The $p1$ model for mutuality

567 Statistical analysis of social networks

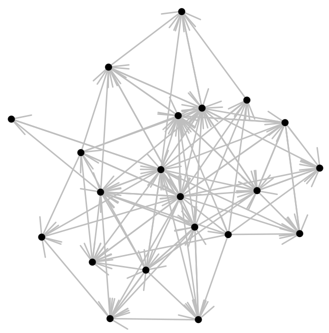
Peter Hoff

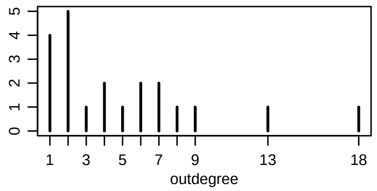
Statistics, University of Washington

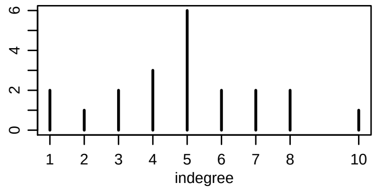
Running example

Managers data:

- nodeset = 21 managers in high-tech companies
- $y_{i,j}$ = presence of a directed friendship relation.







WARNING: These data have been tweaked for didactic purposes:

```
Y[7,11] <- Y[9,17] <- 1
```

Candidate models

$$M_0 : \Pr(Y_{i,j} = 1) = \frac{e^{\mu}}{1 + e^{\mu}}$$

$$M_r : \Pr(Y_{i,j} = 1) = \frac{e^{\mu + a_i}}{1 + e^{\mu + a_i}}$$

$$M_c : \Pr(Y_{i,j} = 1) = \frac{e^{\mu + b_j}}{1 + e^{\mu + b_j}}$$

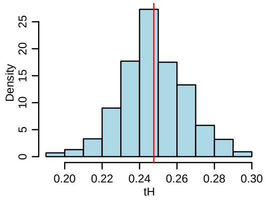
$$M_{rc} : \Pr(Y_{i,j} = 1) = \frac{e^{\mu + a_i + b_j}}{1 + e^{\mu + a_i + b_j}}$$

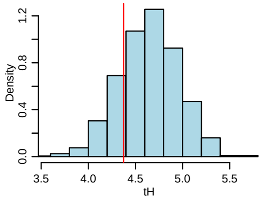
Model selection

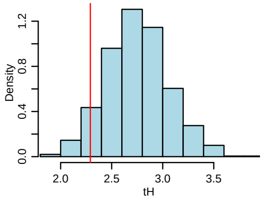
```
ridx<-c(matrix((1:nrow(Y)),nrow(Y),nrow(Y)))
cidx<-c(t(matrix((1:nrow(Y)),nrow(Y),nrow(Y)) ))
y<-c(Y)

fit.0<-glm( y ~ 1, family=binomial)
fit.r<-glm( y ~ C(factor(ridx),sum) , family=binomial)
fit.c<-glm( y ~ C(factor(cidx),sum) , family=binomial)
fit.rc<-glm( y ~ C(factor(ridx),sum)+C(factor(cidx),sum), family=binomial)

AIC(fit.0)

## [1] 472.1516

AIC(fit.r)

## [1] 412.2251

AIC(fit.c)

## [1] 481.9417

AIC(fit.rc)

## [1] 408.3688
```

Best case scenario comparison

```
mu.hat<-fit.rc$coef[1]
a.hat<- fit.rc$coef[1+1:(nrow(Y)-1)]   ; a.hat<-c(a.hat,-sum(a.hat) )
b.hat<- fit.rc$coef[nrow(Y)+1:(nrow(Y)-1)]   ; b.hat<-c(b.hat,-sum(b.hat) )

#### Best case scenario comparison
muij.mle<- mu.hat+ outer(a.hat,b.hat,"+")
p.mle<-exp(muij.mle)/(1+exp(muij.mle))

S.H<-NULL
for(s in 1:S)
{
  Ysim<-matrix(rbinom(nrow(Y)^2,1,p.mle),nrow(Y),nrow(Y)) ; diag(Ysim)<-NA
  S.H<-rbind(S.H, c(gmean(Ysim),sd(rsum(Ysim)),sd(csum(Ysim))) )
}

s.obs<-c(gmean(Y),sd(rsum(Y)),sd(csum(Y)))
mean(s.obs[3]<= S.H[,3])

## [1] 0.942
```







Evaluating reciprocity

```
M<-sum(Y*t(Y),na.rm=TRUE)/2
A<-sum(Y,na.rm=TRUE) - 2*M
N<- choose(nrow(Y),2) - M - A

M

## [1] 24

A

## [1] 56

N

## [1] 130

p11<-2*M/(2*M+A)
p10<-A/(A+2*N)
p11

## [1] 0.4615385

p10

## [1] 0.1772152

s.obs<-log( p11 * (1-p10) /( (1-p11) * p10) )
s.obs

## [1] 1.381179
```

Empirical reciprocity

Evididence for reciprocity:

$$\frac{\Pr(Y_{i,j} = 1 | Y_{j,i} = 1)}{\Pr(Y_{i,j} = 1 | Y_{j,i} = 0)} \approx 3.98$$

The corresponding log-odds are about 1.38.

