1

Statistical Testing: Beyond Basics

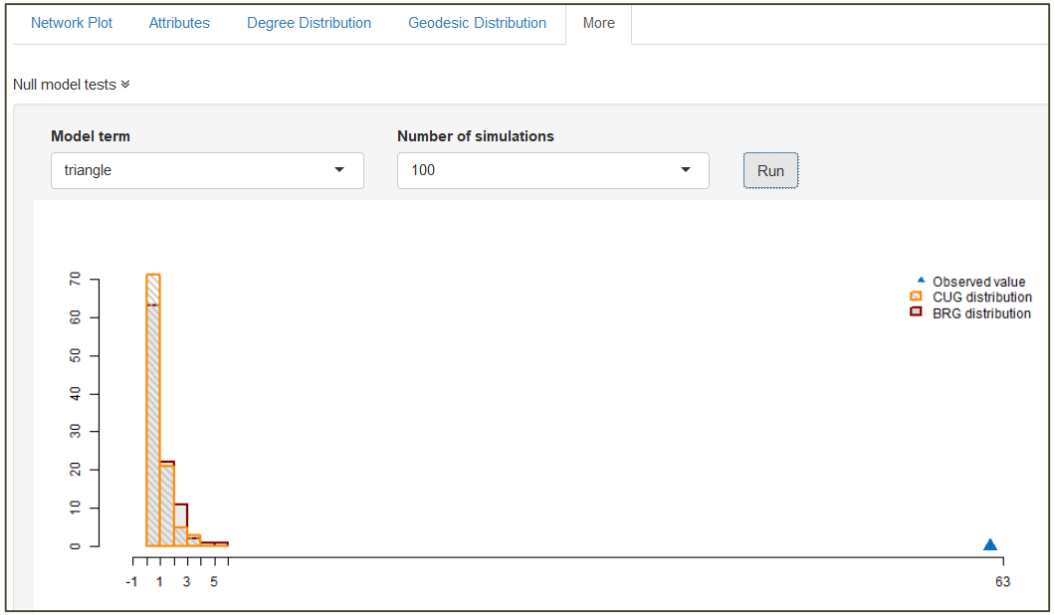
Can you control for more than just density?

What if you want to test more than one network feature?

And you want a model grounded in generative social theory?

… That's when you need ERGMs

Yes the observed triangle count is high



- But why?

… a simple null hypothesis test doesn't provide any insight about that.

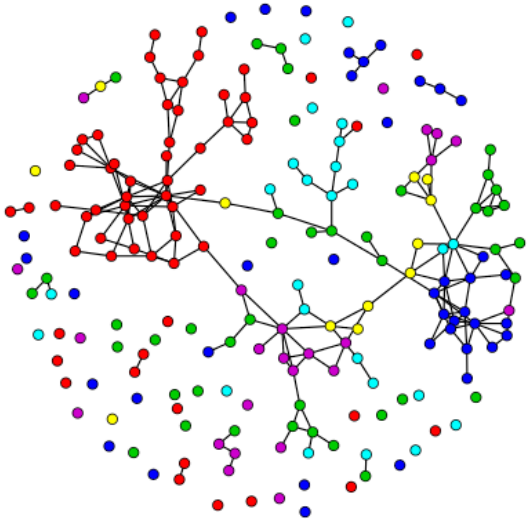
Limitations of simple null hypotheses

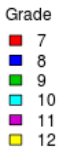
- If we are ***only*** interested in whether the triangle counts are different than expected given the density of the graph
  - One can use these simple null hypothesis tests
  - Like a t-test in traditional statistics
- But if we want to understand the underlying generative process, quantify the impact of each process on our network, and control for other network features …
  - This requires a more general statistical modeling framework

Motivation

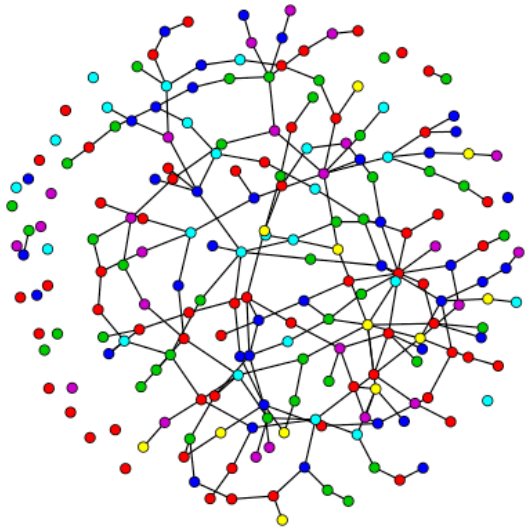
- *Why* are there so many more triangles?
- What do you see when color-coding the nodes by their attributes?

faux.mesa.high network





Simple random graph with the same tie probability



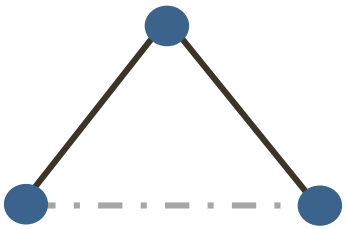
Friend of a friend, or birds of a feather?

