Social Network Analysis
ERGM 2

Robin Burke

DePaul University

Chicago, IL

Thanks to Carter Butts for some slides and illustrations

Exponential Random Graph Model

θ = [p1, p2, p3 ... pk]
A vector of parameters
What we are trying to fit

t(g) = [t1(g), t2(g), t3(g), ... tk(g)]
t(g) = a vector of computed properties of g
What we are measuring about g

$$P(g|\theta, t, \gamma) = \frac{e^{\boxed{\theta^T t(g)}}}{\sum_{g' \in \gamma} e^{\theta^T t(g')}}$$

Dot product of two vectors = scalar value

g = the graph we observed

γ = the set of all possible graphs of interest

ergm package

- Dedicated statnet package for fitting, simulating models in ERG form
- Basic call structure
  - `ergm(net~term1(arg)+term2(arg)...)`

- net
  - `network` object
  - we will need to convert from `igraph`
- term1, term2, etc.
  - terms to use (see ?”ergm-terms”)
  - arg: where relevant, arguments to the term functions
- Output: `ergm` object
  - Summary, print, and other methods can be used to examine it
  - `simulate` command can also be used to take draws from the fitted model

Terms

- We have to choose what terms to fit
  - these are the f(g) elements
  - what elementary processes we think might explain what we see
  - Many options
- Kind of like regression
  - what variables will be part of our model?

Density

- Number of nodes is assumed to be fixed
- Almost always want to control density
  - otherwise the fitting produced will add many edges
  - to satisfy other constraints
  - overfitting problem
- edge term
  - says models are more likely
  - if they have the same number of edges
  - as the observed graph

Edge term

- ergm(net~edges)
  - This is the same as an ER random graph

- Learn a single parameter
  - related to the edge probability, p

Assortativity

- nodemix term
  - nominal only
  - k possible values
- Select an attribute
  - creates k(k-1)-1 terms
- Learns a parameter for an edge
  - between nodes with different combinations of values

Example: Dolphin sex

- Values: male / female / unknown
- ergm(net~edges+nodemix("Sex"))

- Parameters

  - edge
  - male / female
  - male / unknown
  - female / female
  - female / unknown
  - unknown / unknown

male/male is the "base" case

Assortativity scalar

- Use edge covariate term
  - edgecov

- For example

  - age

  - Adds the covariate term $\Sigma_i \Sigma_j \mathbf{y}_{ij} X_{ij}$
  - where is X is some attribute
- Adds up the covariance
  - for the whole network

Triangle trouble

- It is useful to measure the tendency of a network to form triangles
  - global transitivity

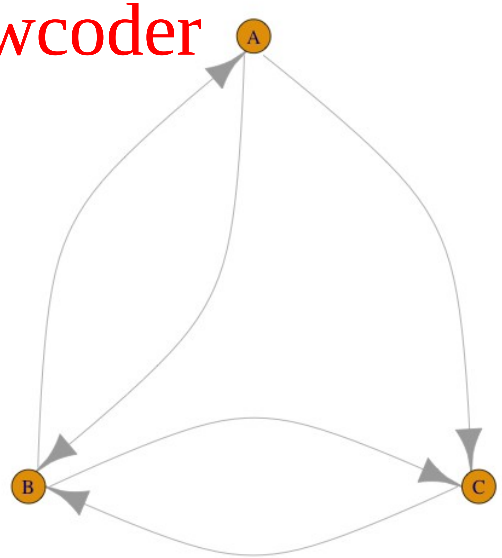
- Problem
  - triangle count is “brittle”
  - adding a single edge can change the total number of triangles significantly
  - models with triangle terms often fail to converge
- But
  - we still want to capture transitivity

Alternatives

- In a directed network
  - you can use terms from the triadic census
- In any network
  - you can use GWESP
  - wait for it...