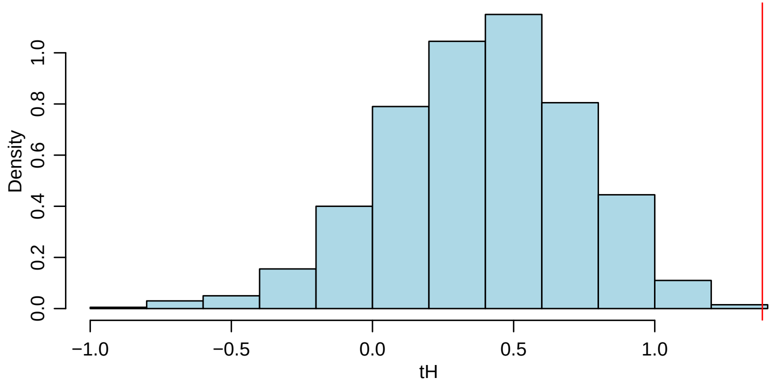
Is this large?

We need to compare this number to what we'd expect under our RCE model.

Best case scenario

```
S.H<-NULL
for(s in 1:S)
{
  Ysim<-matrix(rbinom(nrow(Y)^2,1,p.mle),nrow(Y),nrow(Y))
  diag(Ysim)<-NA

  M<-sum(Ysim*t(Ysim),na.rm=TRUE)/2
  A<-sum(Ysim,na.rm=TRUE) - 2*M
  N<- choose(nrow(Ysim),2) - M - A

  p11<-2*M/(2*M+A)
  p10<-A/(A+2*N)
  s.sim<-log( p11 * (1-p10) /( (1-p11) * p10) )

  S.H<-c(S.H,s.sim)
}
```

Best case scenario

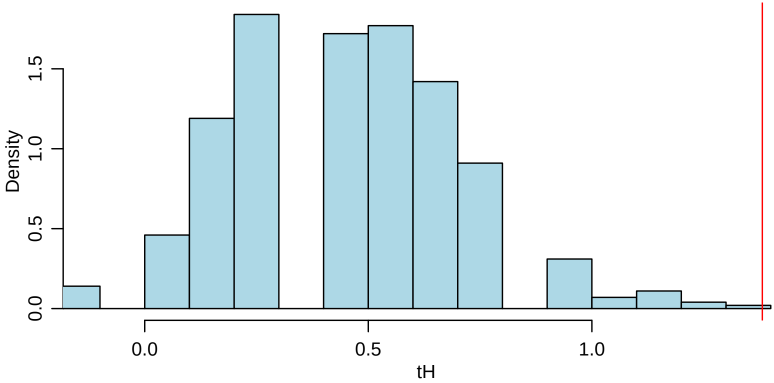


```
mean(S.H>=s.obs)

## [1] 0.001
```

Formal test of reciprocity with MCMC

```
S.H<-NULL
Ysim<-Y
for(s in 1:S)
{
  Ysim<-rY.Yrc(Ysim)

  M<-sum(Ysim*t(Ysim),na.rm=TRUE)/2
  A<-sum(Ysim,na.rm=TRUE) - 2*M
  N<- choose(nrow(Ysim),2) - M - A

  p11<-2*M/(2*M+A)
  p10<-A/(A+2*N)
  s.sim<-log( p11 * (1-p10) /( (1-p11) * p10) )

  S.H<-c(S.H,s.sim)
}
```

Formal test and p-value



```
mean(S.H>=s.obs)

## [1] 0.002
```

Within-dyad dependence

The RCE model fails in terms of representing mutuality:

- $s(\mathbf{Y}) = \sum_{i<j} y_{i,j} y_{j,i}$ larger than expected under independent RCE model.
- A large M relative to A and N is interpretable as within-dyad dependence:

$$\Pr(Y_{j,1} = 1 | Y_{i,j} = 1) > \Pr(Y_{j,i} = 1) > \Pr(Y_{j,1} = 1 | Y_{i,j} = 0)$$

So we have

- Model failure: The model doesn't represent the data feature $s(\mathbf{Y})$;
- Statistical failure: The data suggest statistical dependence in $(Y_{i,j}, Y_{j,i})$.

Statistical independence versus dependence

Under the RCE model

$$\Pr(\mathbf{Y} = y) = \prod_{i \neq j} \frac{e^{(\mu + a_i + b_j) y_{i,j}}}{1 + e^{\mu + a_i + b_j}}$$

In particular,

$$\begin{aligned}\Pr(Y_{i,j} = y_{i,j}, Y_{j,i} = y_{j,i}) &= \frac{e^{(\mu + a_i + b_j) y_{i,j}}}{1 + e^{\mu + a_i + b_j}} \times \frac{e^{(\mu + a_j + b_i) y_{i,j}}}{1 + e^{\mu + a_j + b_i}} \\ &= \Pr(Y_{i,j} = y_{i,j}) \times \Pr(Y_{j,i} = y_{j,i})\end{aligned}$$

This means, for example

$$\begin{aligned}\Pr(Y_{i,j} = 1 | Y_{j,i} = y_{i,j}) &= \frac{\Pr(Y_{i,j} = 1, Y_{j,i} = y_{j,i})}{\Pr(Y_{j,i} = y_{j,i})} \\ &= \frac{\Pr(Y_{i,j} = 1) \times \Pr(Y_{j,i} = y_{j,i})}{\Pr(Y_{j,i} = y_{j,i})} \\ &= \Pr(Y_{i,j} = 1)\end{aligned}$$

$$\Pr(Y_{i,j} = 1 | Y_{j,i} = 1) = \Pr(Y_{i,j} = 1 | Y_{j,i} = 0) = \Pr(Y_{i,j} = 1)$$

A model for dependence

For notational convenience, let $\mu_{i,j} = \mu + a_i + b_j$.

