What drives social networks? A gentle introduction to exponential random graph models (with a focus on small networks)

George G Vega Yon

Department of Preventive Medicine

LAERUG
June 10, 2019

Social networks

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)

What drives ~~social~~ networks?

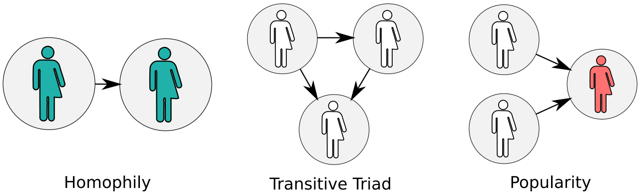
If *[blank]* asks you to predict a network

What kind of model?

What features would you include?

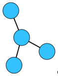
Exponential Family Random Graph Models (ERGMs)

Why are you and I are *[blank]* ? (friends, collaborators, etc.)



Let's build a model for this!

ERGMs from scratch

We need to build a probability function for  ...

$$\#edges, \#homophilic\ ties, \ldots$$

$$\theta_1 \times \#edges + \theta_2 \times \#homophilic\ ties + \ldots$$

$$\exp\left\{\theta_1 \times \#edges + \theta_2 \times \#homophilic\ ties + \ldots\right\}$$

$$\frac{\exp\{\theta_1 \times \#edges + \theta_2 \times \#homophilic\ ties + \ldots\}}{\sum \exp\{\ldots\}}$$

You got yourself an ERGM!

ERGMs... the *lingua franca* of SNA

A vector of model parameters

A vector of sufficient statistics

$$\Pr\left(\mathbf{Y} = \underbrace{\mathbf{y} \mid \theta, \mathbf{X}}_{\text{Observed data}}\right) = \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{y}, \mathbf{X}\right)\right\}}{\sum_{\mathbf{y}' \in \mathcal{Y}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{y}', \mathbf{X}\right)\right\}}, \quad \forall \mathbf{y} \in \mathcal{Y}$$

The normalizing constant

All possible networks

There is one problem with this model ...

A vector of model parameters

A vector of sufficient statistics

$$\Pr\left(\mathbf{Y} = \mathbf{y} \mid \theta, \mathbf{X}\right) = \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{y}, \mathbf{X}\right)\right\}}{\sum_{\mathbf{y}' \in \mathcal{Y}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{y}', \mathbf{X}\right)\right\}}, \quad \forall \mathbf{y} \in \mathcal{Y}$$

Observed data

The normalizing constant

All possible networks

because of $\mathcal{Y}$, the **normalizing constant** is a summation of $2^{n(n-1)}$ terms 😱!

To solve this, instead of directly computing this function, estimation is done by approximating ratios of likelihood functions instead (TL;DR we use simulations).

Let's get going

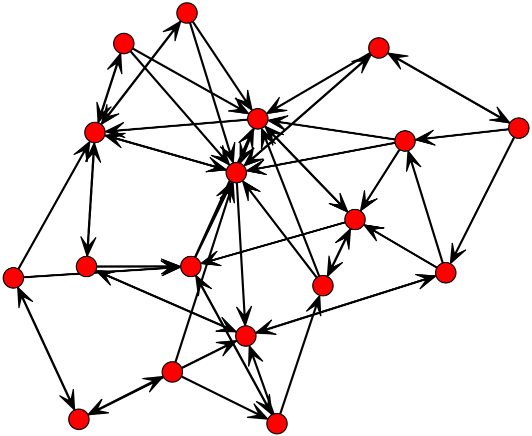
We will use the famous Monk data from Sampson (1969)

```r
library(ergm)
data(samplk, package="ergm")

# A glimpse into a network object (from the network package loaded with ergm)
samplk1
```

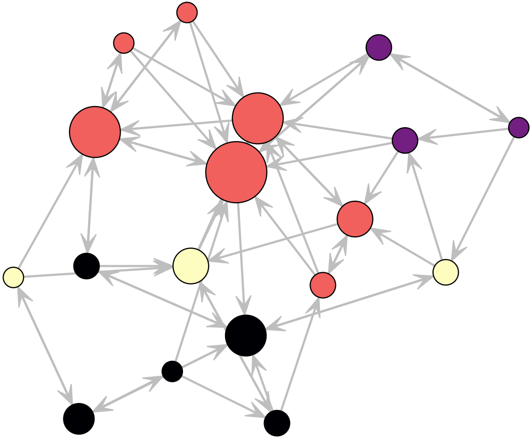
```
##  Network attributes:
##   vertices = 18
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 55
##     missing edges= 0
##     non-missing edges= 55
##
##  Vertex attribute names:
##     cloisterville group vertex.names
##
## No edge attributes
```

```r
library(sna) # Tools for SNA
set.seed(1)  # Graph layout is usually random-driven
gplot(samplk1)
```



Let's add some color and other features

```r
set.seed(1)
cols <- viridisLite::magma(4)[as.factor((samplk1 %v% "group"))]
gplot(samplk1, vertex.cex = degree(samplk1)/4, vertex.col = cols, edge.col = "gray")
```



A simple ergm model

```r
# Estimating the model
ans <- ergm(
  samplk1 ~ edges + nodematch("group") + ttriad,
  control = control.ergm(seed = 112)
  )
```

```
## Starting maximum pseudolikelihood estimation (MPLE):

## Evaluating the predictor and response matrix.

## Maximizing the pseudolikelihood.

## Finished MPLE.

## Starting Monte Carlo maximum likelihood estimation (MCMLE):

## Iteration 1 of at most 20:

## Optimizing with step length 1.

## The log-likelihood improved by 0.02337.

## Step length converged once. Increasing MCMC sample size.

## Iteration 2 of at most 20:

## Optimizing with step length 1.

## The log-likelihood improved by 0.002011.
```

```r
summary(ans)
```