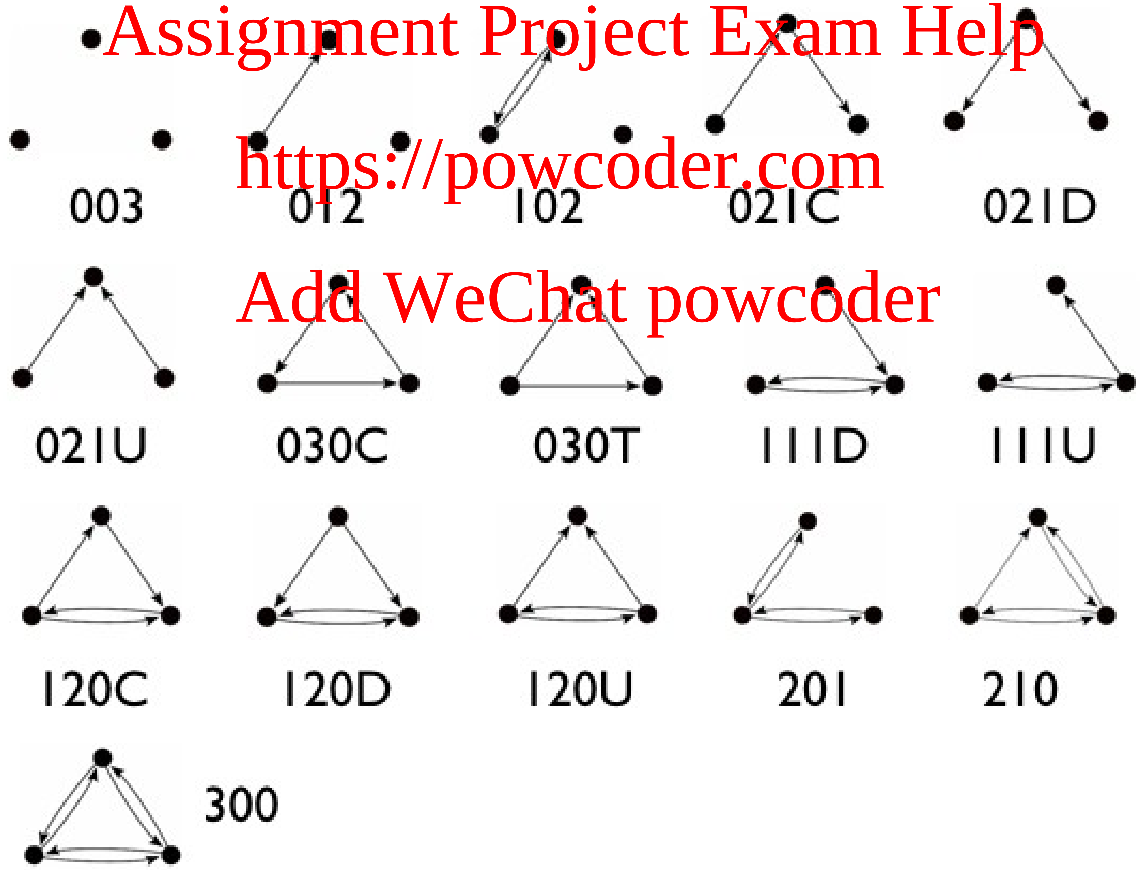
Triadic census (directed)

- Different ways that three individuals can be connected
- “Love triangle”
  - type 210



Triadic census (directed)

- 16 categories of "motifs"

