

Transitivity

567 Statistical analysis of social networks

Peter Hoff

Statistics, University of Washington

Beyond second order dependence

Second order dependence:

Variances, covariances and correlations all involve **second order moments**:

$$\text{Cov}[\epsilon_{i,j}, \epsilon_{k,l}] = \text{E}[\epsilon_{i,j}, \epsilon_{k,l}]$$

Higher order dependence:

Variances and covariances cannot represent **higher order moments**:

$$\text{E}[\epsilon_{i,j}\epsilon_{k,l}\epsilon_{m,n}] = ?$$

Questions:

Are there higher order dependencies in network data?

Is the SRM covariance structure sufficient or deficient?

Goodness of fit

Consider the following goodness of fit statistic:

$$t(\mathbf{Y}) = \sum_i \sum_j \sum_k \tilde{y}_{i,j} \tilde{y}_{j,k} \tilde{y}_{k,i}$$

where $\tilde{y}_{i,j} = 1 \times (y_{i,j} + y_{j,i} > 0)$.

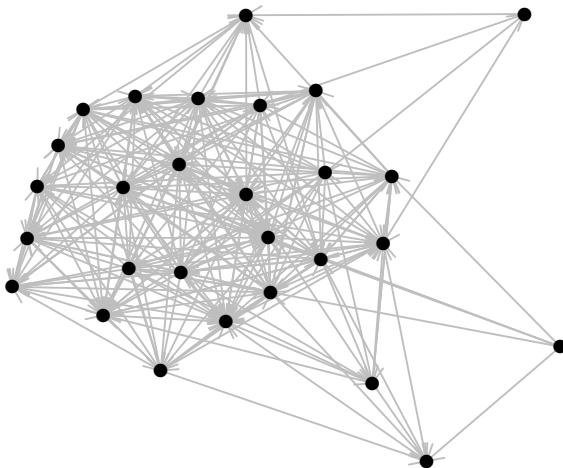
```
t_trans

## function (Y)
## {
##   YS <- 1 * (Y + t(Y) > 0)
##   sm <- 0
##   for (i in 1:nrow(YS)) {
##     ci <- which(YS[i, ] > 0)
##     sm <- sm + sum(YS[ci, ci], na.rm = TRUE)
##   }
##   sm/6
## }
## <environment: namespace:amen>
```

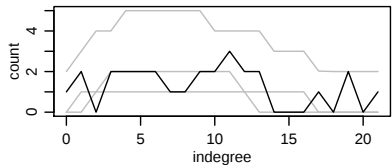
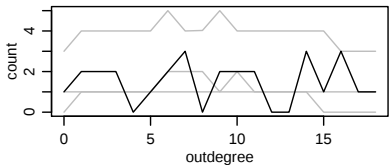
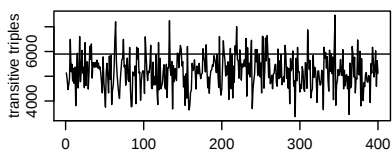
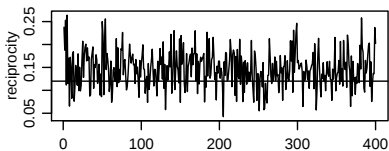
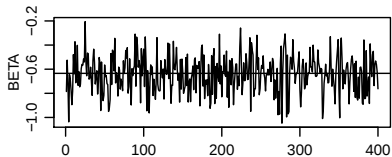
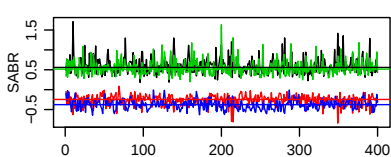
$t(\mathbf{Y})$ counts the number of triangles in the graph.

- it overcounts - it is really six times the number of triangles;
- it counts triangles regardless of the direction of the ties.