To accommodate within-dyad dependence, we want something like

$$\Pr(Y_{i,j} = 1 | Y_{j,i} = 0) = \frac{e^{\mu_{i,j}}}{1 + e^{\mu_{i,j}}}$$
$$\Pr(Y_{i,j} = 1 | Y_{j,i} = 1) = \frac{e^{\mu_{i,j}+\gamma}}{1 + e^{\mu_{i,j}+\gamma}}$$

Convince yourself that if $\gamma \gtreqqless 0$, then

$$\Pr(Y_{i,j} = 1 | Y_{j,i} = 1) \gtreqqless \Pr(Y_{i,j} 1 | Y_{j,i} = 0).$$

Additionally, you should be able to show

$$\log \text{odds}(Y_{i,j} = 1 | Y_{j,i} = 1, Y_{i,j} = 0) = \gamma$$

A model for dependence

The proposed conditional model is as follows:

$$\Pr(Y_{i,j} = y_{i,j} | Y_{j,i} = y_{j,i}) \propto e^{\mu_{i,j} y_{i,j} + \gamma y_{i,j} y_{j,i}}$$

To get a joint model for $\{Y_{i,j}, Y_{j,i}\}$, we need to multiply by $\Pr(Y_{j,i} = y_{j,i})$:

$$\begin{aligned} \Pr(Y_{i,j} = y_{i,j}, Y_{j,i} = y_{j,i}) &= \Pr(Y_{i,j} = y_{i,j} | Y_{j,i} = y_{j,i}) \times \Pr(Y_{j,i} = y_{j,i}) \\ &\propto e^{\mu_{i,j} y_{i,j} + \gamma y_{i,j} y_{j,i}} \times \Pr(Y_{j,i} = y_{j,i}). \end{aligned}$$

Symmetry suggests

$$\Pr(Y_{i,j} = y_{i,j}, Y_{j,i} = y_{j,i}) \propto e^{\mu_{i,j} y_{i,j} + \mu_{j,i} y_{j,i} + \gamma y_{i,j} y_{j,i}}$$

A model for dependence

$$\Pr(Y_{i,j} = y_{i,j}, Y_{j,i} = y_{j,i}) \propto \exp(\mu_{i,j} y_{i,j} + \mu_{j,i} y_{j,i} + \gamma y_{i,j} y_{j,i})$$

Denote $\Pr(Y_{i,j} = y_{i,j}, Y_{j,i} = y_{j,i}) = p(y_{i,j}, y_{j,i})$. Then

$$\begin{aligned} p(0,0) &= c_{i,j} \\ p(1,0) &= c_{i,j} \exp(\mu_{i,j}) \\ p(0,1) &= c_{i,j} \exp(\mu_{j,i}) \\ p(1,1) &= c_{i,j} \exp(\mu_{i,j} + \mu_{j,i} + \gamma) \end{aligned}$$

What is $c_{i,j}$?

$$\begin{aligned} 1 &= p(0,0) + p(1,0) + p(0,1) + p(1,1) \\ 1 &= c_{i,j}(1 + e^{\mu_{i,j}} + e^{\mu_{j,i}} + e^{\mu_{i,j}+\mu_{j,i}+\gamma}) \\ c_{i,j} &= \frac{1}{1 + e^{\mu_{i,j}} + e^{\mu_{j,i}} + e^{\mu_{i,j}+\mu_{j,i}+\gamma}} \end{aligned}$$

p_1 dependence model

Our proposed statistical model is as follows:

- Between-dyad relations are independent;
- Within dyad relations have the following distribution:

$$p(y_{i,j}, y_{j,i} | \mu, a_i, b_j, \gamma) = \frac{e^{\mu_{i,j} y_{i,j} + \mu_{j,i} y_{j,i} + \gamma y_{i,j} y_{j,i}}}{1 + e^{\mu_{i,j}} + e^{\mu_{j,i}} + e^{\mu_{i,j} + \mu_{j,i} + \gamma}}$$

This is the so-called "p_1" network model (Holland and Leinhardt, 1981).

As you might suspect, this is a type of ERGM.

Let's find the sufficient statistics.

Sufficient statistics for p1

$$\begin{aligned}
\Pr(\mathbf{Y}=\mathbf{y}|\mu,\mathbf{a},\mathbf{b},\gamma) &= \prod_{i<j} p(y_{i,j},y_{j,i}|\mu,a_i,a_j,b_i,b_j,\gamma) \\
&= \prod_{i<j} c_{i,j} e^{\mu_{i,j}y_{i,j}+\mu_{j,i}y_{j,i}+\gamma y_{i,j}y_{j,i}} \\
&= \left(\prod_{i<j} c_{i,j}\right)\exp(\sum_{i<j}\mu_{i,j}y_{i,j}+\sum_{i<j}\mu_{j,i}y_{j,i}+\sum_{i<j}\gamma y_{i,j}y_{j,i}) \\
&= c(\mu,\mathbf{a},\mathbf{b},\gamma)\exp(\sum_{i\neq j}\mu_{i,j}y_{i,j}+\gamma\sum_{i<j}y_{i,j}y_{j,i})
\end{aligned}$$

Sufficient statistics for p_1

Recall $\mu_{i,j} = \mu + a_i + b_j$, so

$$\begin{aligned}\sum_{i \neq j} \mu_{i,j} y_{i,j} &= \mu \sum_{i \neq j} y_{i,j} + \sum_i a_i \sum_{j:j \neq i} y_{i,j} + \sum_j b_j \sum_{i:i \neq j} y_{i,j} + \gamma \sum_{i<j} y_{i,j} y_{j,i} \\ &= (\mu, a_1, \ldots, a_n, b_1, \ldots, b_n, \gamma) \cdot (y_{\cdot\cdot}, y_{1\cdot}, \ldots, y_{n\cdot}, y_{\cdot 1}, \ldots, y_{\cdot n}, \sum_{i<j} y_{i,j} y_{j,i})\end{aligned}$$

Sufficient statistics for p_1 are therefore

- the edge total;
- the outdegrees and indegrees;
- the total mutual dyads.