Two theories about the process that generates triangles:

1. **Homophily**: People tend to chose friends who are like them, in terms of grade, race, etc. (*"birds of a feather"*), triad closure is a by-product

2. **Transitivity**: People who have friends in common tend to become friends (*"friend of a friend"*), triad closure is the key process

So, for three actors in the same grade



A cycle-closing tie may form due to transitivity

But it may be due instead to homophily

Transitivity and homophily are *partially* confounded

But not completely. Any tie may be classified by whether it is:

	Triangle forming:	
Within Grade:	*Yes*	*No*
Yes	Both	Homophily
No	Transitivity	*Neither*

The cells represent how the processes jointly influence that tie, so the distribution of ties in this table is informative.

This suggests we should be able to disentangle the two processes statistically

ERGMs: Basic idea

- We want to model the probability of a tie as a function of:
  - Nodal attributes (that influence degree and mixing)
  - The propensity for certain "configurations" (like triangles)
- The dyads may be dependent
  - Nodal attribute effects do not induce dyad dependence
  - But triad closure does
- So we model the joint distribution directly

Exponential Random Graph Model (ERGM)

Probability of observing a graph (set of relationships) y on a fixed set of nodes:

$$P(Y = y \mid \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}' \boldsymbol{g}(\boldsymbol{y}))}{k(\boldsymbol{\theta})}$$

where: $\mathbf{g(y)}$ = vector of network statistics

θ = vector of model parameters

$\mathbf{k(\theta)}$ = numerator summed over all possible networks on node set y

- Exponential family model
- Well understood statistical properties (≠ well understood models)
- Very general and flexible

Exponential Random Graph Model (ERGM)

Probability of observing a graph (set of relationships) y on a fixed set of nodes:

$$P(Y = y \mid \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}' \boldsymbol{g}(\boldsymbol{y}))}{k(\boldsymbol{\theta})}$$

If you're not familiar with this kind of compact vector notation, the numerator is just:

$$\exp(\boldsymbol{\theta}_1 x_1 + \boldsymbol{\theta}_2 x_2 + \cdots + \boldsymbol{\theta}_p x_p)$$

Kind of like a linear model, but a bit different (watch out for this later)

The conditional odds of a tie

The probability of the graph $P(Y = y \mid \boldsymbol{\theta}) = \dfrac{\exp(\boldsymbol{\theta}' \boldsymbol{g}(\boldsymbol{y}))}{k(\boldsymbol{\theta})}$ *can be re-expressed as*

The conditional log odds of a specific tie

$$logit\left(P\left(Y_{ij} = 1 \middle| \text{rest of the graph}\right)\right) = log\left(\frac{P\left(Y_{ij} = 1 \middle| \text{rest of the graph}\right)}{P\left(Y_{ij} = 0 \middle| \text{rest of the graph}\right)}\right)$$

$$= \boldsymbol{\theta}' \boldsymbol{\partial}\left(\boldsymbol{g}(\boldsymbol{y})\right)$$

where $\boldsymbol{\partial}\left(\boldsymbol{g}(\boldsymbol{y})\right)$ represents the change in $\boldsymbol{g}(\boldsymbol{y})$ when Y_{ij} is toggled between 0 and 1

This is an auto logistic regression (auto because of the possible dependence)

ERGM specification: $\boldsymbol{\theta}'\boldsymbol{g}(\boldsymbol{y})$

The $\boldsymbol{g}(\boldsymbol{y})$ terms in the model are summary "network statistics"

- Counts of network configurations, for example:
  1. Edges: $\sum y_{ij}$
  2. Within-group ties: $\sum y_{ij} I(i \in C, j \in C)$
  3. 2-stars: $\sum y_{ij} y_{ik}$
  4. 3-cycles: $\sum y_{ij} y_{ik} y_{jk}$

- A key distinction in the types of terms:
  - Dyad independent (1 & 2 are examples)
  - Dyad dependent (3 & 4 are examples)

ERGM specification: $\boldsymbol{\theta' g(y)}$

Model specification involves:

1. Choosing the set of network statistics $\boldsymbol{g(y)}$
   - From minimal : # of edges
   - To saturated: one term for every dyad in the network

   NB: statnetWeb allows you to choose from the list of terms and retrieve documentation for each one

2. Choosing "homogeneity constraints" on the parameter θ, for example, with edges:
   - all homogeneous
   - dyad specific (as fixed or random effects)

Definition of a network model

- This term is used loosely in the published literature
  - Individual/Agent-based models are often called network models
- There is some overlap
  - Network models are individual-based
  - And individual-based models create networks
- But we make a distinction
  - A network model has an explicit *model* for the network
  - You can tell, because the network is on the left hand side of the eqn:

$$P(network) = f(covariates)$$

… to StatnetWeb

Let’s explore the Florentine marriage network

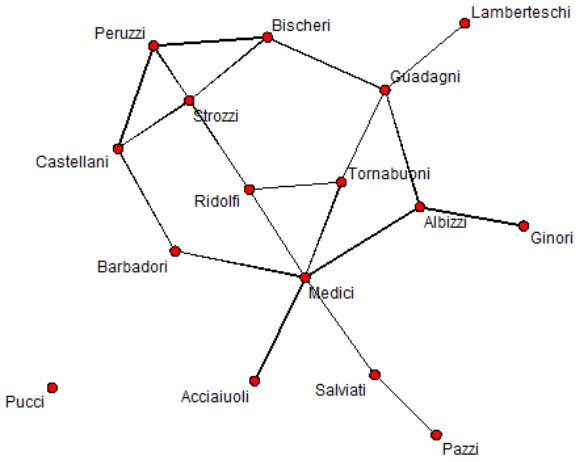
Small, so calculations are quick

You’ll return to faux.mesa.high for group lab

Flomarriage: Bernoulli Model

- Load the flomarriage network

Network of marriage ties between families in Renaissance Florence



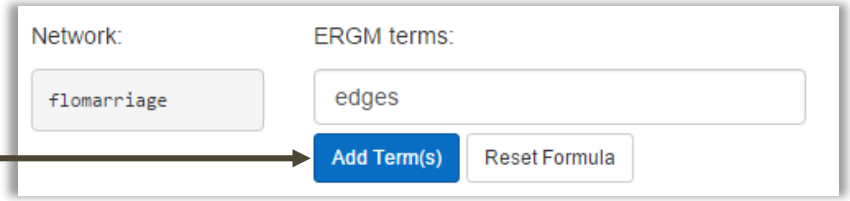
- The nodes have 4 attributes:

```
Vertex attribute names:
   priorates totalties vertex.names wealth
```

Flomarriage: Bernoulli Model

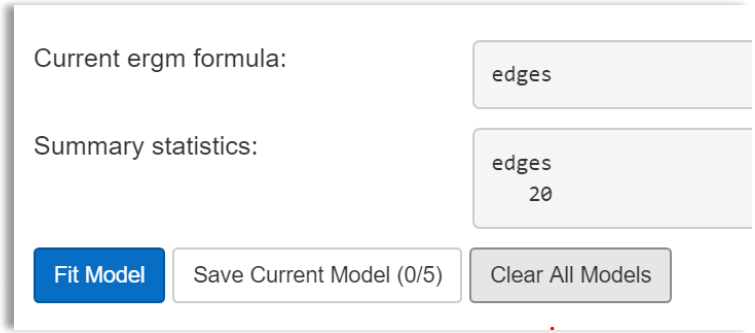
- Add edges to the ergm formula

Step 1 →



- Fit the model

Step 2 →



- What does this model imply? Homogeneous edge probability
  - Every tie is equally likely
  - Not a very interesting model

Interpreting the coefficients

- The log-odds of any tie existing is: $\theta = ln\left(\frac{p}{(1-p)}\right)$

  $= -1.609 \times \text{change in \# ties}$

  $= -1.609 \times 1$

- Corresponding probability:

  $= \frac{\exp(-1.609)}{1+\exp(-1.609)}$ $\quad p = \frac{e^{\theta}}{1+e^{\theta}}$

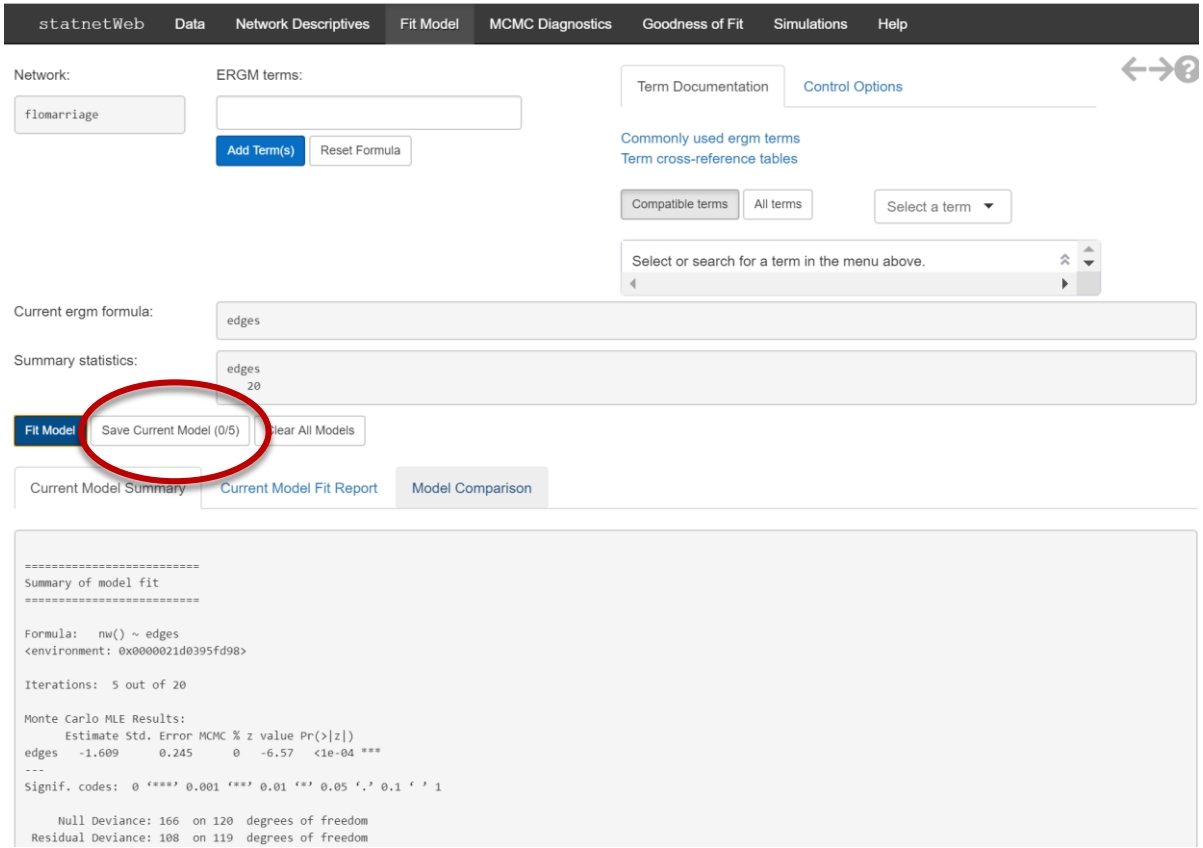
  $= 0.1667$

You can confirm that this is the density of the network

```
Call:
ergm(formula = ergm.formula())

Maximum Likelihood Results:

      Estimate Std. Error MCMC % z value Pr(>|z|)
edges  -1.6094     0.2449      0  -6.571   <1e-04 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 166.4  on 120  degrees of freedom
 Residual Deviance: 108.1  on 119  degrees of freedom

AIC: 110.1  BIC: 112.9  (Smaller is better. MC Std. Err. = 0)
```

Save this model

So we can compare it to the next models we will run.