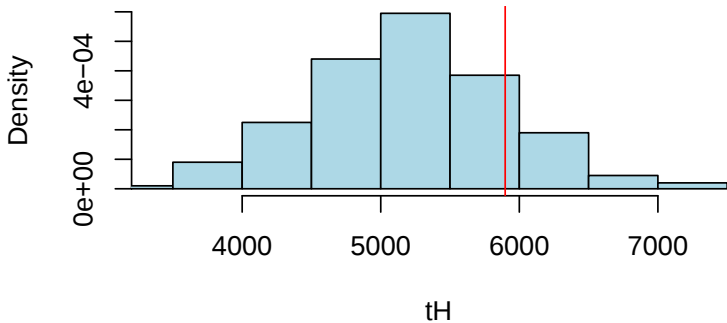
GOF - sheep data



GOF - sheep data



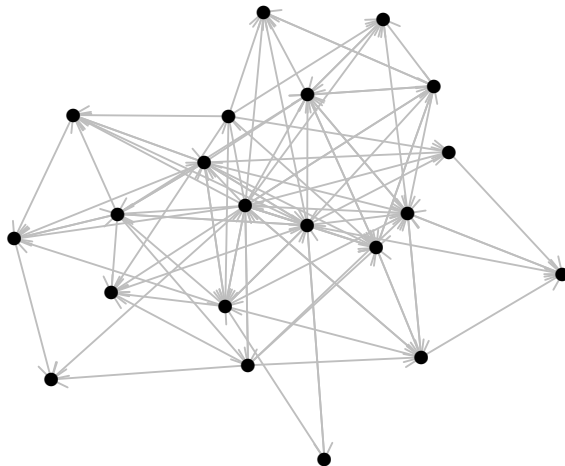
GOF - sheep data



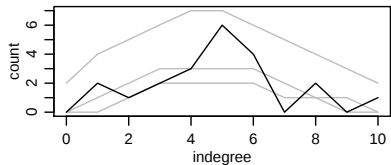
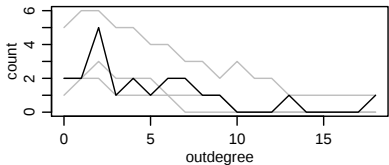
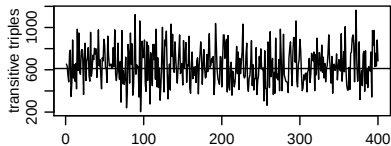
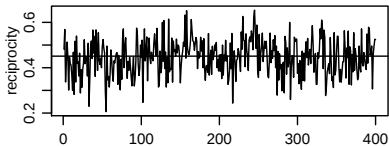
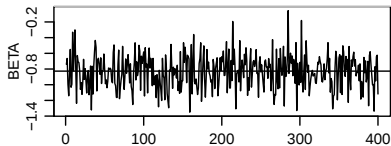
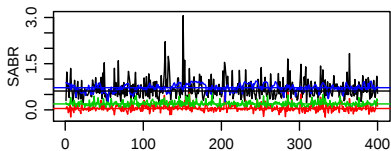
```
mean(fit$TT >= fit$tt)
```

```
## [1] 0.15
```

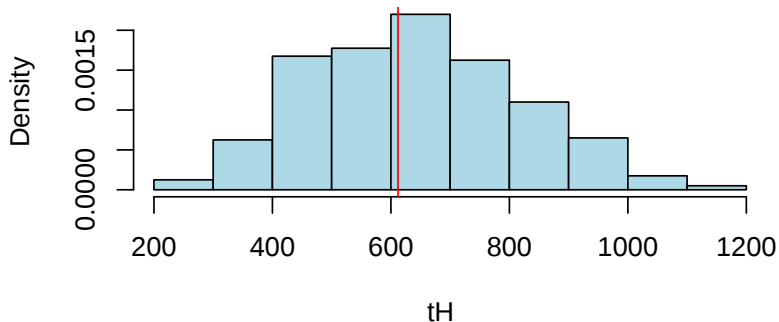
GOF - high tech managers



GOF - high tech managers



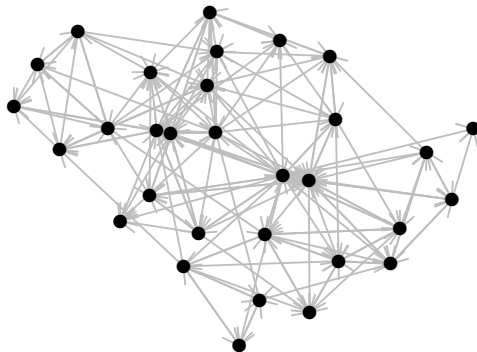
GOF - high tech managers



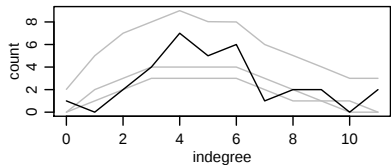
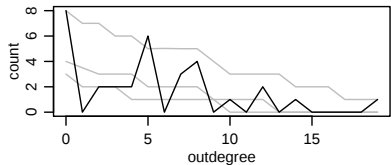
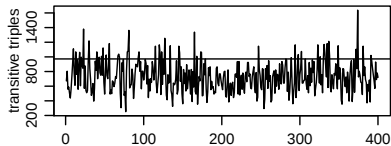
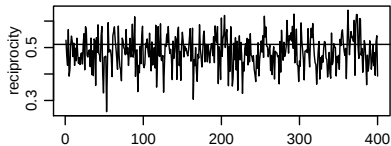
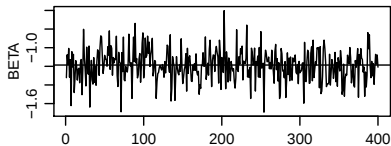
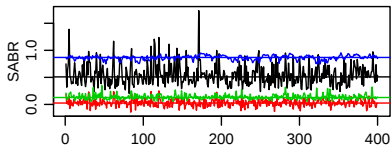
```
mean(fit$TT >= fit$tt)
```

```
## [1] 0.5625
```