Estimation

Maximum likelihood estimation: Find $(\mu, \mathbf{a}, \mathbf{b}, \gamma)$ to maximize

$$\begin{aligned} l(\mu, \mathbf{a}, \mathbf{b}, \gamma : \mathbf{y}) &= \log \Pr(\mathbf{Y} = \mathbf{y} | \mu, \mathbf{a}, \mathbf{b}, \gamma) \\ &= \sum_{i<j} \log p(y_{i,j}, y_{j,i} | \mu, a_i, b_j, \gamma) \end{aligned}$$

```
ll.p1<-function(Y,mu,a,b,g)
{
  mij<- mu+outer(a,b,"+")
  diag(mij)<-NA
  lnum<- sum(  mij*Y  + t(Y*mij) + g*(Y*(t(Y))),na.rm=TRUE )/2
  lden<-sum( log( 1+exp(mij)+exp(t(mij))+exp(mij+t(mij)+g)),na.rm=TRUE )/2
  lnum-lden
}
```

Profile likelihood

Recall, the RCE model is obtained by setting $\gamma = 0$:

$$RCE = \{\Pr(\mathbf{Y} = \mathbf{y} | \mu, \mathbf{a}, \mathbf{b}, \gamma) : \gamma = 0\}$$

We can obtain estimates of $(\mu, \mathbf{a}, \mathbf{b})$ under the RCE via `glm`:

```
fit.rc<-glm( y ~ C(factor(ridx),sum)+C(factor(cidx),sum), family=binomial)
mu.hat<-fit.rc$coef[1]
a.hat<- fit.rc$coef[1+1:(nrow(Y)-1)]    ; a.hat<-c(a.hat,-sum(a.hat) )
b.hat<- fit.rc$coef[nrow(Y)+1:(nrow(Y)-1)]    ; b.hat<-c(b.hat,-sum(b.hat) )
```

Reality check:
The maximized log-likelihood of this model should equal $l(\hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}}, 0 : \mathbf{y})$

```
logLik(fit.rc)

## 'log Lik.' -163.1844 (df=41)

ll.p1(Y,mu.hat, a.hat, b.hat, 0 )

## [1] -163.1844
```

Profile likelihood

The **profile likelihood** is the likelihood

as a function of one parameter,

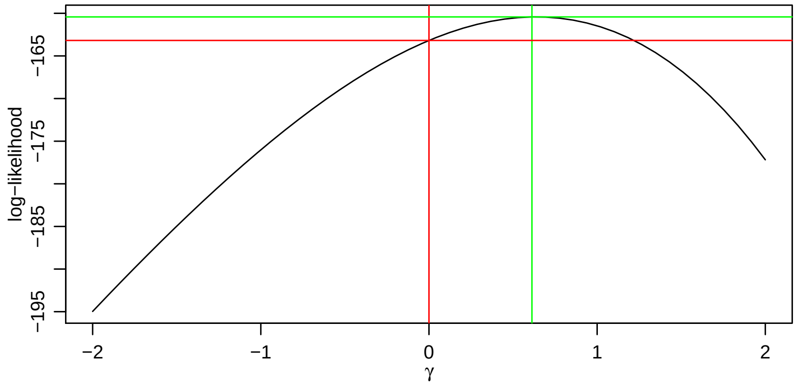
with the other parameters fixed at a particular estimate.

Let's examine the profile likelihood in γ:

$$\tilde{l}(\gamma : \mathbf{Y}, \hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}}) = l(\hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}}, \gamma : \mathbf{Y}) =$$

```r
GLL<-NULL
for(g in seq(-2,2,length=50))
{
  GLL<-rbind(GLL,c(g,ll.p1(Y,mu.hat,a.hat,b.hat,g)) )
}
```

Profile likelihood



```
GLL[  which.max(GLL[,2]),]

## [1]    0.6122449 -160.4248852

logLik(fit.rc)

## 'log Lik.' -163.1844 (df=41)
```

Limits of profile likelihood

Let $\tilde{\gamma}$ be the maximizer of the profile likelihood:

$$\tilde{\gamma} = \arg\max_{\gamma} \tilde{l}(\gamma : \mathbf{Y}, \hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}})$$

The values $(\hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}}, \tilde{\gamma})$ are *not generally* the MLE:

- $(\hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}})$ are only "best" when $\gamma = 0$.
- $\tilde{\gamma}$ is only "best" for $(\hat{\mu}, \hat{\mathbf{a}}, \hat{\mathbf{b}})$.
- The MLEs *simultaneously* maximize the likelihood.

Fitting *p*1 with ergm

The *p*1 and other ERGMs can be fit in R with some additional software.

```
> library(ergm)

ergm: version 3.2.4, created on 2014-12-13
Copyright (c) 2014, Mark S. Handcock, University of California -- Los Angeles
                    David R. Hunter, Penn State University
                    Carter T. Butts, University of California -- Irvine
                    Steven M. Goodreau, University of Washington
                    Pavel N. Krivitsky, University of Wollongong
                    Martina Morris, University of Washington
                    with contributions from
                    Li Wang
                    Kirk Li, University of Washington
Based on "statnet" project software (statnet.org).
For license and citation information see statnet.org/attribution
or type citation("ergm").
```

The `ergm` package allows (in theory) estimation and inference for ERGMs.

- accommodates a variety of sufficient statistics;
- accommodates covariate effects;
- accommodates more complicated model features (eg. random effects).