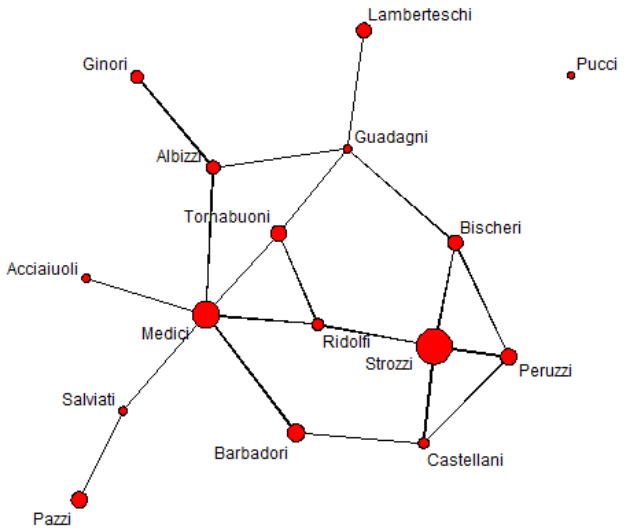


Click the Save Model tab

It will change to (1/5)

Flomarriage: Nodal covariates

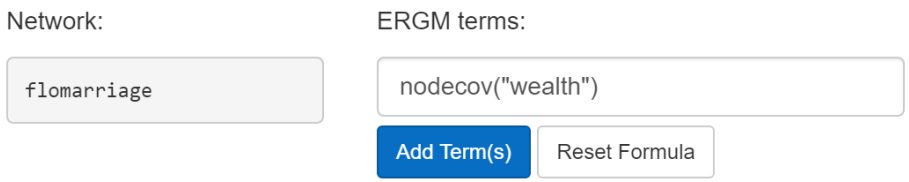
Flomarriage: Nodes sized by wealth



- What do you notice?
- We can test whether edge probabilities are a function of wealth
- This is a quantitative nodal attribute, so we use the ergm term “nodecov”

Flomarriage: Nodal covariates

- Type `nodecov("wealth")` to the ergm terms box, add the term, and fit the following model:



- There is a significant positive wealth effect on the odds of a tie

```
                Estimate Std. Error MCMC % z value Pr(>|z|)
edges          -2.594929   0.536056      0  -4.841   <1e-04 ***
nodecov.wealth  0.010546   0.004674      0   2.256   0.0241 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

- What does the positive coefficient mean?
  - Wealthy nodes have more ties
  - Note that the wealth effect operates on both nodes in a dyad.
  - But: Does not mean that there is homophily by wealth

Flomarriage: Nodal covariates

- The conditional log-odds of a tie between two actors is:

$$-2.59 \times \text{change in \# ties} + 0.01 \times \text{wealth of node 1} + 0.01 \times \text{wealth of node 2}$$

  - For a tie between two nodes with minimum wealth (3)

$$-2.59 + 0.01 \times (3 + 3) = -2.53$$

  - For a tie between two nodes with maximum wealth (146)

$$-2.59 + 0.01 \times (146 + 146) = 0.33$$

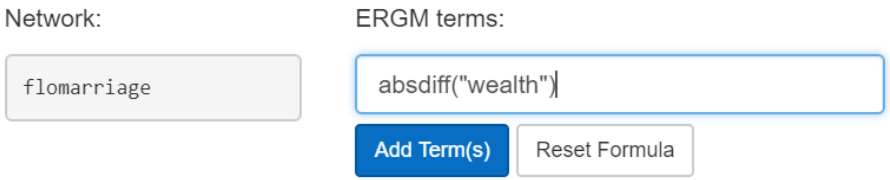
  - For a tie between nodes with maximum and minimum wealth

$$-2.59 + 0.01 \times (146 + 3) = -1.1$$

Save this model (2/5)

Flomarriage: mixing by wealth

- Type `absdiff("wealth")` to the ergm terms box, add the term, and fit the following model:



- There is a (small) positive effect on the odds of a tie

```
Call:
ergm(formula = ergm.formula())

Maximum Likelihood Results:

                Estimate Std. Error MCMC % z value Pr(>|z|)
edges          -2.527091   0.535994      0  -4.715   <1e-04 ***
nodecov.wealth  0.004506   0.006791      0   0.664    0.507
absdiff.wealth  0.011143   0.008950      0   1.245    0.213
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

This coefficient is not significant, and now `nodecov.wealth` is no longer significant

But this is a small network, and these terms are correlated.

Flomarriage: mixing by wealth

absdiff.wealth	0.011143	0.008950	0	1.245	0.213

- What does this positive coefficient mean?
  - We'll ignore the fact that it is not statistically significant for now

- That an increase in the *absolute difference in wealth* increases the odds of a tie.