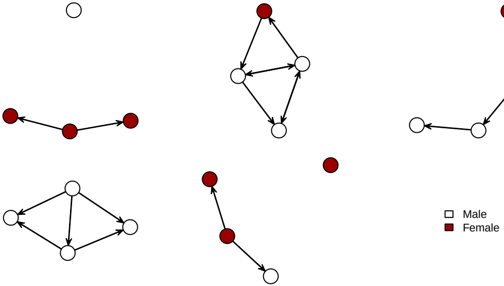
```
##
## ==========================
## Summary of model fit
## ==========================
##
## Formula:   samplk1 ~ edges + nodematch("group") + ttriad
##
## Iterations:  2 out of 20
##
## Monte Carlo MLE Results:
##                 Estimate Std. Error MCMC % z value Pr(>|z|)
## edges            -1.7738     0.3049      0  -5.819   <1e-04 ***
## nodematch.group   1.9730     0.3906      0   5.052   <1e-04 ***
## ttriple          -0.2984     0.1954      0  -1.527    0.127
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##      Null Deviance: 424.2  on 306  degrees of freedom
##  Residual Deviance: 255.8  on 303  degrees of freedom
##
## AIC: 261.8    BIC: 272.9    (Smaller is better.)
```

The common way to continue is: adding/removing terms, checking convergence, and checking goodness-of-fit.

Now its time for small networks!

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
```

How can we fit an ERGMito to this 5 networks?

`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
Convergence	0

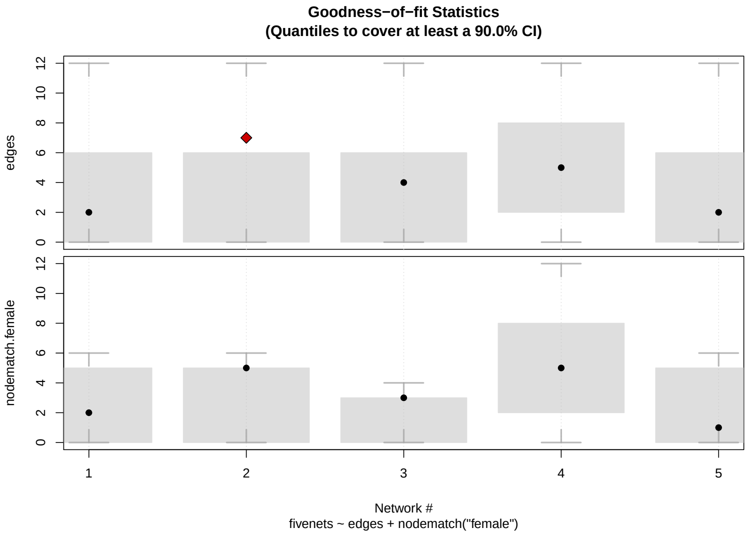
$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 1: Statistical models

```r
(gof1 <- gof_ergmito(model1))
```

```
##
## Goodness-of-fit for edges
##
##       obs min mean max lower upper lower prob. upper prob.
## net 1   2   0  3.7  12     0     6      0.0081        0.96
## net 2   7   0  3.7  12     0     6      0.0081        0.96
## net 3   4   0  3.1  12     0     6      0.0206        0.99
## net 4   5   0  5.6  12     2     8      0.0309        0.95
## net 5   2   0  3.7  12     0     6      0.0081        0.96
##
##
## Goodness-of-fit for nodematch.female
##
##       obs min mean max lower upper lower prob. upper prob.
## net 1   2   0  2.8   6     0     5       0.022        0.99
## net 2   5   0  2.8   6     0     5       0.022        0.99
## net 3   3   0  1.9   4     0     3       0.079        0.95
## net 4   5   0  5.6  12     2     8       0.031        0.95
## net 5   1   0  2.8   6     0     5       0.022        0.99
##
## Note: Exact confidence intervals where used. This implies that the requestes CI may differ from th
```

```r
plot(gof1)
```



Goodness−of−fit Statistics
(Quantiles to cover at least a 90.0% CI)
edges
nodematch.female
0
2
4
6
8
10
12
1
2
3
4
5
Network #
fivenets ~ edges + nodematch("female")

Thanks!

George G. Vega Yon

Let's chat!

vegayon@usc.edu

https://ggvy.cl

@gvegayon

@gvegayon

Appendix

Structures

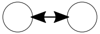
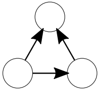
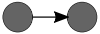
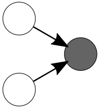
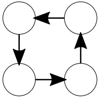
Representation	Description
	Mutual Ties (Reciprocity) $\sum_{i \neq j} y_{ij} y_{ji}$
	Transitive Triad (Balance) $\sum_{i \neq j \neq k} y_{ij} y_{jk} y_{ik}$
	Homophily $\sum_{i \neq j} y_{ij} \mathbf{1}(x_i = x_j)$
	Covariate Effect for Incoming Ties $\sum_{i \neq j} y_{ij} x_j$
	Four Cycle $\sum_{i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

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.

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.

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

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.

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

Sampson, Samuel F. 1969. "A Novitiate in a Period of Change: An Experimental and Case Study of Social Relationships."

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.

References II

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