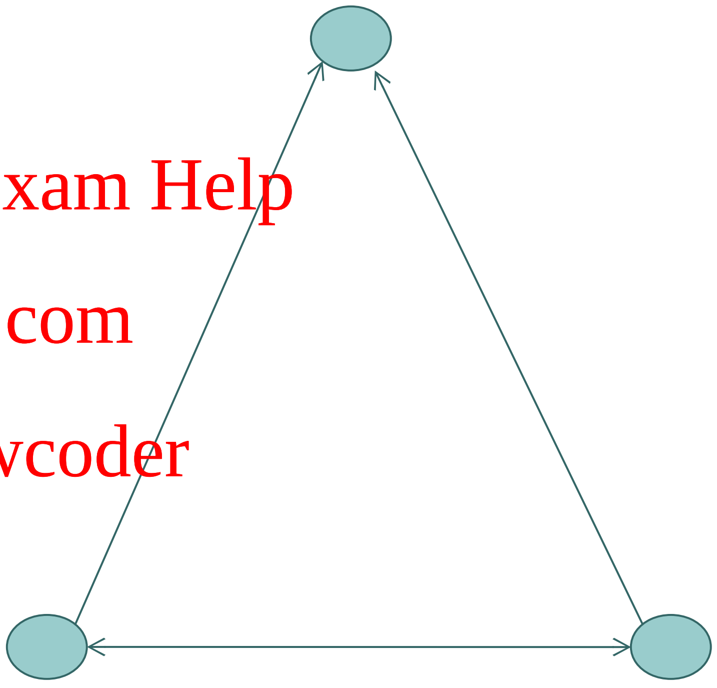


Triadic census naming

- Three digits for the three dyads
  - null
  - asymmetric
  - mutual
- Letter to distinguish variants
  - Up
  - Down
  - Transitive
  - Cyclic

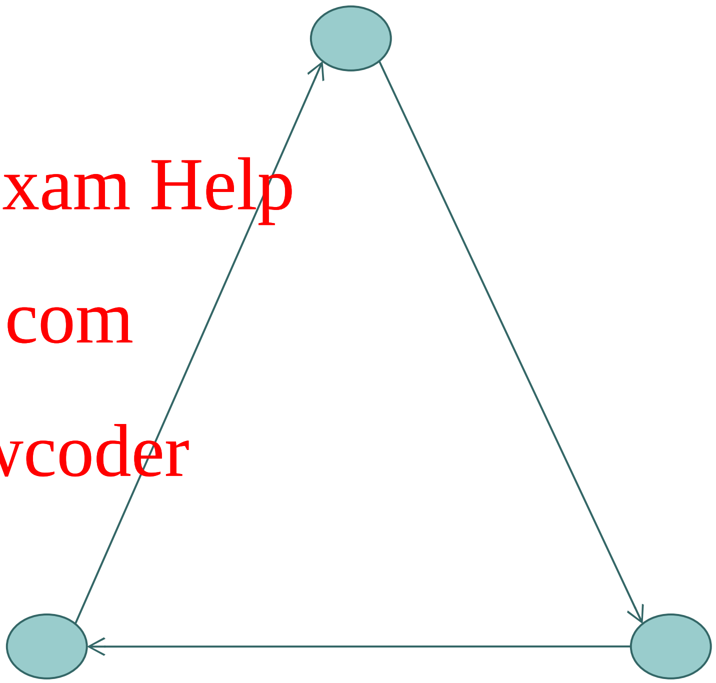
Example: 120U

- 1 mutual pair
- 2 asymmetric pair
- 0 null pairs
- U = up
  - both asymmetic edges point to the non-mutual node



Example: 030C

- 3 asymmetric edges
- arranged in a cycle



Triadic census

- Higher numbers =
  - greater density
- In igraph
  - triad_census
- Returns in order
  - 003, 012, 102, 021D, 021U, 021C, 111D, 111U, 030T, 030C, 201, 120D, 120U, 120C, 210, 300

Example

- Krackhardt friends vs advice
- Difference in density
  - 0.452 (advice)
  - 0.243 (friend)