This represents disassortative mixing on wealth.

The greater the wealth disparity between two families, the more likely the marriage.

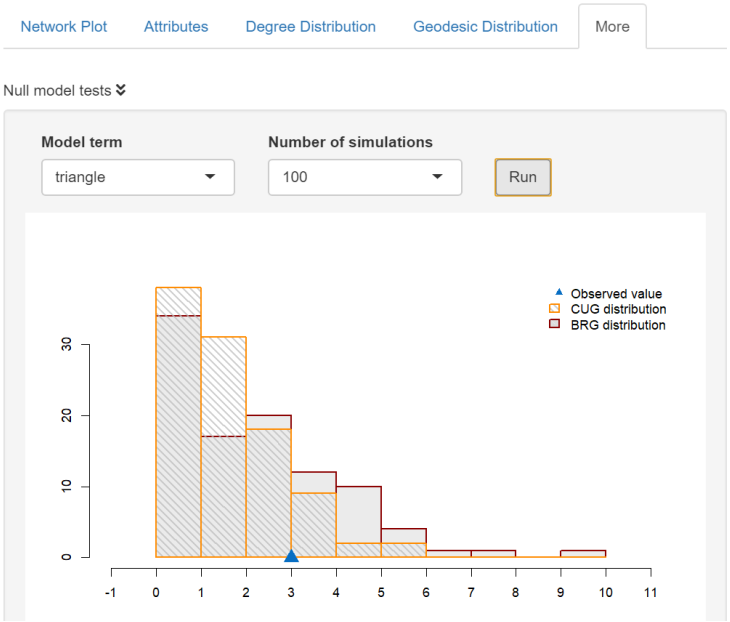
Save this model (3/5)

Flomarriage: Triads

- There were many more triangles than expected in the faux.mesa.high data

- What about flomarriage?

  The null hypothesis tests suggest # triangles is about what we would expect



Flomarriage: Triads

- The “triangle” term is a measure of clustering
  - Read the documentation for the triangle term for more info

- Here we’ll fit a non-nested model, since this is a small network
  - Fit the model `edges + triangle`

```
Call:
ergm(formula = ergm.formula())

Monte Carlo Maximum Likelihood Results:

         Estimate Std. Error MCMC % z value Pr(>|z|)
edges     -1.6758     0.3499      0  -4.789   <1e-04 ***
triangle   0.1511     0.5909      0   0.256    0.798
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 166.4  on 120  degrees of freedom
 Residual Deviance: 108.1  on 118  degrees of freedom

AIC: 112.1  BIC: 117.6  (Smaller is better. MC Std. Err. = 0.008452)
```

Note: MC MLE now

As expected, triangle term not significant

But we’ll work through the interpretation anyway…

Save this model (4/5)

Flomarriage: Triads

triangle	0.1511	0.5909	0	0.256	0.798

Now how to interpret the coefficients?

Conditional log-odds of two actors having a tie:

$$(-1.68 \times \text{change in the \# of ties}) + (0.15 \times \text{change in \# of triangles})$$

always=1

how many triangles can one tie change?

- For a tie that will create zero triangles $-1.68 + 0 = -1.68$
- One triangle $-1.68 + (0.15 \times 1) = -1.53$
- Two triangles $-1.68 + (0.15 \times 2) = -1.38$

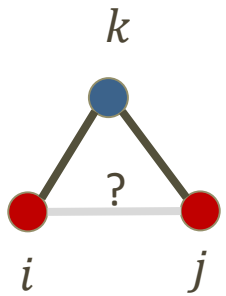
Still unlikely, but a bit less so

Estimation, part 1

Now the fit takes longer. Why?

- Because `triangle` is a "**dyad-dependent**" term

- Now the probability of a tie between nodes i and j depends on whether it will close a triangle
  - And that depends on whether i and j share any other partners
  - That is, their ties/non-ties *with every other node in the network*



Not just their ties with k,

Their ties with *every other node* must be checked to see if those two other legs of the triangle are in place

Dyad dependent terms change estimation

- When all model terms are “dyad-independent”
  - ergm uses the same algorithm as logistic regression
  - Usually very quick

- When you add a dyad dependent term
  - This changes the estimation algorithm to MCMC
    - Markov Chain Monte Carlo
  - This takes longer

What is MCMC?

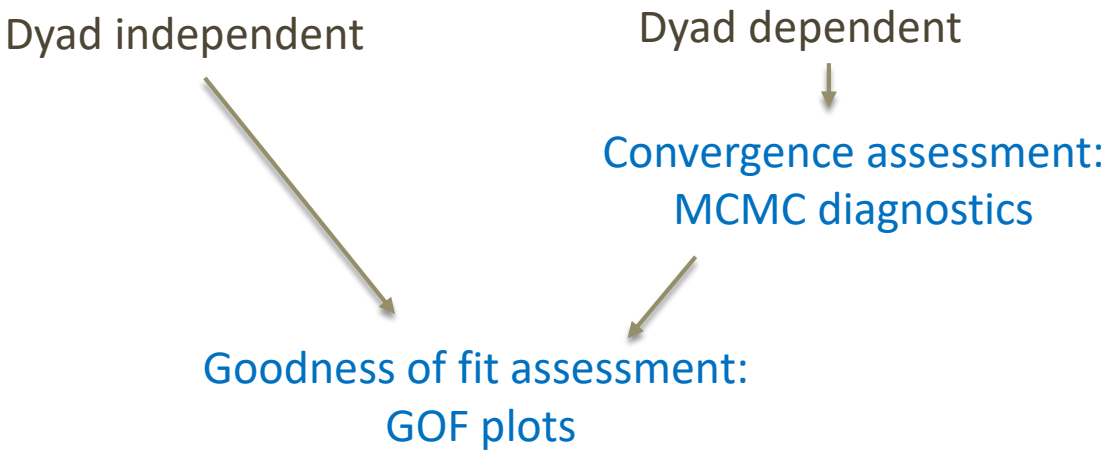
- A computationally intensive estimation algorithm
  - Set a starting value for your coefficients
  - Simulate networks by proposing ties between nodes: “toggles”
    - Some are accepted, some not, based on the probability defined by your model with the candidate coefficients
    - Every 1000 toggles, grab the network and calculate the netstats, and repeat
    - After 1000 sampled networks:
      - Compare the observed netstats
      - To the distribution of the netstats from this run
    - Adjust the coefficients as indicated (higher, or lower)
    - Repeat
- Note: this is a network simulation algorithm

