religiousness, age, and ethnicity were not significant.

Part II: Association between network structure and team performance

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:
  - Permutation: shuffle the dependent variable and compute, for example, correlations.

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:
  - Permutation: shuffle the dependent variable and compute, for example, correlations. **This assumes the outcome is completely independent from the observation.**

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:
  - Permutation: shuffle the dependent variable and compute, for example, correlations. **This assumes the outcome is completely independent from the observation.**
  - Rewiring (see Milo et al. 2004): Sample networks by generating new graphs using a rewiring algorithm, usually preserving some property such as degree sequence–the observed sequence of in/out degree.

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:
  - Permutation: shuffle the dependent variable and compute, for example, correlations. **This assumes the outcome is completely independent from the observation.**
  - Rewiring (see Milo et al. 2004): Sample networks by generating new graphs using a rewiring algorithm, usually preserving some property such as degree sequence–the observed sequence of in/out degree.**This assumes an oversimplifying data generating process.**

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:
  - Permutation: shuffle the dependent variable and compute, for example, correlations. **This assumes the outcome is completely independent from the observation.**
  - Rewiring (see Milo et al. 2004): Sample networks by generating new graphs using a rewiring algorithm, usually preserving some property such as degree sequence–the observed sequence of in/out degree.**This assumes an oversimplifying data generating process.** And worse, in a network of size 4, how many different networks can be observed **holding the degree sequence fixed**?

Testing effects of social network structure on group performance

Two common approaches: Generalized Linear Models (GLMs), or permutation-like tests. Both have limitations:

- GLMs:
  - Sample size is problematic: How costly is getting enough teams to run get a desired level of power?
- Permutation-like tests:
  - Permutation: shuffle the dependent variable and compute, for example, correlations. **This assumes the outcome is completely independent from the observation.**
  - Rewiring (see Milo et al. 2004): Sample networks by generating new graphs using a rewiring algorithm, usually preserving some property such as degree sequence–the observed sequence of in/out degree.**This assumes an oversimplifying data generating process.** And worse, in a network of size 4, how many different networks can be observed **holding the degree sequence fixed**?

BTW Degree sequence $\mapsto$ Scale-free networks

> “*The* ***structural diversity of real-world networks*** *uncovered here presents both a puzzle and an opportunity. The strong focus in the scientific literature on* ***explaining and exploiting scale-free patterns has meant relatively less is known about mechanisms that produce non-scale-free structural patterns****, e.g., those with degree distributions better fitted by a log-normal. Two important directions of future work will be the* ***development and validation of novel mechanisms for generating more realistic degree structure in networks****, and novel statistical techniques for identifying or untangling them given empirical data*”
>
> **– p. 8, Broido and Clauset (2019)**

See Holme (2019) for a recent reference on the Scale-free issue.

An Idea

What about using ERGMs to generate null distributions?

An Idea

What about using ERGMs to generate null distributions?

- Even for small networks, we can generate thousands of unique graphs, so sample size is not a problem.

An Idea

What about using ERGMs to generate null distributions?

- Even for small networks, we can generate thousands of unique graphs, so sample size is not a problem.
- As a difference from permutation tests:

An Idea

What about using ERGMs to generate null distributions?

- Even for small networks, we can generate thousands of unique graphs, so sample size is not a problem.
- As a difference from permutation tests:
  - ERGMs provide a more meaningful/realistic (or comprehensive if you like) data generating process

An Idea

What about using ERGMs to generate null distributions?

- Even for small networks, we can generate thousands of unique graphs, so sample size is not a problem.
- As a difference from permutation tests:
  - ERGMs provide a more meaningful/realistic (or comprehensive if you like) data generating process
  - On average, sampled networks will **look like the original** graph (addressing Broido and Clauset (2019)'s comments).

An Idea

What about using ERGMs to generate null distributions?

- Even for small networks, we can generate thousands of unique graphs, so sample size is not a problem.
- As a difference from permutation tests:
  - ERGMs provide a more meaningful/realistic (or comprehensive if you like) data generating process
  - On average, sampled networks will **look like the original** graph (addressing Broido and Clauset (2019)'s comments).

In principle, this would be equivalent to a revised rewiring test...

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.
2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.

2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).

3. Now, for $b \in \{1, \ldots, B\}$ do:

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.

2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).

3. Now, for $b \in \{1, \dots, B\}$ do:

   3.1 For each group j in $\{1, \dots, J\}$, draw a new network $\mathbf{g}_j^b \sim \mathcal{D}(\hat{\theta}, X_j)$, this new sequence is denoted $\mathbf{G}^b$

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.