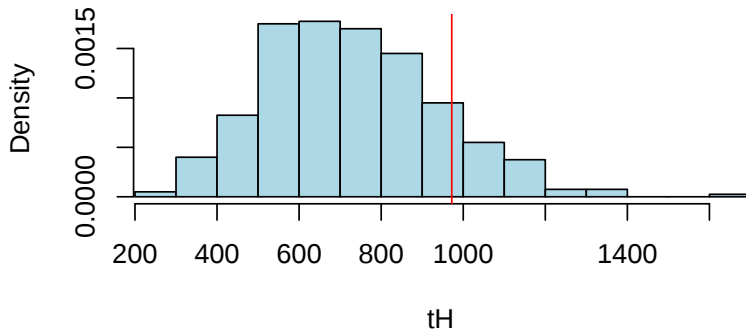
GOF - Dutch college friendships



GOF - Dutch college friendships



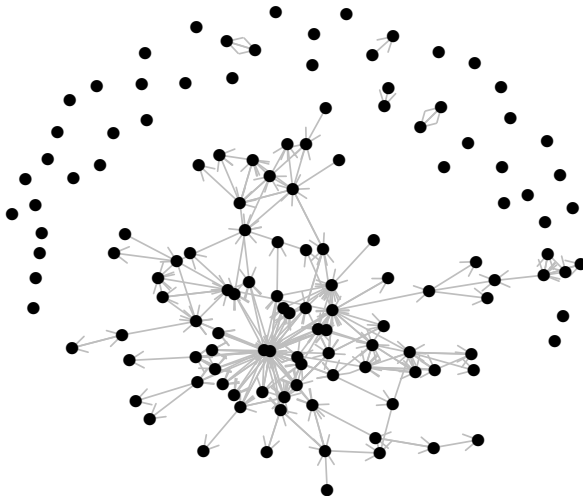
GOF - Dutch college friendships



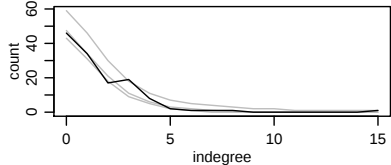
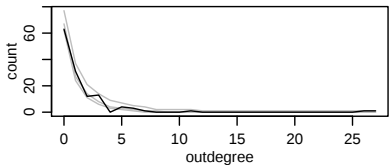
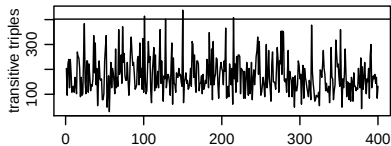
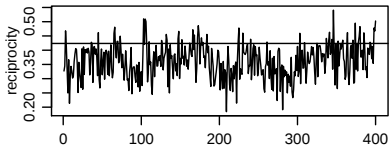
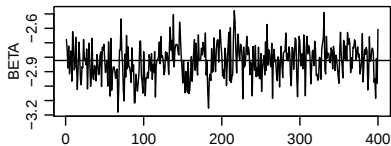
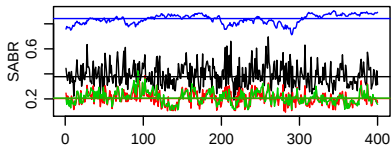
```
mean(fit$TT >= fit$tt)
```

```
## [1] 0.13
```

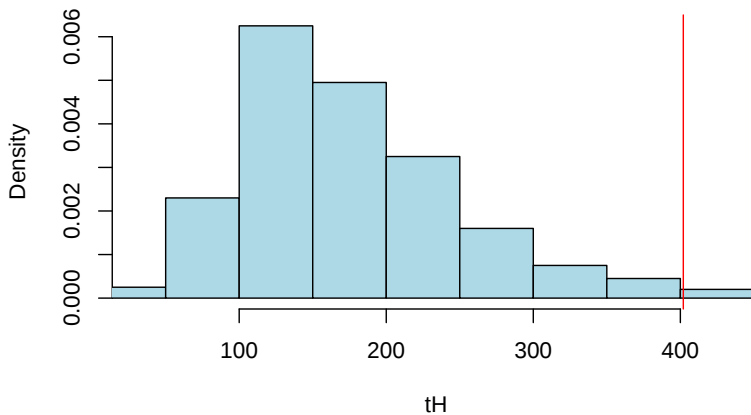
GOF - Conflict in the 90s



GOF - Conflict in the 90s



GOF - Conflict in the 90s



```
mean(fit$TT >= fit$tt)
```

```
## [1] 0.01
```

Excess triangles and transitivity

Some evidence that the SRM may not always be sufficient:

- Networks often have more “triangles” than predicted under the model.