ERGM fit assessment

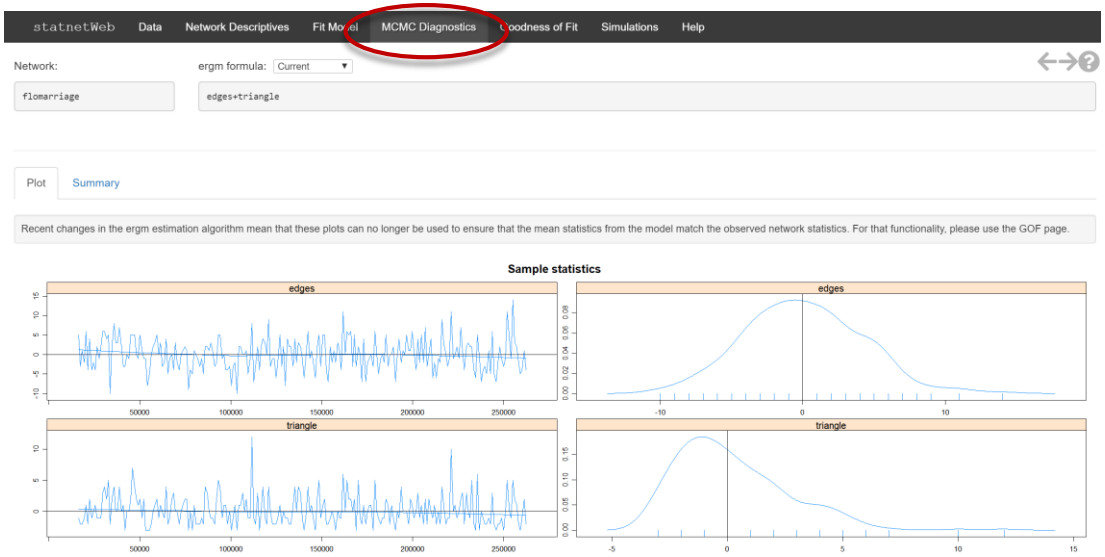
Fitting and diagnosing a model

- The steps depend on the type of model you have
  - If you have a dyad dependent model, you first check convergence

- In both cases you end with goodness of fit:



What are MCMC Diagnostics?



MCMC Diagnostics tell us if the estimation algorithm is mixing well, and converged to the target value

These look ok

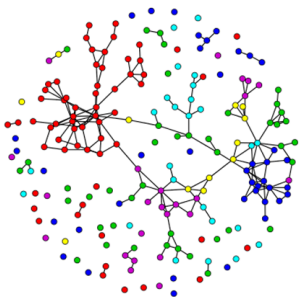
The **traceplots** on the left show random walks around the target value (you're looking for a fuzzy caterpillar)

The distribution of sampled statistics on the right is roughly centered on the target values

Plots are from the last run in the MCMC chain

Goodness of Fit (GOF)

- Traditional GOF stats can be used
  - AIC, BIC are included in the model summary
- We also take another approach
  - Does the model reproduce other network properties that were not included as model terms?
  - We use the full distributions of 3 “higher order” statistics:
    - Degree
    - Shared partners (local clustering)
    - Geodesic distances (global clustering)



DATA

MODEL

ESTIMATED COEFFICIENTS

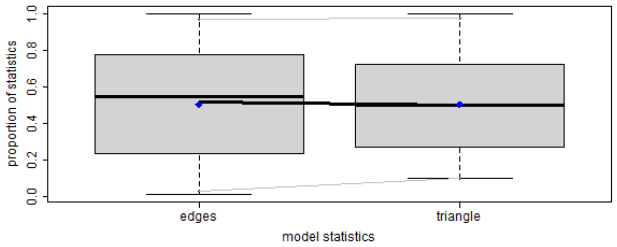
SIMULATED DATA
(draws from the prob. dist.)