2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).

3. Now, for $b \in \{1, \dots, B\}$ do:

   3.1 For each group j in $\{1, \dots, J\}$, draw a new network $\mathbf{g}_j^b \sim \mathcal{D}(\hat{\theta}, X_j)$, this new sequence is denoted $\mathbf{G}^b$

   3.2 Using $\mathbf{G}^b$ and Y, calculate $t_b = t(\mathbf{G}^b, Y)$

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.
2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).
3. Now, for $b \in \{1, \dots, B\}$ do:
    - **3.1** For each group j in $\{1, \dots, J\}$, draw a new network $\mathbf{g}_j^b \sim \mathcal{D}(\hat{\theta}, X_j)$, this new sequence is denoted $\mathbf{G}^b$
    - **3.2** Using $\mathbf{G}^b$ and Y, calculate $t_b = t(\mathbf{G}^b, Y)$
    - **3.3** Next b.

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.
2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).
3. Now, for $b \in \{1, \dots, B\}$ do:
   - **3.1** For each group j in $\{1, \dots, J\}$, draw a new network $\mathbf{g}_j^b \sim \mathcal{D}(\hat{\theta}, X_j)$, this new sequence is denoted $\mathbf{G}^b$
   - **3.2** Using $\mathbf{G}^b$ and Y, calculate $t_b = t(\mathbf{G}^b, Y)$
   - **3.3** Next b.

This will generate a null distribution for the statistic t, which we can use to compare against the observed statistic, t_0.

Algorithm

1. Estimate an ERGM (estimates can come from a single graph or pooled estimates). We denote the data-generating-process of this model as $\mathcal{D} : \Theta \times \mathcal{X} \mapsto \mathcal{G}$.

2. Calculate the value $t_0 = t(\mathbf{G}, Y)$, the observed test statistic (e.g. correlation between triads and y).

3. Now, for $b \in \{1, \dots, B\}$ do:

  3.1 For each group j in $\{1, \dots, J\}$, draw a new network $\mathbf{g}_j^b \sim \mathcal{D}(\hat{\theta}, X_j)$, this new sequence is denoted $\mathbf{G}^b$

  3.2 Using $\mathbf{G}^b$ and Y, calculate $t_b = t(\mathbf{G}^b, Y)$

  3.3 Next b.

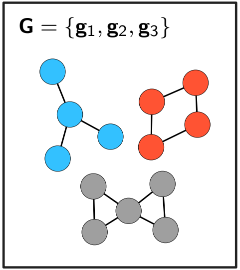
This will generate a null distribution for the statistic t, which we can use to compare against the observed statistic, t_0.

Note An important distinction to make is that structures that gave origin to the graph need not to be relevant for the team's performance per se.

Illustrated example

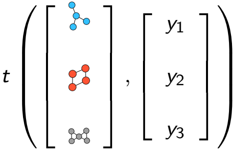
Suppose that we have a 3 networks of sizes 4, 4, and 5 respectively. The

Step 1:
Fit the ERGMito



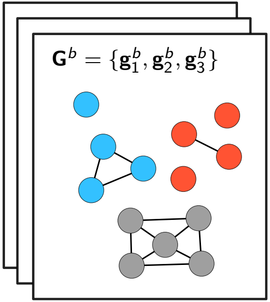
Fit the ERGMito,
This will give us $\mathcal{D}(\hat{\theta}, X_j)$

Step 2:
Calculate $t_0 =$



Throughout the simulations the only part that changes is the networks, not Y

Step 3:
For $b \in 1, \dots, B$ do



3.1) For $j \in \{1, 2, 3\}$ draw a new network from $\mathcal{D}$
3.2) Use the new sample to calculate $t_b = t(\mathbf{G}^b, Y)$

We can use the distribution of the sequence $\{t_1, \dots, t_B\}$ as null to compare against t_0

Toy simulation

Going back to our `fivenets` example:/pause

- Recall that our data-generating process for **G** was an ERGMito with parameters $\left(\theta_{edges}, \theta_{\texttt{nodematch("female")}}\right)$.