Fitting $p1$ with ergm

First, lets fit the RCE with the `ergm` function.

A model is fit to data with `ergm` by specifying the sociomatrix `Y` and the sufficient statistics:

```
fit <- ergm( Y ~ sstat1 + sstat2 + sstat3 )
```

The above pseudocode fits an ERGM to `Y` having sufficient statistics `sstat1`, `sstat2` and `sstat3`.

The sufficient statistics have particular predefined names.

For example, the following command fits the RCE model:

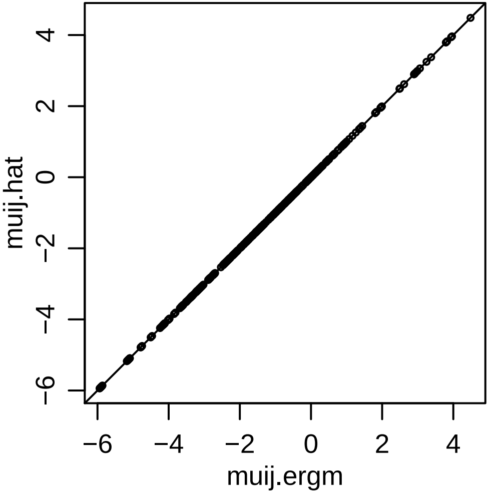
```
fit.rc.ergm <- ergm( Y ~ edges + sender + receiver )
```

Reality check

```
betas<-fit.rc.ergm$coef
mu.ergm<-betas[1]
a.ergm<-c(0,betas[2:nrow(Y)]  )
b.ergm<-c(0,betas[nrow(Y)+1:(nrow(Y)-1) ] )


mu.ergm-mu.hat

##    edges
## 1.550538

a.ergm-a.hat

##             sender2   sender3   sender4   sender5   sender6   sender7
## -0.335315 -0.335315 -0.335315 -0.335315 -0.335315 -0.335315 -0.335315
##   sender8   sender9  sender10  sender11  sender12  sender13  sender14
## -0.335315 -0.335315 -0.335315 -0.335315 -0.335315 -0.335315 -0.335315
##  sender15  sender16  sender17  sender18  sender19  sender20  sender21
## -0.335315 -0.335315 -0.335315 -0.335315 -0.335315 -0.335315 -0.335315

b.ergm-b.hat

##             receiver2  receiver3  receiver4  receiver5  receiver6
##  -1.215223  -1.215223  -1.215223  -1.215223  -1.215223  -1.215223
##  receiver7  receiver8  receiver9 receiver10 receiver11 receiver12
##  -1.215223  -1.215223  -1.215223  -1.215223  -1.215223  -1.215223
## receiver13 receiver14 receiver15 receiver16 receiver17 receiver18
##  -1.215223  -1.215223  -1.215223  -1.215223  -1.215223  -1.215223
## receiver19 receiver20 receiver21
##  -1.215223  -1.215223  -1.215223
```

Reality check

```
muij.ergm <- mu.ergm +outer(a.ergm,b.ergm,"+")
muij.hat<- mu.hat+ outer(a.hat,b.hat,"+")
```



Reality check

```
logLik(fit.rc)

## 'log Lik.' -163.1844 (df=41)

logLik(fit.rc.ergm)

## 'log Lik.' -219.2304 (df=41)
```

```
ll.p1(Y,mu.ergm,a.ergm,b.ergm,0)

## [1] -163.1844

ll.p1(Y,mu.hat,a.hat,b.hat,0)

## [1] -163.1844
```

Fitting the $p1$ model

We have just used `ergm` to fit the RCE ERGM:

$$\Pr(\mathbf{Y} = \mathbf{y} | \mu, \mathbf{a}, \mathbf{b}) = c(\mu, \mathbf{a}, \mathbf{b}) \exp(\mu y_{\cdot\cdot} + \sum_{i=1}^{n} a_i y_{i\cdot} + \sum_{j=1}^{n} b_j y_{\cdot j})$$

The $p1$ model is just the RCE model with an additional sufficient statistic:

$$\Pr(\mathbf{Y} = \mathbf{y} | \mu, \mathbf{a}, \mathbf{b}, \gamma) = c(\mu, \mathbf{a}, \mathbf{b}, \gamma) \exp(\mu y_{\cdot\cdot} + \sum_{i=1}^{n} a_i y_{i\cdot} + \sum_{j=1}^{n} b_j y_{\cdot j} + \gamma \sum_{i<j} y_{i,j} y_{j,i})$$

Fitting the $p1$ model

This model is easily specified in `ergm`:

```
fit.rcm.ergm<-ergm(Y ~ edges + sender + receiver + mutual)

## Iteration 1 of at most 20:
## The log-likelihood improved by 1.476
## Step length converged once. Increasing MCMC sample size.
## Iteration 2 of at most 20:
## The log-likelihood improved by 0.4795
## Step length converged twice. Stopping.
##
## This model was fit using MCMC.  To examine model diagnostics and check for degeneracy,
```