This corresponds with several social theories about relations:

transitivity: a social preference to be friends with your friends' friends.

balance: a social preference to be friends with your enemies' enemies.

homophily: a social preference to be friends with others similar to you.

These social models may not be distinguishable from network data.

Exercise: Explain why each of these may lead to triangles in a network.

Triads and triangles

A **triad** is an unordered subset of three nodes.

Consider for simplicity an undirected binary relation.

The node-generated subgraph of a triad is given by

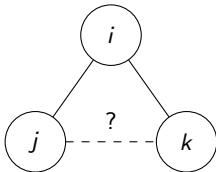
$$\mathbf{Y}[(i, j, k), (i, j, k)] = \begin{pmatrix} \square & y_{i,j} & y_{i,k} \\ y_{i,j} & \square & y_{j,k} \\ y_{i,k} & y_{j,k} & \square \end{pmatrix}$$

3 relations and 2 possible states per relation $\Rightarrow 2^3 = 8$ possible triad states.

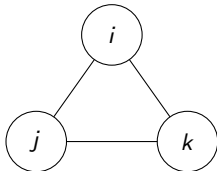
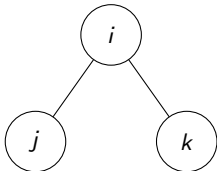
Exercise: Draw the triad states.

Triads, two-stars and triangles

Consider a node i connected to both nodes j and k .



What are the possibilities for this triad?



Triads, two-stars and triangles

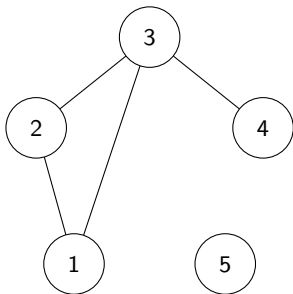
For a given network, how much transitivity is there?

How many triangles occur, relative to how many are “possible”?

Triad census: A count of the number of each type of triad.

$$t(\mathbf{Y}) = \{ \# \text{ null, one edge, two-star, triangle} \}$$

Triad census



<i>i</i>	<i>j</i>	<i>k</i>	type
1	2	3	triangle
1	2	4	one edge
1	2	5	one edge
1	3	4	two star
1	3	5	one edge
1	4	5	null
2	3	4	two star
2	3	5	one edge
2	4	5	null
3	4	5	one edge

$$t(\mathbf{Y}) = (2, 5, 2, 1)$$

Computing the triad census

For a given triad, the state is given by the number of edges:

0 edges: null

1 edge: one edge

2 edges: two star

3 edges: triangle

```
tcensus <- c(0, 0, 0, 0)
for (i in 1:(n - 2)) {
  for (j in (i + 1):(n - 1)) {
    for (k in (j + 1):n) {

      Yijk <- Y[c(i, j, k), c(i, j, k)]
      nedges <- sum(Yijk, na.rm = TRUE)/2
      tcensus[nedges + 1] <- tcensus[nedges + 1] + 1

    }
  }
}

tcensus

## [1] 2 5 2 1
```

Computing the triad census

Nested loops will be too slow for large graphs.
Here is a more efficient algorithm:

```
tt <- c(0, 0) # count two stars and triangles
for (i in 1:n) {
  i.alterns <- which(Y[i, ] > 0)
  tt <- tt + c(sum(1 - Y[i.alterns, i.alterns], na.rm = TRUE), sum(Y[i.alterns,
    i.alterns], na.rm = TRUE))
}
tt/c(2, 6)

## [1] 2 1
```

This counts the two stars and the triangles.
The nulls and one-edges can be found by applying the algorithm to $1 - Y$.

Computing the triad census

```
triad_census(Y)

## [1] 2 5 2 1

triad_census(Yht) # high tech managers

## [1] 376 509 343 102

triad_census(Ydc) # Dutch college

## [1] 2158 2016 624 162

triad_census(Ysd) # sheep dominance

## [1] 235 955 1103 983

triad_census(Y90) # 90s conflict

## [1] 338443 18221 1029 67
```

Evaluating transitivity

Transitivity: “more triangles than expected”

One measure of transitivity is to compare the number of triangles to the number of possible triangles:

number of triangles: number of three-tie triads.

number of possible triangles: number of two- or three-tie triads.

$$\begin{aligned}\text{transitivity index} &= \frac{\# \text{triangles}}{\# \text{triangles or two-stars}} \\ &\approx \Pr(y_{k,j} = 1 | y_{i,j} = 1 \text{ and } y_{i,k} = 1)\end{aligned}$$

This transitivity index can be viewed as how $y_{k,j}$ depends on $y_{i,j}$ and $y_{i,k}$.