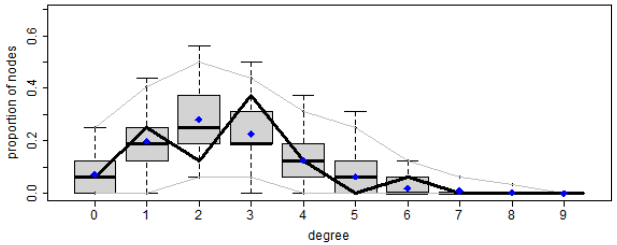
HIGHER ORDER GRAPH STATISTICS OF DATA

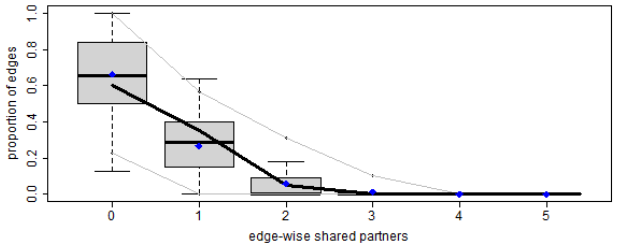
HIGHER ORDER GRAPH STATISTICS OF SIMULATED DATA

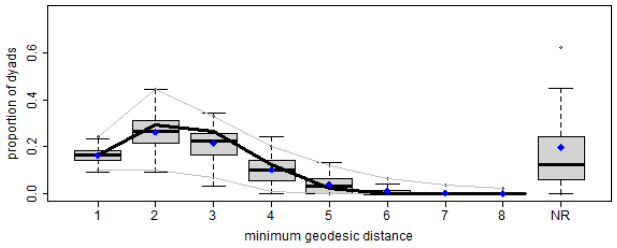
GOODNESS OF FIT OF MODEL TO DATA

GOF plots in statnet (defaults)









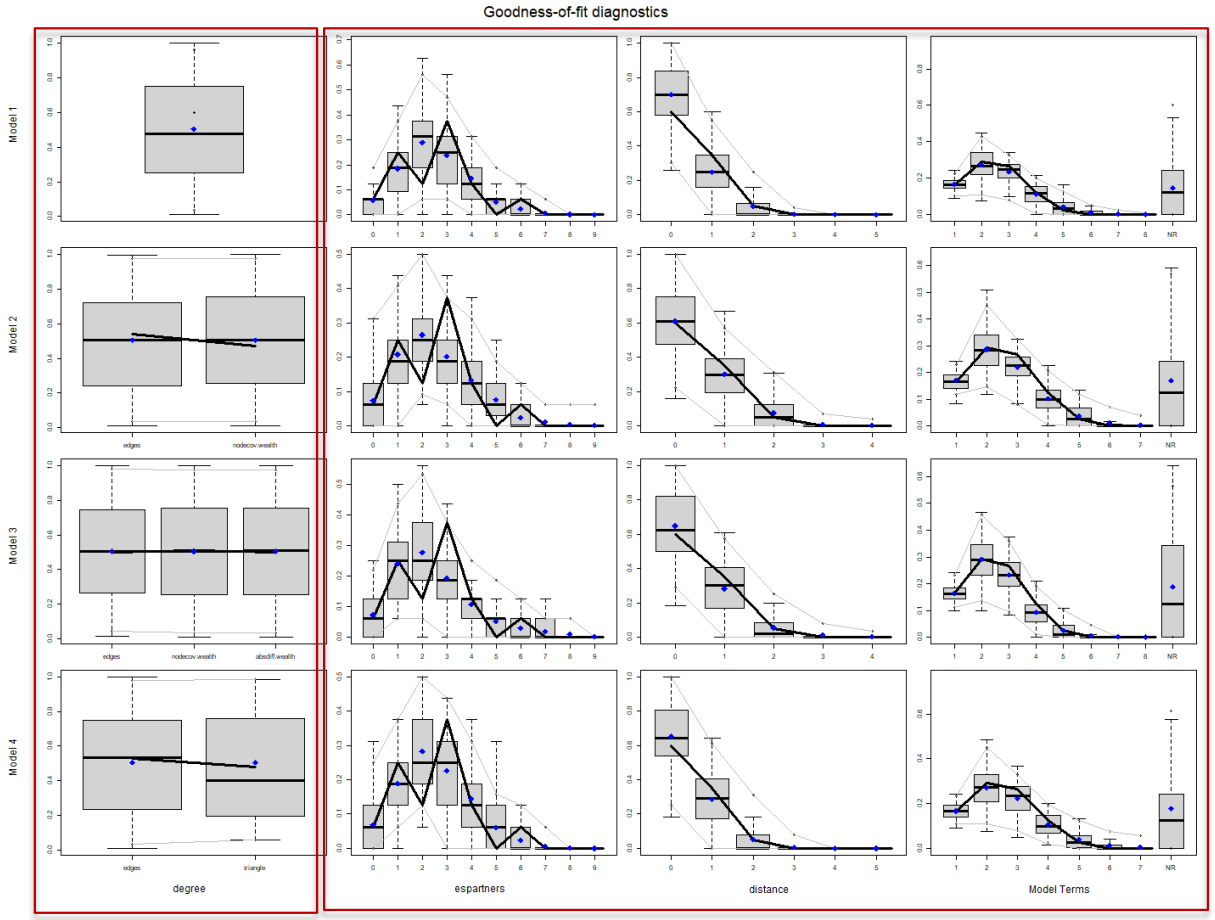
- The top plot is the model terms

Calibration assessment

- The bottom 3 plots are the higher order stat distributions
  - Degree
  - Shared partners
  - Geodesics

Validation assessment

Flomarriage: GOF for our 4 models



The model terms

The higher order fit statistics

Bottom line:

The edges only model does pretty well

The other terms don't add much

Makes sense, as they were not significant...