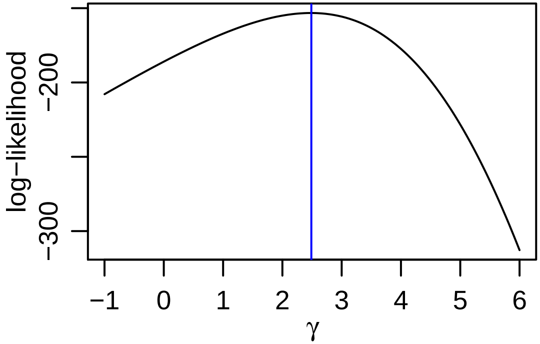
Fitting the $p1$ model

```
summary(fit.rcm.ergm)

##
## ==========================
## Summary of model fit
## ==========================
##
## Formula:   Y ~ edges + sender + receiver + mutual
##
## Iterations:  2 out of 20
##
## Monte Carlo MLE Results:
##            Estimate Std. Error MCMC % p-value
## edges      -1.20687    0.73364      0 0.10079
## sender2    -1.01988    0.92642      0 0.27165
## sender3    -1.02331    0.99691      0 0.30532
## sender4     0.65827    0.85327      0 0.44091
## sender5     0.84375    0.84413      0 0.31817
## sender6     1.04552    0.85719      0 0.22334
## sender7    -1.53711    1.27560      0 0.22895
## sender8    -1.86464    1.23701      0 0.13255
## sender9    -2.04243    1.28152      0 0.11183
## sender10    1.45133    0.83505      0 0.08302 .
## sender11    2.46813    0.85584      0 0.00415 **
## sender12   -0.39952    0.88042      0 0.65025
## sender13   -0.42186    1.00983      0 0.67636
## sender14   -1.04542    1.01202      0 0.30226
## sender15    1.36164    0.84258      0 0.10692
## sender16   -0.87130    1.02607      0 0.39633
## sender17    4.61969    1.14183      0 < 1e-04 ***
```

Reality check

```
betas<-fit.rcm.ergm$coef
mu.rcm<-betas[1]
a.rcm<-c(0,betas[2:nrow(Y)]  )
b.rcm<-c(0,betas[nrow(Y)+1:(nrow(Y)-1) ] )
gamma.rcm<-betas[2*nrow(Y)]

gamma.rcm

##   mutual
## 2.484522

GLLM<-NULL
for(g in seq(-1,6,length=300))
{
  GLLM<-rbind(GLLM,c(g,ll.p1(Y,mu.rcm,a.rcm,b.rcm,g)) )
}
```

Reality check



Reality check

```
gamma.rcm ; logLik(fit.rcm.ergm)

##   mutual
## 2.484522
## 'log Lik.' -153.3574 (df=42)

GLLM[ which.max(GLLM[,2]), ]

## [1]    2.488294 -153.261507
```

```
logLik(fit.rc.ergm)

## 'log Lik.' -219.2304 (df=41)

GLL[ which.max(GLL[,2]), ]

## [1]    0.6122449 -160.4248852
```

How does `ergm` work?

Recall the general ERGM:

$$\Pr(\mathbf{Y} = \mathbf{y} | \boldsymbol{\theta}) = c(\boldsymbol{\theta}) \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta})$$

$t(\mathbf{y})$	$(t_1(\mathbf{y}), \ldots, t_p(\mathbf{y}))$
θ	$(\theta_1, \ldots, \theta_p)$
$c(\boldsymbol{\theta})$	is a normalizing constant.

Let $\mathbf{y}$ be the observed value of the network. The MLE $\hat{\boldsymbol{\theta}}$ maximizes the log-likelihood:

$$l(\boldsymbol{\theta} : \mathbf{y}) = \log \Pr(\mathbf{Y} = \mathbf{y} | \boldsymbol{\theta}) = \boldsymbol{\theta}^T \mathbf{t}(\mathbf{y}) + \log c(\boldsymbol{\theta}).$$

A standard approach to function estimation is with **gradient ascent** This approach requires the derivatives of $l(\boldsymbol{\theta} : \mathbf{y})$

$$\nabla l(\boldsymbol{\theta} : \mathbf{y}) = \mathbf{t}(\mathbf{y}) + \nabla \log c(\boldsymbol{\theta})$$

Likelihood derivatives

$$\nabla \log c(\boldsymbol{\theta}) = \frac{\nabla c(\boldsymbol{\theta})}{c(\boldsymbol{\theta})}$$

What is $c(\boldsymbol{\theta})$? Recall, $\sum_{\mathbf{y}\in\mathcal{Y}} \Pr(\mathbf{Y} = \mathbf{y}|\boldsymbol{\theta}) = 1$. Therefore,

$$\begin{aligned}
1 &= \sum_{\mathbf{y}} \Pr(\mathbf{Y} = \mathbf{y}|\boldsymbol{\theta}) \\
1 &= \sum_{\mathbf{y}} c(\boldsymbol{\theta}) \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta}) \\
1 &= c(\boldsymbol{\theta}) \sum_{\mathbf{y}} \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta}) \\
c^{-1}(\boldsymbol{\theta}) &= \sum_{\mathbf{y}} \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta}) \\
c(\boldsymbol{\theta}) &= \frac{1}{\sum_{\mathbf{y}} \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta})}
\end{aligned}$$

Likelihood derivatives

$$\nabla \log c(\boldsymbol{\theta}) = -\nabla \log c(\boldsymbol{\theta})^{-1}$$
$$= -\frac{\sum_{\mathbf{y}} \mathbf{t}(\mathbf{y}) \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta})}{\sum_{\mathbf{y}} \exp(t(\mathbf{y}) \cdot \boldsymbol{\theta})}$$

For each step of gradient ascent we need to calculate $\nabla \log c(\boldsymbol{\theta})$.
This requires summing over all possible $n \times n$ graphs $\mathbf{y}$.

n	number of graphs
2	$2^2 = 4$
3	$2^6 = 64$
n	$2^{n(n-1)}$
20	$2^{300} = 2.46 \times 10^{114}$

This isn't going to work.

MCMCMLE

The `ergm` package takes a different strategy.