Transitivity index

```
tc <- triad_census(Yht) # high tech managers
tc[4]/(tc[3] + tc[4])

## [1] 0.2292

tc <- triad_census(Ydc) # Dutch college
tc[4]/(tc[3] + tc[4])

## [1] 0.2061

tc <- triad_census(Ysd) # sheep dominance
tc[4]/(tc[3] + tc[4])

## [1] 0.4712

tc <- triad_census(Y90) # 90s conflict
tc[4]/(tc[3] + tc[4])

## [1] 0.06113
```

Scaling the transitivity index

These results may seem counter-intuitive:

- Y90 seemed to be “more transitive” according to GOF plots;
- Y90 has the lowest transitivity index.

To what should the transitivity index be compared?

Transitivity index

```
tc <- triad_census(Yht) # high tech managers
tc[4]/(tc[3] + tc[4])

## [1] 0.2292

mean(Yht, na.rm = TRUE)

## [1] 0.2429

##

tc <- triad_census(Y90) # 90s conflict
tc[4]/(tc[3] + tc[4])

## [1] 0.06113

mean(Y90, na.rm = TRUE)

## [1] 0.0121
```

A scaled transitivity index

Intuitively, think of transitivity as the “effect” of $y_{i,j} = 1$ and $y_{i,k} = 1$ on $y_{j,k}$. This can be measured with a log-odds ratio. Let

- $\tilde{p}$ = transitive triads/(transitive + two-star triads) ;
- $p = \bar{y}$.

$$\begin{aligned}\tau &= \log \frac{\text{odds}(y_{j,k} = 1 : y_{i,j} = 1, y_{i,k} = 1)}{\text{odds}(y_{j,k} = 1)} \\ &= \log \frac{\tilde{p}}{1 - \tilde{p}} \frac{1 - p}{p}\end{aligned}$$

A scaled transitivity index

```
tlor <- function(Y) {  
  p <- mean(Y, na.rm = TRUE)  
  tc <- triad_census(Y)  
  pt <- tc[4]/(tc[3] + tc[4])  
  
  od <- p/(1 - p)  
  odt <- pt/(1 - pt)  
  
  log(odt/od)  
}
```

A scaled transitivity index

```
tlor(Yht) # high tech managers
```

```
## [1] -0.07568
```

```
tlor(Ydc) # Dutch college
```

```
## [1] 0.2417
```

```
tlor(Ysd) # sheep dominance
```

```
## [1] 0.6438
```

```
tlor(Y90) # 90s conflict
```

```
## [1] 1.67
```

By this measure Y90 is the most transitive network, matching our intuition.

Triad census for directed data

For undirected relations in a triad there are

- $2^3 = 8$ possible graphs;
- 4 isomorphic graphs.

For directed relations the situation is more complicated:

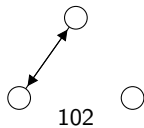
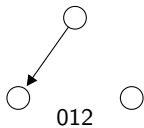
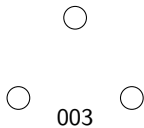
- $2^2 = 4$ states per dyad, and 3 dyads, means

$$(2^2)^3 = 2^6 = 64 \text{ possible states}$$

- 16 isomorphic states.

Directed triad states

Each triad state is named according to its dyad census:



Transitivity

Transitive triple:

Consider a triple $\{i, j, k\}$ for which $i \rightarrow j$ and $j \rightarrow k$.

The triple is

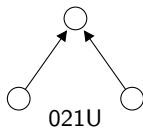
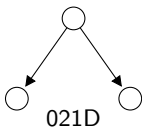
transitive if $i \rightarrow k$;

intransitive if $i \not\rightarrow k$.

Social theory: Nodes seek out transitive relations, avoid intransitive ones.

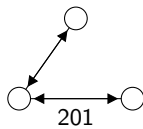
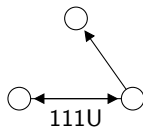
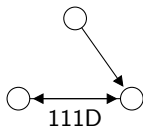
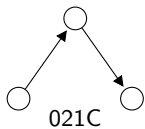
Transitivity

Based on the definition, 003, 012, 102 and the following triads are neither transitive nor intransitive:



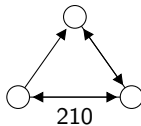
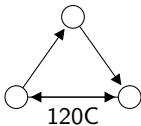
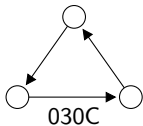
Transitivity

Any triad with a null dyad cannot be transitive:



Transitivity

Triads with no null dyads can be intransitive, transitive or mixed:

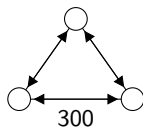
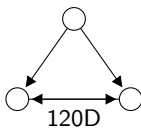
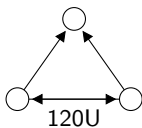
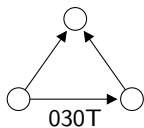


Which of these is intransitive? Why are the others “mixed”?