Motif	Friend	Advice
003	27	74
012	102	153
102	34	90
021D	27	160
021U	14	86
021C	25	49
111D	26	59
111U	31	101
030T	21	190
030C	3	2
201	16	72
120D	10	62
120U	7	78
120C	14	17
210	7	107
300	0	30

hierarchy

transitivity

reciprocity

Triadic Terms

- A number of terms related to the triadic census
  - triadcensus() lets you pick specific ones you want to count
  - transitive() counts the transitive triads
  - 120D, 030T, 120U and 300
  - etc.
- Issues
  - Only for directed networks
  - These sometimes have the same problems with convergence
    - brittleness

Edgewise shared partner distribution

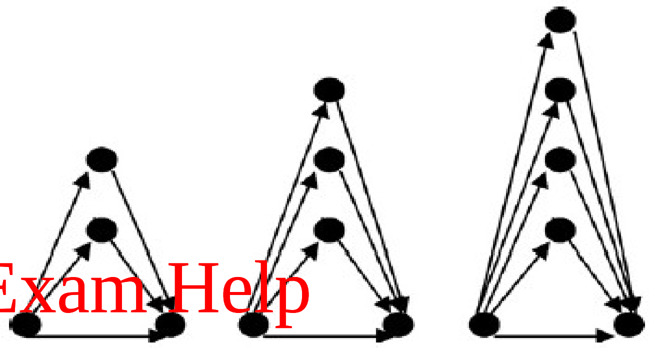


Fig. 4. Directed 2-, 3- and 4-triangles.

- EP_i
  - number of times that a pair of nodes has an edge and *i* other nodes to which they are both connected
- EP_1
  - just counts triangles
- EP_2
  - counts pairs of triangles sharing an edge

ESP distribution

- The counts of each such EP motif
  - EP1, EP2, EP3, EP4 ...
- Just like degree distribution

Why do this?

- The problem with trying to match the number of triangles
  - it is a hard constraint
  - influenced by every addition of an edge
- ESP distribution
  - is a softer constraint
  - adding an edge (usually) doesn't change the distribution that much

ESP measures transitive closure

- Quantitative measure of transitive closure
  - if there is transitive closure,
    - links become more likely as you move up the distribution
    - “the more triangles would be closed by a given link, the more likely that link is to exist”
- Transitive closure is an influence only up to a point
  - if A and B are connected to 10 shared partners, but are not connected
  - adding 10 more partners doesn’t make the edge 10 times more likely
  - maybe they just don’t like each other

GWESP

- Geometrically Weighted Edgewise Shared Partner
  - A single value
  - Tries to capture the overall triangular tendency

GWESP

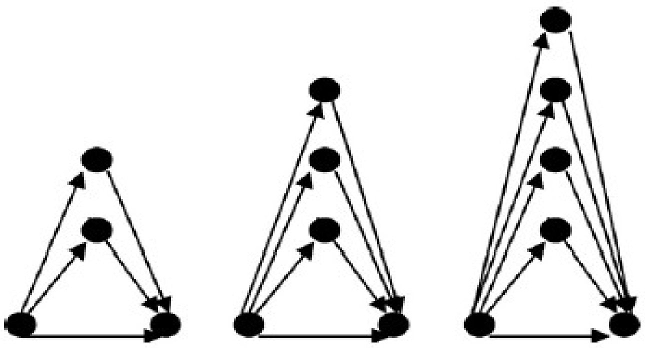


Fig. 4. Directed 2-, 3- and 4-triangles.

- $EP_i(y)$ is the number of edges in y that share exactly i neighbors

- GWESP sums over the whole distribution

  - Weighted so that each additional triangle counts for less
- Need the parameter τ but that can be fitted, too

$$v(y;\tau) = e^{\tau} \sum_{i=1}^{n-2} \left[1 - \left(1 - e^{-\tau}\right)^{i} \right] \mathrm{EP}_i(y)$$

GWESP

- Hard to interpret the associated parameter
- But it is basically trying to model the probability of edges
  - based on how likely they are to close open triangles