Toy simulation

Going back to our `fivenets` example:/pause

- Recall that our data-generating process for **G** was an ERGMito with parameters $\left(\theta_{edges}, \theta_{\texttt{nodematch("female")}}\right)$.
- Imagine that $y_{\mathbf{g}} = \texttt{nodeicov("female")}_{\mathbf{g}} + \varepsilon$, $\varepsilon \sim N(0,1)$

Toy simulation

Going back to our `fivenets` example:/pause

- Recall that our data-generating process for **G** was an ERGMito with parameters $\left(\theta_{edges}, \theta_{\texttt{nodematch("female")}}\right)$.
- Imagine that $y_{\mathbf{g}} = \texttt{nodeicov("female")}_{\mathbf{g}} + \varepsilon$, $\varepsilon \sim N(0,1)$
- Our test statistic t is the correlation between y and `nodeicov("female")`

Toy simulation

Going back to our `fivenets` example:/pause

- Recall that our data-generating process for **G** was an ERGMito with parameters $\left(\theta_{edges}, \theta_{\texttt{nodematch("female")}}\right)$.
- Imagine that $y_{\mathbf{g}} = \texttt{nodeicov("female")}_{\mathbf{g}} + \varepsilon$, $\varepsilon \sim N(0,1)$
- Our test statistic t is the correlation between y and `nodeicov("female")`
- The data looks something like this

Toy simulation

Going back to our `fivenets` example:/pause

- Recall that our data-generating process for **G** was an ERGMito with parameters $\left(\theta_{edges}, \theta_{\texttt{nodematch("female")}}\right)$.
- Imagine that $y_{\mathbf{g}} = \texttt{nodeicov("female")}_{\mathbf{g}} + \varepsilon$, $\varepsilon \sim N(0,1)$
- Our test statistic t is the correlation between y and `nodeicov("female")`
- The data looks something like this

y	nodeicov("female")
1.0138091	2
0.6051448	1
4.3085153	2
0.9547600	0
-0.1330788	1

Toy simulation

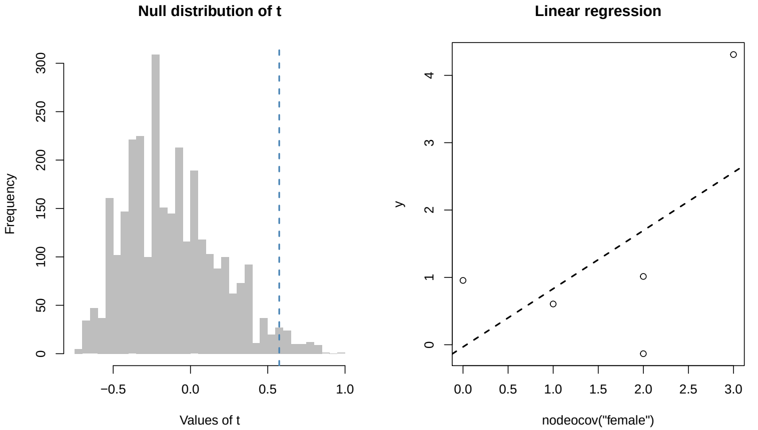


Figure 5: Comparing our method against a linear regression. Our proposed method returned a two sided p-value of 0.045, while the pvalue for the OLS coefficient was 0.311.

Concluding remarks

Exponential Random Graph Models for Small Networks:

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing,

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.

2. Preliminary results (both from simulations and applications) look encouraging.

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.

2. Preliminary results (both from simulations and applications) look encouraging.

3. Still work to do (on the development side of things): Goodness of fit tests, better algorithms for drawing random graphs, Bayesian models (because is fun. . .), etc.

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.

2. Preliminary results (both from simulations and applications) look encouraging.

3. Still work to do (on the development side of things): Goodness of fit tests, better algorithms for drawing random graphs, Bayesian models (because is fun. . .), etc.

4. Can be extended to other types of ERGMs. . . our next target: TERGMs (Separable Exponential Random Graph Models)

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.
2. Preliminary results (both from simulations and applications) look encouraging.
3. Still work to do (on the development side of things): Goodness of fit tests, better algorithms for drawing random graphs, Bayesian models (because is fun...), etc.
4. Can be extended to other types of ERGMs... our next target: TERGMs (Separable Exponential Random Graph Models)

Test for Association between graph level outcomes and graph structures:

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.

2. Preliminary results (both from simulations and applications) look encouraging.

3. Still work to do (on the development side of things): Goodness of fit tests, better algorithms for drawing random graphs, Bayesian models (because is fun. . .), etc.