Transitivity

The triads below are “transitive” in that they all have

- some transitivity;
- no intransitivity.



Modeling transitivity in ERGMs

Recall the basic ERGModel:

$$\Pr(\mathbf{Y} = \mathbf{y}) = c(\boldsymbol{\theta}) \exp(\theta_1 t_1(\mathbf{y}) + \cdots + \theta_K t_K(\mathbf{y}))$$

Evaluation of various forms of transitivity is accomplished by including such sufficient statistics among $t_1(\mathbf{y}), \dots, t_K(\mathbf{y})$.

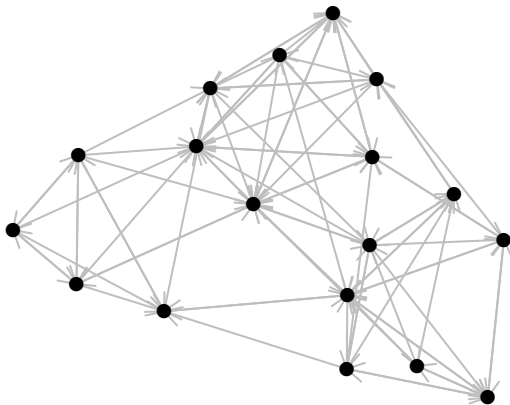
Let's try this out through a model fitting exercise:

1. $\mathbf{Y} \sim \text{edges} + \text{mutual}$
2. $\mathbf{Y} \sim \text{edges} + \text{mutual} + \text{transitivity}$

where transitivity is some network statistic involving triples.

Monk data

Relation: $y_{i,j} = 1$ if i "liked" j at any one of the three time periods.



ERGM fit

```
fit_erg1 <- ergm(Y ~ edges + mutual)

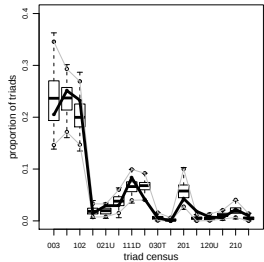
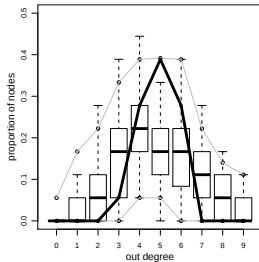
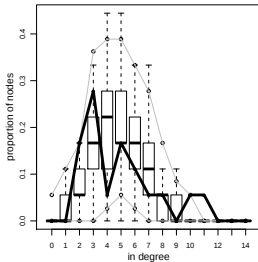
## Iteration 1 of at most 20:
## Convergence test P-value: 6.6e-01
## Convergence detected. Stopping.
## The log-likelihood improved by < 0.0001
##
## This model was fit using MCMC. To examine model diagnostics and check for degeneracy, use the mcmc.di

summary(fit_erg1)

##
## =====
## Summary of model fit
## =====
##
## Formula: Y ~ edges + mutual
##
## Iterations: 20
##
## Monte Carlo MLE Results:
##      Estimate Std. Error MCMC % p-value
## edges    -1.761     0.205      0 <1e-04 ***
## mutual     2.319     0.412      0 <1e-04 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##      Null Deviance: 424 on 306 degrees of freedom
##      Residual Deviance: 333 on 304 degrees of freedom
##
## AIC: 337    BIC: 344    (Smaller is better.)
```

GOF

```
gof_1 <- gof(fit_erg1, GOF = ~idegree + odegree + triadcensus)
```



degree distributions: The fitted model generates degree distributions similar to the observed;

triad census: The fitted model generates a triad census similar to the observed, perhaps with a few discrepancies (021U,111D,111U and 201).

Incorporating transitivity in ERGMs

```
fit_erg2 <- ergm(Y ~ edges + mutual + transitive)
fit_erg3 <- ergm(Y ~ edges + mutual + triadcensus)
fit_erg4 <- ergm(Y ~ edges + mutual + triadcensus(10))
```

transitive: Includes the network statistic

$$t(\mathbf{y}) = \text{number of transitive triples}$$

where a “transitive triple is of the type” 030T, 120U, 120D or 300.
(note: none of these seemed to stand out for the monk GOF plots.)

triacdcensus: Includes 15 network statistics counting the number of triples of each type (excluding null triples).

triacdcensus(k): Includes the network statistic

$$t(\mathbf{y}) = \text{number of triples of type } k$$

Counting transitive triples

```
fit_erg2 <- ergm(Y ~ edges + mutual + transitive)
```

```
## Iteration 1 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 16.83  
## Iteration 2 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 16.83  
## Iteration 3 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 16.85  
## Iteration 4 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 16.87  
## Iteration 5 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 16.91  
## Iteration 6 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 16.98  
## Iteration 7 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 17.08  
## Iteration 8 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 17.24  
## Iteration 9 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 17.58  
## Iteration 10 of at most 20:  
## Convergence test P-value: 0e+00  
## The log-likelihood improved by 13.87  
## Iteration 11 of at most 20:  
## Convergence test P-value: 0e+00
```