Consider comparing a "reference" value $\boldsymbol{\theta}_0$ to $\boldsymbol{\theta}$:

$$\begin{aligned} l(\boldsymbol{\theta}) - l(\boldsymbol{\theta}_0) &= \boldsymbol{\theta} \cdot \mathbf{t}(\mathbf{y}) + \log c(\boldsymbol{\theta}) - \boldsymbol{\theta}_0 \cdot \mathbf{t}(\mathbf{y}) - \log c(\boldsymbol{\theta}_0) \\ &= (\boldsymbol{\theta} - \boldsymbol{\theta}_0) \cdot \mathbf{t}(\mathbf{y}) + \log \frac{c(\boldsymbol{\theta})}{c(\boldsymbol{\theta}_0)} \end{aligned}$$

It turns out that

$$\log \frac{c(\boldsymbol{\theta})}{c(\boldsymbol{\theta}_0)} = \mathrm{E}[\exp((\boldsymbol{\theta}_0 - \boldsymbol{\theta}) \cdot \mathbf{t}(\mathbf{Y}))|\boldsymbol{\theta}_0]$$

This is an average, that can be approximated with an MCMC routine.

The `ergm` fitting routine is roughly as follows: Given a current value of $\boldsymbol{\theta}_0$

1. Run an MCMC routine to approximate $\log \frac{c(\boldsymbol{\theta})}{c(\boldsymbol{\theta}_0)}$ for $\boldsymbol{\theta}$ near $\boldsymbol{\theta}_0$.
2. Approximate the derivative near $\boldsymbol{\theta}_0$
3. Move along the derivative to a new value of $\boldsymbol{\theta}_0$
4. Repeat until convergence.

Speed and convergence

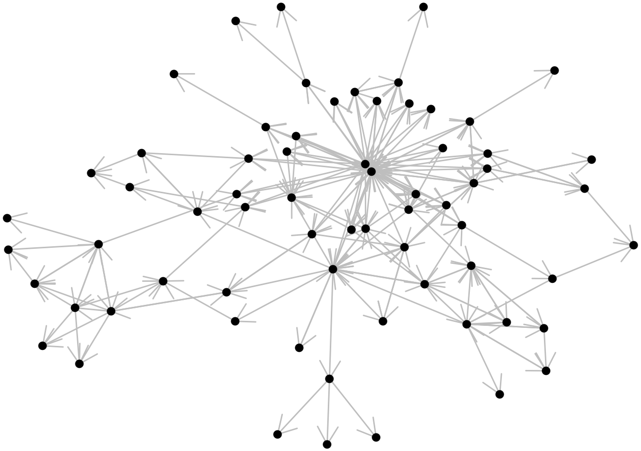
This procedure can take a long time:

```
> date()
[1] "Wed Feb  5 14:20:43 2014"
> fit.rcm.ergm<-ergm(Y ~ edges + sender + receiver + mutual)
Iteration 1 of at most 20:
Convergence test P-value: 0e+00
The log-likelihood improved by 0.4616
Iteration 2 of at most 20:
Convergence test P-value: 9.9e-163
The log-likelihood improved by 0.08207
Iteration 3 of at most 20:
Convergence test P-value: 3.2e-47
The log-likelihood improved by 0.03287
Iteration 4 of at most 20:
Convergence test P-value: 1.9e-14
The log-likelihood improved by 0.01313
.
.
.
Iteration 19 of at most 20:
Convergence test P-value: 5.6e-01
Convergence detected. Stopping.
The log-likelihood improved by 0.003658

This model was fit using MCMC.  To examine model diagnostics and check for degeneracy, use the mcmc.diagno
> date()
[1] "Wed Feb  5 14:22:30 2014"
```

Speed and convergence

```
> date()
[1] "Fri Feb 20 15:35:02 2015"
> fit.rcm.ergm<-ergm(Y ~ edges + sender + receiver + mutual)
Iteration 1 of at most 20:
The log-likelihood improved by 1.617
Step length converged once. Increasing MCMC sample size.
Iteration 2 of at most 20:
The log-likelihood improved by 0.1834
Step length converged twice. Stopping.

This model was fit using MCMC.  To examine model diagnostics and check for degeneracy, use the mcmc.diagno
Warning message:
In .ergm.mvar.spec0(z) :
  Excessive correlation among the statistics. Using a no-crosscorrelation approximation.
> date()
[1] "Fri Feb 20 15:35:20 2015"
```

A larger datatset



Speed and convergence

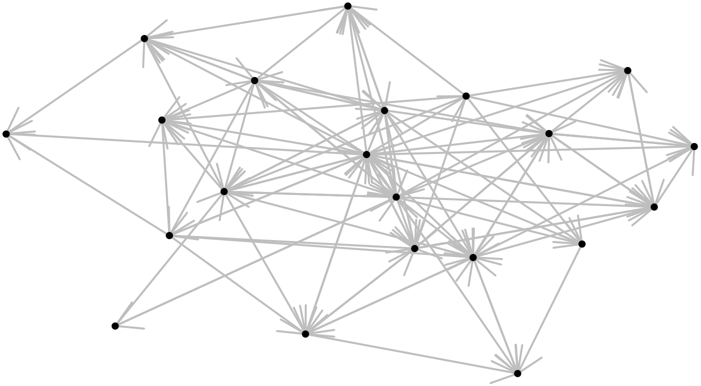
```
> date()
[1] "Wed Feb 13 15:11:57 2013"

> fit.rcm.ergm<-ergm(Yc90 ~ edges + sender + receiver + mutual)
Iteration 1 of at most 20:
the log-likelihood improved by 3.602
Iteration 2 of at most 20:
the log-likelihood improved by 1.886
Iteration 3 of at most 20:
the log-likelihood improved by 5.315
Iteration 4 of at most 20:
the log-likelihood improved by 4.038
.
.
Iteration 17 of at most 20:
the log-likelihood improved by 0.275
Iteration 18 of at most 20:
the log-likelihood improved by 0.3019
Iteration 19 of at most 20:
the log-likelihood improved by 0.4783
Iteration 20 of at most 20:
the log-likelihood improved by 0.3691

date()
[1] "Wed Feb 13 15:22:17 2013"
```