To the Lab

Fitting ERGMs with statnetWeb

Instructions are in the online course materials
Day 2, session 3

Individual work, but a group Slack report of findings

Selected references

Journal of Statistical Software (v42) 2008 – Eight papers on the **statnet** software; covering theory, algorithms and usage

Hunter DR, Goodreau SM, Handcock MS. Goodness of fit of social network models. (2008) J Am Stat Assoc. 103(481):248-58. doi: 10.1198/016214507000000446. PubMed PMID: WOS:000254311500029.

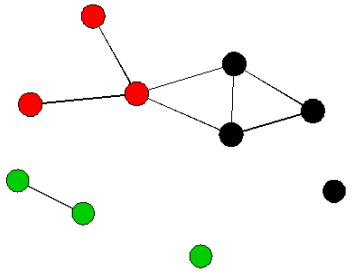
Goodreau, S., et al. (2009). "Birds of a Feather, or Friend of a Friend? Using Statistical Network Analysis to Investigate Adolescent Social Networks." *Demography 46(1): 103–125.*

Appendix

1. Descriptions of some common terms used in ERG network models, with simple examples to help show how the network statistics for each term are calculated

2. A bit more on MCMC estimation

ergm terms commonly used in EpiModel

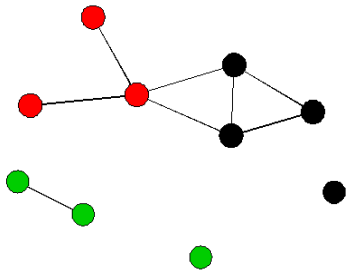


undirected network of 10 nodes, including nodal attribute “color”, with values:

1=black, 2=red, 3=green

Degree related terms	Calculation of network statistic	Unit for counting	Statistic Value(s)
`~edges`	# of edges	edges	8
`~nodefactor("color")`	Sum of degrees for nodes of each color	nodes/edges*	[8,] 6, 2
`~nodefactor("color", levels= -2)`	Sum of degrees for nodes of each color, using level 2 as the reference category	nodes/edges*	8, [6,] 2
`~degree(0)`	# of nodes of degree 0	nodes	2
`~degree(2:5)`	# of nodes of degrees 2, 3, 4, 5 each	nodes	1, 2, 1, 0
`~concurrent`	# of nodes of at least degree 2	nodes	4

ergm terms commonly used in EpiModel

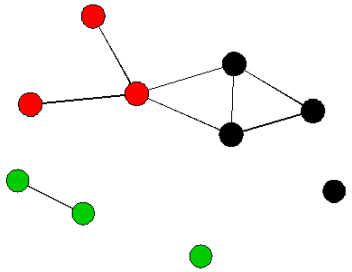


undirected network of 10 nodes, including nodal attribute "color", with values:

1=black, 2=red, 3=green

Mixing related terms (on color)	Calculation of network statistic	Unit	Statistic Value(s)
`~nodematch("color")`	# of edges between nodes of same color	edges	6
`~nodematch("color", diff = TRUE)`	# of edges between nodes of same color, for each color	edges	3, 2, 1
`~nodemix("color", base=1)`	# of edges between nodes of each color combo	edges	[3,] 2, 2, 0, 0, 1
`absdiff("color")` Note: this uses the values 1, 2 and 3	Sum of the difference in values of node color for every tie	edges	2

Common triad terms for ergms



undirected network of 10 nodes, including nodal attribute “color”, with values:

1=black, 2=red, 3=green

Triad related terms	Calculation of network statistic	Unit	Statistic Value(s)
`~triangle`	# of triangles (beware!)	triangles	2
`~gwesp(0)`	# of edges in at least one triangle	edges	5
`~gwesp(∞)`	# of edges in triangles total (=3 * # triangles)	triangles	6

These are just examples

- There are over 100 built-in terms in the ergm package.
  - They are documented, and have an interactive search utility
  - In the R console window type either of the commands below:

```
> ?"ergm-terms"

> vignette('ergm-term-crossRef')
```

  - You can also access the vignette online here
- And there is a package for building your own terms
  - ergm.userterms
  - With a tutorial

2. A bit more on MCMC

- MCMC MLE is used in many different fields now
  - Not just network analysis
  - Foundation for most Bayesian estimation
  - And anytime you have dependent data

- Relatively recent development
  - The theory preceded the computational feasibility…
  - Nice review of the history: https://arxiv.org/pdf/0808.2902.pdf

Why it works (in one slide)

- There is no "closed form" or analytic solution for the estimated coefficients (as there is in OLS: $\beta = (X'X)^{-1}(X'Y)$)

- Instead, we rely on a defining property of Maximum Likelihood Estimates (MLEs) for exponential family models
  - At the MLE of the coefficients:

    expected values of the statistics under the model = the observed statistics

- And we find these MLEs using an iterative search algorithm
  - A "Markov Chain Monte Carlo" (MCMC) algorithm
    - Start with some initial θ values, simulate a sample of networks from those values
    - Compare the means of the simulated statistics to the observed values
    - Update the values of θ based on the deviations
    - Repeat until the (expected – observed) < epsilon

(ok, I needed 2 slides)

- What does it mean to "simulate networks from those values"?
  - Pick a dyad at random
  - Toss a coin to set the tie status
    - The probability of the tie is determined by the model
    - And the details of the MCMC sampling algorithm *(Gibbs, Metropolis, Metropolis-Hastings)*
  - Repeat (many many many times)

- This produces a Markov Chain of networks
  - Sample from this chain, every 1000th element (say)

- Calculate the mean of the model statistics from this sample
  - And compare the this mean to the observed network statistics