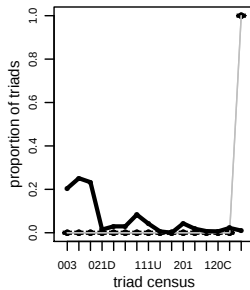
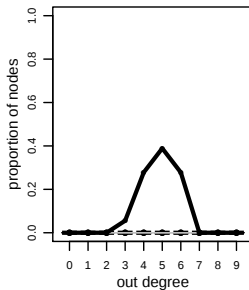
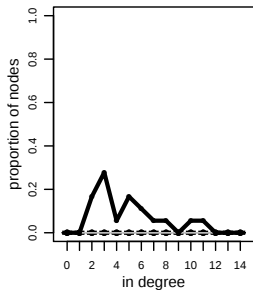
Counting transitive triples

```
as.matrix(simulate(fit_erg2))
```

```
##          ROMUL BONAVERN AMBROSE BERTH PETER LOUIS VICTOR WINF JOHN GREG HUGH
## ROMUL      0         1         1         1         1         1         1         1         1         1         1
## BONAVERN    1         0         1         1         1         1         1         1         1         1         1
## AMBROSE     1         1         0         1         1         1         1         1         1         1         1
## BERTH       1         1         1         0         1         1         1         1         1         1         1
## PETER       1         1         1         1         0         1         1         1         1         1         1
## LOUIS       1         1         1         1         1         0         1         1         1         1         1
## VICTOR      1         1         1         1         1         1         0         1         1         1         1
## WINF        1         1         1         1         1         1         1         0         1         1         1
## JOHN        1         1         1         1         1         1         1         1         0         1         1
## GREG        1         1         1         1         1         1         1         1         1         0         1
## HUGH        1         1         1         1         1         1         1         1         1         1         0
## BONI        1         1         1         1         1         1         1         1         1         1         1
## MARK        1         1         1         1         1         1         1         1         1         1         1
## ALBERT      1         1         1         1         1         1         1         1         1         1         1
## AMAND       1         1         1         1         1         1         1         1         1         1         1
## BASIL       1         1         1         1         1         1         1         1         1         1         1
## ELIAS       1         1         1         1         1         1         1         1         1         1         1
## SIMP        1         1         1         1         1         1         1         1         1         1         1
##          BONI MARK ALBERT AMAND BASIL ELIAS SIMP
## ROMUL      1         1         1         1         1         1         1
## BONAVERN    1         1         1         1         1         1         1
## AMBROSE     1         1         1         1         1         1         1
## BERTH       1         1         1         1         1         1         1
## PETER       1         1         1         1         1         1         1
## LOUIS       1         1         1         1         1         1         1
## VICTOR      1         1         1         1         1         1         1
## WINF        1         1         1         1         1         1         1
```

Goodness of fit

```
gof_2 <- gof(fit_erg2, GOF = ~idegree + odegree + triadcensus)
```



Model degeneracy

This strange phenomenon is (as far as I know) not a coding error:

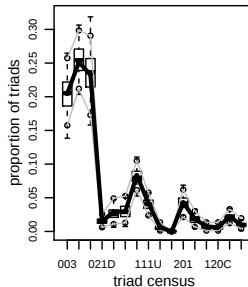
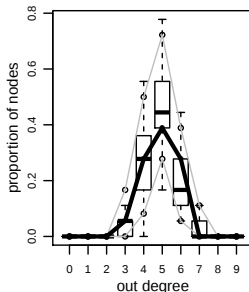
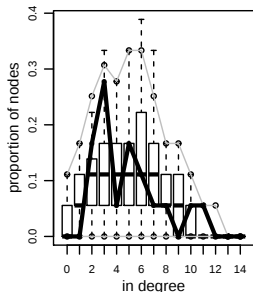
- MLEs for ERGMs sometimes produce **degenerate** distributions;
- A degenerate distribution puts most of its probability on the null or full graph;
- See “Assessing Degeneracy in Statistical Models of Social Networks” (Handcock, 2003) for more details.
- Recent research on Bayesian ERGM estimation avoids such degeneracy.