4. Can be extended to other types of ERGMs. . . our next target: TERGMs (Separable Exponential Random Graph Models)

Test for Association between graph level outcomes and graph structures:

1. Still need to run simulation studies to explore **power** and **false discovery** rates.

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.
2. Preliminary results (both from simulations and applications) look encouraging.
3. Still work to do (on the development side of things): Goodness of fit tests, better algorithms for drawing random graphs, Bayesian models (because is fun. . .), etc.
4. Can be extended to other types of ERGMs. . . our next target: TERGMs (Separable Exponential Random Graph Models)

Test for Association between graph level outcomes and graph structures:

1. Still need to run simulation studies to explore **power** and **false discovery** rates.
2. Also on the development side of things, need to make things a bit faster and lightweight.

Concluding remarks

Exponential Random Graph Models for Small Networks:

1. Not a new thing, what's new is the tool to do so in a smooth way.
2. Preliminary results (both from simulations and applications) look encouraging.
3. Still work to do (on the development side of things): Goodness of fit tests, better algorithms for drawing random graphs, Bayesian models (because is fun. . .), etc.
4. Can be extended to other types of ERGMs. . . our next target: TERGMs (Separable Exponential Random Graph Models)

Test for Association between graph level outcomes and graph structures:

1. Still need to run simulation studies to explore **power** and **false discovery** rates.
2. Also on the development side of things, need to make things a bit faster and lightweight.
3. Working on a more formal statistical framework (when is it a good/bad idea to use this kind of method).

Thanks!

George G. Vega Yon

Let's chat!

vegayon@usc.edu

https://ggvy.cl

@gvegayon

@gvegayon

References I

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. 2018. Rmarkdown: Dynamic Documents for R. https://rmarkdown.rstudio.com.

Broido, Anna D, and Aaron Clauset. 2019. "Scale-Free Networks Are Rare." Nature Communications 10 (1): 1017.

Csardi, Gabor. 2015. Igraphdata: A Collection of Network Data Sets for the 'Igraph' Package. https://CRAN.R-project.org/package=igraphdata.

Geyer, Charles J., and Elizabeth A. Thompson. 1992. "Constrained Monte Carlo Maximum Likelihood for Dependent Data." Journal of the Royal Statistical Society. Series B (Methodological) 54 (3): 657–99. http://www.jstor.org/stable/2345852.

Handcock, Mark S. 2003. "Assessing Degeneracy in Statistical Models of Social Networks." Working Paper No. 39 76 (39): 33–50. https://doi.org/10.1.1.81.5086.

References II

Handcock, Mark S., David R. Hunter, Carter T. Butts, Steven M. Goodreau, Pavel N. Krivitsky, and Martina Morris. 2018. Ergm: Fit, Simulate and Diagnose Exponential-Family Models for Networks. The Statnet Project (http://www.statnet.org). https://CRAN.R-project.org/package=ergm.

Handcock, Mark, Peng Wang, Garry Robins, Tom Snijders, and Philippa Pattison. 2006. "Recent developments in exponential random graph (p*) models for social networks." Social Networks 29 (2): 192–215. https://doi.org/10.1016/j.socnet.2006.08.003.

Holme, Petter. 2019. "Rare and everywhere: Perspectives on scale-free networks." Nature Communications 10 (1): 1016. https://doi.org/10.1038/s41467-019-09038-8.

Hunter, David R., Mark S. Handcock, Carter T. Butts, Steven M. Goodreau, and Martina Morris. 2008. "Ergm: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks." Journal of Statistical Software 24 (3): 1–29.

References III

Kim, Young Ji, David Engel, Anita Williams Woolley, Jeffrey Yu-Ting Lin, Naomi McArthur, and Thomas W. Malone. 2017. “What Makes a Strong Team?: Using Collective Intelligence to Predict Team Performance in League of Legends.” In Proceedings of the 2017 Acm Conference on Computer Supported Cooperative Work and Social Computing, 2316–29. CSCW ’17. New York, NY, USA: ACM. https://doi.org/10.1145/2998181.2998185.

Milo, R, N Kashtan, S Itzkovitz, M E J Newman, and U Alon. 2004. “On the uniform generation of random graphs with prescribed degree sequences.” Arxiv Preprint Condmat0312028 cond-mat/0: 1–4. http://arxiv.org/abs/cond-mat/0312028.

R Core Team. 2018. R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Wasserman, Stanley, and Philippa Pattison. 1996. “Logit models and logistic regressions for social networks: I. An introduction to Markov graphs andp.” Psychometrika 61 (3): 401–25. https://doi.org/10.1007/BF02294547.

Xie, Yihui. 2018. Knitr: A General-Purpose Package for Dynamic Report Generation in R. https://yihui.name/knitr/.