ERGM Terms

- Many variants
  - You can write your own (!!)
  - new terms being added to the package all the time
- You want the simplest model
  - exotic terms only when you need them
- A lot to learn here

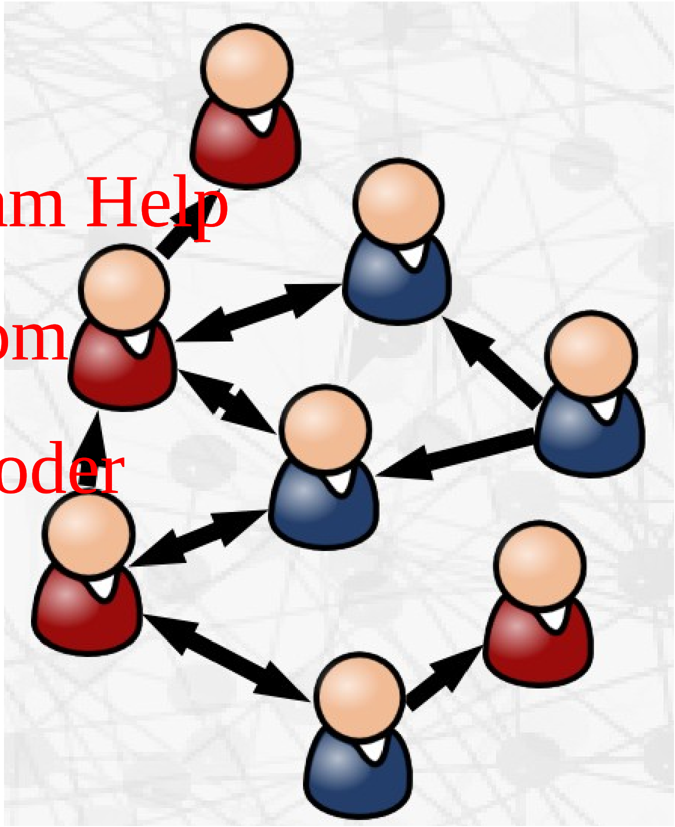
Social Network Analysis
ERGM Example

Robin Burke
DePaul University
Chicago, IL

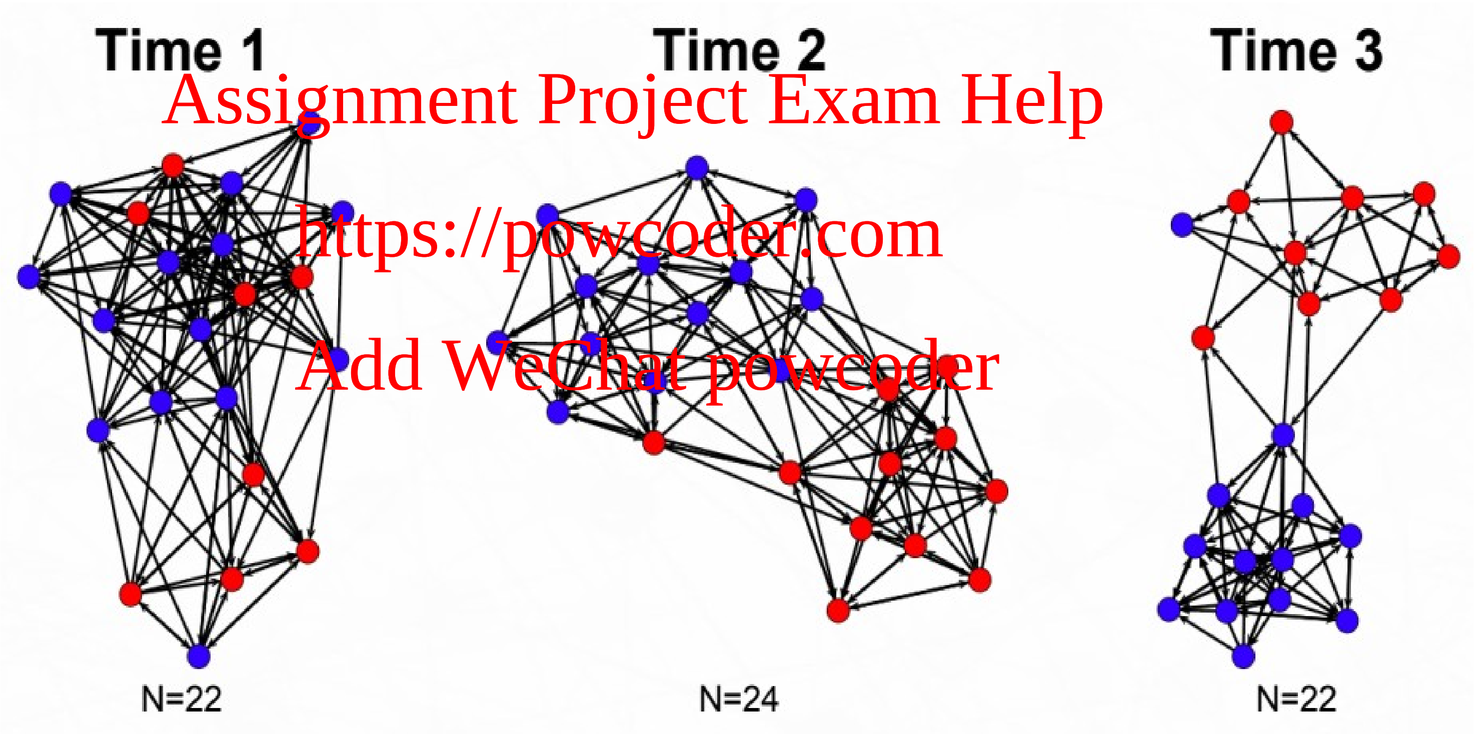
Thanks to Carter Butts for some slides and illustrations

Example: The Reds and The Blues

- A community w/two groups - the "Reds" and the "Blues"
- Question
  - exploring cooperation and trust in the community
  - during a period of upheaval
- We can observe networks of trust/friendship within representative subgroups....



Polarization: Why?



Polarization

- How to describe this process?
  - remember directed data
  - assortativity
  - mutuality
  - triangle-formation

Note

- Time-based ERGM
- Treat the probability of edges forming and dissolving
- Based on the prior state
- We won’t get to that in this class
- Model each network separately
  - look at differences

Also note

- There is a lot to say about the internals of ERGM fitting
- We will talk more about this next week

Appropriate ergm terms

- edges
  - you always want to limit the number of edges