Incorporating transitivity in ERGMs

```
summary(fit_erg3)

##
## =====
## Summary of model fit
## =====
##
## Formula:   Y ~ edges + mutual + triadcensus
##
## Iterations: 20
##
## Monte Carlo MLE Results:
##           Estimate Std. Error MCMC % p-value
## edges           5.657      0.205    NA < 1e-04 ***
## mutual          -13.175      0.125    NA < 1e-04 ***
## triadcensus.012   -0.126      0.164    NA 0.44275
## triadcensus.102    0.723      0.281    NA 0.01069 *
## triadcensus.021D  -1.445      0.306    NA < 1e-04 ***
## triadcensus.021U  -0.402      0.363    NA 0.26948
## triadcensus.021C  -0.481      0.302    NA 0.11259
## triadcensus.111D   0.519      0.435    NA 0.23366
## triadcensus.111U  -0.765      0.176    NA < 1e-04 ***
## triadcensus.030T  -1.684      0.449    NA 0.00021 ***
## triadcensus.030C   -Inf         NA     NA      NA
## triadcensus.201   -0.116      0.207    NA 0.57399
## triadcensus.120D   0.231      0.474    NA 0.62569
## triadcensus.120U  -1.410      0.511    NA 0.00616 **
## triadcensus.120C  -0.982      0.466    NA 0.03577 *
## triadcensus.210   -0.814      0.320    NA 0.01148 *
## triadcensus.300   -1.112      0.818    NA 0.17515
```

Incorporating transitivity in ERGMs



- The degree distribution fit is improved, the triad fit is near perfect.
 - Is this latter fact surprising?
- The number of parameters in the model is quite large.
 - 15 additional parameters to represent 3rd order dependence.
- What about a reduced model?

Backwards elimination

```
summary(fit_erg4)

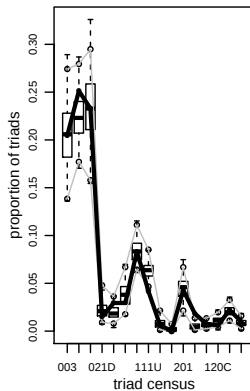
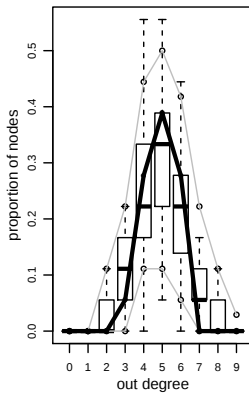
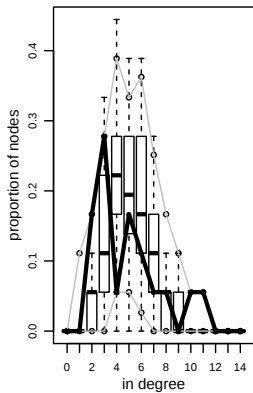
##
## =====
## Summary of model fit
## =====
##
## Formula:   Y ~ edges + mutual + triadcensus(c(2, 3, 6, 8, 10))
##
## Iterations: 20
##
## Monte Carlo MLE Results:
##           Estimate Std. Error MCMC % p-value
## edges           1.152      1.639    45  0.483
## mutual          -7.898      5.930    45  0.184
## triadcensus.102    0.772      0.381    42  0.043 *
## triadcensus.021D  -0.798      0.464    22  0.087 .
## triadcensus.111D   0.792      0.351    31  0.025 *
## triadcensus.030T  -0.706      0.625    21  0.259
## triadcensus.201    0.282      0.430    37  0.512
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##      Null Deviance: 424  on 306  degrees of freedom
## Residual Deviance: 312  on 299  degrees of freedom
##
## AIC: 326    BIC: 352    (Smaller is better.)
```

Backwards elimination

```
summary(fit_erg5)

##
## =====
## Summary of model fit
## =====
##
## Formula:   Y ~ edges + mutual + triadcensus(c(2, 6))
##
## Iterations: 20
##
## Monte Carlo MLE Results:
##           Estimate Std. Error MCMC % p-value
## edges          -0.619      0.549      14 0.2603
## mutual          -2.256      1.630      13 0.1672
## triadcensus.102   0.500      0.151      12 0.0010 **
## triadcensus.111D  0.542      0.192       1 0.0051 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Null Deviance: 424 on 306 degrees of freedom
## Residual Deviance: 317 on 302 degrees of freedom
##
## AIC: 325    BIC: 340    (Smaller is better.)
```

Goodness of fit

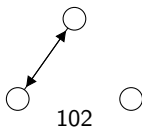
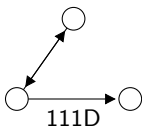


Fishing expeditions

Notice that `mutual` is no longer “significant”

- this does not mean that there is not much reciprocity in the network;
- it means reciprocity can be explained by a tendency for 102 and 111D triples.

Is such a tendency meaningful/interpretable?



Fishing expeditions

Comments:

Iterative model selection procedures

- produce parsimonious descriptions of the network dataset;
- produce p -values and standard errors that may be misleading:
 - in a large set of model statistics, some will **appear** significant due to chance.

Advice:

- descriptive modeling: choose a parsimonious, interpretable model.
- hypothesis testing: choose your models to reflect your hypotheses.