Speed and convergence

```
> date()
[1] "Fri Feb 20 15:36:05 2015"
> fit.rcm.ergm<-ergm(Yc90 ~ edges + sender + receiver + mutual)
Observed statistic(s) sender2, sender7, sender10, sender12, sender14, sender15, sender21, sender24, sender
Iteration 1 of at most 20:
The log-likelihood improved by 1.143
Iteration 2 of at most 20:
The log-likelihood improved by 1.218
Iteration 3 of at most 20:
The log-likelihood improved by 1.14
Iteration 4 of at most 20:
The log-likelihood improved by 1.188
Step length converged once. Increasing MCMC sample size.
Iteration 5 of at most 20:
The log-likelihood improved by 0.6905
Step length converged twice. Stopping.

This model was fit using MCMC.  To examine model diagnostics and check for degeneracy, use the mcmc.diagno
Warning messages:
1: In .ergm.mvar.spec0(z) :
  Excessive correlation among the statistics. Using a no-crosscorrelation approximation.
2: In ergm.checkextreme.model(model = model.initial, nw = nw, init = control$init,  :
  Observed statistic(s) sender2, sender7, sender10, sender12, sender14, sender15, sender21, sender24, send
> date()
[1] "Fri Feb 20 15:37:28 2015"
```

An ergm bug or two

```
Y<-htmanagers$Y[,,2]
rsum(Y)

##  [1]  5  3  2  6  7  6  0  1  0  7 13  4  2  2  8  2 18  1  9  2  4

csum(Y)

##  [1]  8 10  5  5  6  2  3  5  6  1  6  8  1  5  4  4  6  4  5  3  5
```



An ergm bug or two

Fitting the $p1$ model:

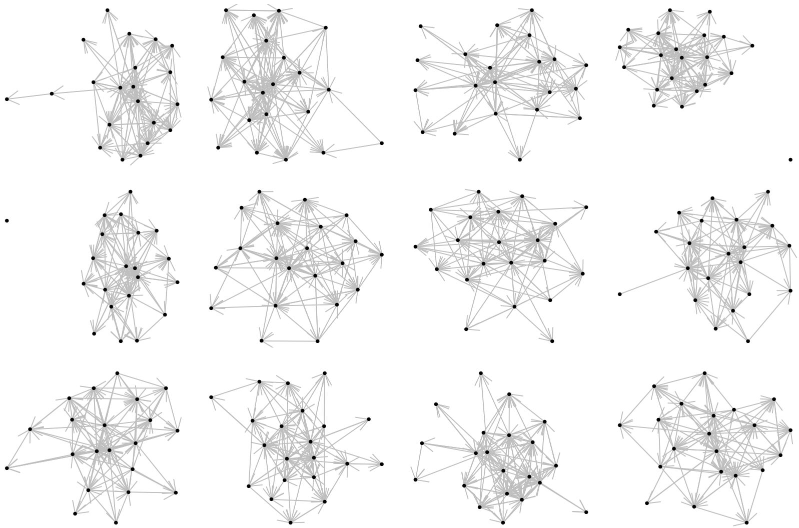
```
fit.rcm.ergm<-ergm( Y ~ edges + sender + receiver + mutual )

## Observed statistic(s) sender7 and sender9 are at their smallest attainable values. Their coefficients w
## Iteration 1 of at most 20:
## The log-likelihood improved by 2.195
## Iteration 2 of at most 20:
## The log-likelihood improved by 0.5805
## Step length converged once. Increasing MCMC sample size.
## Iteration 3 of at most 20:
## The log-likelihood improved by 0.07849
## Step length converged twice. Stopping.
##
## This model was fit using MCMC.  To examine model diagnostics and check for degeneracy, use the mcmc.dia
```

```
logLik(fit.rcm.ergm)

## 'log Lik.' NaN (df=42)
```

Goodness of fit via network simulation

ergm provides a convenient function for network simulation:

```
Ysim<-simulate(fit.rcm.ergm)
```

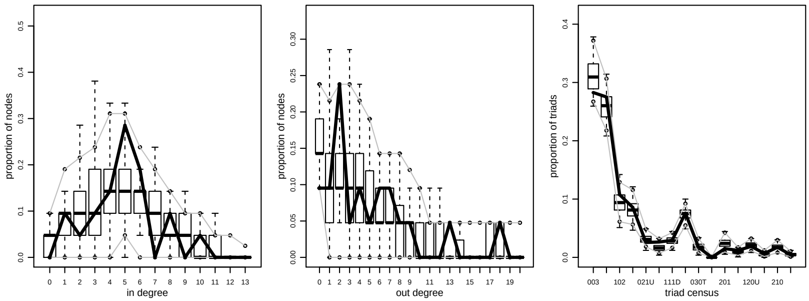


Goodness of fit via network simulation

Simulation can be handy for goodness of fit checks.

ergm uses simulate to evaluate fit for some pre-specified statistics:

```
p1gof<-gof( fit.rcm.ergm ,  GOF= ~ idegree + odegree + triadcensus  )
```


proportion of nodes
in degree
proportion of nodes
out degree
proportion of triads
003 102 021U 111D 030T 201 120U 210
triad census