  - otherwise the tendency will be to create lots of edges to satisfy the other constraints
- nodemix()
  - see how the colors mix
  - 4 terms
  - red -> blue, blue -> red, red -> red, blue -> blue
  - but one is “base” (dependency)
- mutual
  - counts how often an edge is reciprocated
- triangle
  - degree of common friendships
- gwesp
  - similar to triangle, but may be better behaved

Models

```
net3.m1 <- ergm(net3 ~ edges)
net3.m2 <- ergm(net3 ~ edges + nodemix("blue"))
net3.m3 <- ergm(net3 ~ edges + mutual)
net3.m4 <- ergm(net3 ~ edges + gwesp)
net3.m5 <- ergm(net3 ~ edges + triangle)
net3.m6 <- ergm(net3 ~ edges + nodemix("blue") + mutual)
net3.m7 <- ergm(net3 ~ edges + nodemix("blue") + gwesp)
net3.m8 <- ergm(net3 ~ edges + nodemix("blue") + triangle)
net3.m9 <- ergm(net3 ~ edges + nodemix("blue") + mutual + gwesp)
net3.m10 <- ergm(net3 ~ edges + nodemix("blue") + mutual + triangle)
net3.m11 <- ergm(net3 ~ edges + nodemix("blue") + gwesp + triangle)
net3.m12 <- ergm(net3 ~ edges + nodemix("blue") + mutual + gwesp + triangle)
net3.m13 <- ergm(net3 ~ edges + mutual + gwesp)
net3.m14 <- ergm(net3 ~ edges + mutual + triangle)
net3.m15 <- ergm(net3 ~ edges + mutual + gwesp + triangle)
```

Some models fail

- too much dependence between terms
  - heavily correlated
- Example


```
> net3.m5 <- ergm(net3 ~ edges + triangle)
Starting maximum likelihood estimation via MCMLE:
Iteration 1 of at most 20:
The log-likelihood improved by 17.22
Iteration 2 of at most 20:
The log-likelihood improved by 7.232
Iteration 3 of at most 20:
Error in ergm.MCMLE(init, nw, model, initialfit = (initialfit <-
NULL),  :
  Unconstrained MCMC sampling did not mix at all. Optimization cannot
continue.
```

Let’s look at our model fits

- Model ordered by AIC

Filter

	formula	model	AIC
1	net3 ~ edges + mutual	3	590.706956507602
2	net3 ~ edges + nodemix("blue") + mutual	5	597.937719067057
3	net3 ~ edges	1	616.98612127526
4	net3 ~ edges + nodemix("blue")	2	623.827641030355
5	net3 ~ edges + nodemix("blue") + mutual + gwesp	8	1106.40529316562
6	net3 ~ edges + mutual + gwesp	9	1543.49360296337
7	net3 ~ edges + gwesp	4	1668.79832292395
8	net3 ~ edges + nodemix("blue") + gwesp	6	2878.26168511217
9	net3 ~ edges + nodemix("blue") + triangle	7	12628.0590872257

Diagnostics

- Very important
- Tell you if you can trust the model results
- Stay tuned for more discussion next week!

Wrapping up

- Consider this model
  - net ~ edges + nodemix("blue", base=c(1,4)) + mutual
- Pretty good fit for net4 and net5
  - close to the best for net3

Best model

- net ~ edges + nodemix(“blue”, base=c(1,4) + mutual
- Consists of 4 parameters
  - edges
    - density
  - mutual
    - tendency of an edge to complete a reciprocal relation
  - Red -> Blue
    - nodemix.blue.0.1
  - Blue -> Red
    - nodemix.blue.1.0
- Other assortative terms ignored
  - probably too correlated with total edges or with mutual edges

What did we learn?

net3

weakly positive

	Estimate	Std. Error	MCMC %	p-value	
edges	-1.1701	0.1709	0	<1e-04	***
mix.blue.1.0	0.1115	0.2382	0	0.640	
mix.blue.0.1	0.1748	0.2352	0	0.458	
mutual	1.4965	0.2927	0	<1e-04	***

Estimate Std. Error MCMC % p-value

net4

negative

	Estimate	Std. Error	MCMC %	p-value	
edges	-0.9026	0.1966	0	<1e-04	***
mix.blue.1.0	-1.4138	0.3067	0	<1e-04	***
mix.blue.0.1	-1.9715	0.3513	0	<1e-04	***
mutual	1.8244	0.3326	0	<1e-04	***

net5

more negative

	Estimate	Std. Error	MCMC %	p-value	
edges	-0.9728	0.2158	0	< 1e-04	***
mix.blue.1.0	-1.8158	0.4374	0	< 1e-04	***
mix.blue.0.1	-3.2587	0.7296	0	< 1e-04	***
mutual	1.4113	0.3960	0	0.000404	***

Remember our equation

$$P(m_{ij} \mid \theta, t, \mu) = \frac{1}{1+e^{-\theta^T \Delta_{ij}}} = \mathrm{logit}^{-1}(\theta^T \Delta_{ij})$$

- Probability of adding an edge is inversely related to $\exp(-\theta^T \Delta_{ij})$

- Positive θ means that

  - when the "change value" of the metric goes up
  - the probability of an edge with that impact also goes up
- Negative θ means the opposite

So

- mix.blue.0.1 and mix.blue.1.0 go
  - from weakly positive (or neutral)
  - to negative
- mix.blue.0.1 increases if the added edge points from Red to Blue
  - mix.blue.1.0 is the other direction
- So, we have shown
  - a significant portion the differences in these graphs
  - can be explained by a decreasing likelihood of cross-group friend ties
  - esp. Red -> Blue ties

Other explanations

- decreasing density, increased mutuality
  - doesn't show up as a trend
- increasing homophily
  - also not significant when density controlled for

Multiple terms change

- Adding an edge will often change multiple terms in t(G)

- Adding a Red->Blue edge

  - increases the number of total edges
    - edges+1

  - might increase the number of mutual edges
    - mutual+1
  - increases the Red->Blue count
    - mix.blue.0.1+1
  - Blue->Red unchanged

Edge probabilities

- What is the probability of adding a Red->Blue edge at time 3
  - assuming it doesn't change the number of mutual edges
    - a corresponding Blue->Red edge doesn't already exist
  - log-odds = (-1.170)change in edges + (0.175)change in red-blue edges
  - log-odds = -1 (approx)
  - Conditional P(R->B) = $e^{-1}/(1+e^{-1})$ = 0.27
- At time 5
  - log-odds = -0.97+ -3.24 = -4.21
  - Conditional P(R->B) = 0.014
- *Factor of 20x*

Note

- Not overall probability
  - % of edges of that type that exist

- The model is generative

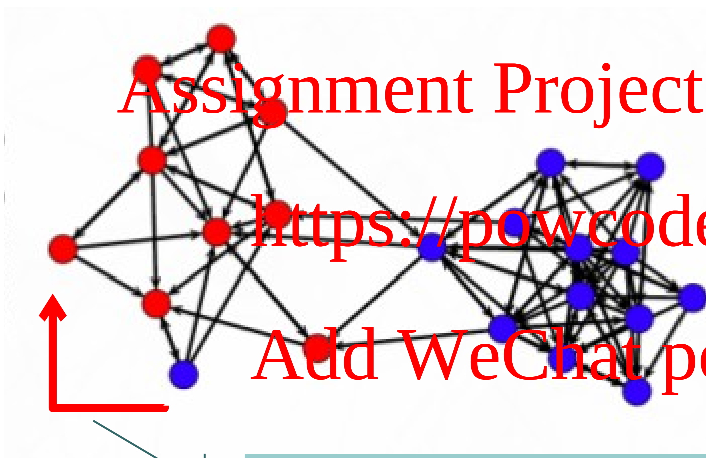
  - like ER random graph

- Conditional probability computed here
  - if an edge is being considered
  - and it changes the test statistics in this particular way
  - this is probability of adding that edge
    - conditioned on the state of the graph

What would we do

- if we wanted to look at reciprocation?

Conceptual model



Add this edge or not?

1. Compute vector of model statistics θ
2. Compute edge probability p
3. Add edge with probability p

That's what ERGM is for

- Showing the connection between
  - global properties

    - decreased assortativity

  - specific local phenomena

    - homophily, transitive closure, etc.
- Being able to quantify those connections
  - and state standard errors

Punch line...

- The graphs are 3rd, 4th and 5th graders
  - Red = female
  - Blue = male
- So, we have quantified "cooties"

Big caveat

- ERGM is controversial
- For some graphs
  - adequate mixing requires exponential time
- If your model is wrong
  - confidence values can be misleading
- You have to really understand this tool to use it reliably

ERGM process complex

- Many controls for MCMC
- Many possible ERGM terms
  - plus you can write your own
- Complex interaction
  - between terms chosen and fitting behavior

Next week

- We will talk a lot more about the internals of ERGM
- Interpreting diagnostics
- And then do a lab