

Introduction to Stochastic Actor-Oriented Models

Fundamentals of SAOMs



Johan Koskinen

Department of Statistics
Stockholm University
University of Melbourne

February 21, 2023



Stockholm
University

- All material is on the workshop repository
<https://github.com/johankoskinen/CHDH-SNA>
 - ▶ Download the RMarkdown file CHDH-SNA-2.Rmd
- In order to run the Markdown you need
 - ▶ The R-package 
 - ▶ The RStudio interface  RStudio
- We will predominantly use the packages
 - ▶ sna
 - ▶ network
 - ▶ RSiena

Outline of workshops

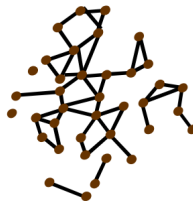
- ① (Basic) Introduction to SAOM (Thursday AM)
 - ▶ SAOM as an agent-based model
 - ▶ How to estimate a SAOM
- ② Analysing data with SAOM
 - ▶ Different model specifications
 - ▶ Different types of data
 - ▶ Trouble shooting and dealing with common issues
- ③ Extensions to SAOM
 - ▶ Even more types of data
 - ▶ Likelihood-based estimation
 - ▶ Settings and imperfect data



The two basic types of data

NETWORK

nodes: Andras, Per, Zsafia
have **ties:** Andras \rightarrow Per



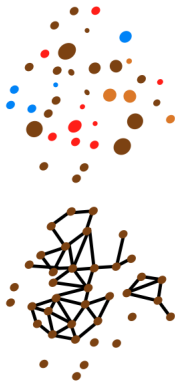
BEHAVIOUR

attributes of nodes: Andras, Per,
Zsafia drink
Zsafia does not smoke



SAOM: longitudinal modelling

We have **observations** on **NETWORK** and **BEHAVIOUR**



At some fixed points
in time

starting at t_0

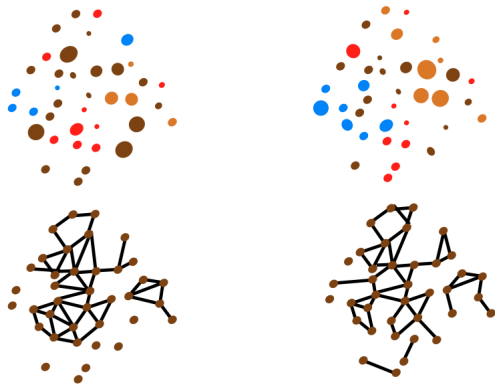
followed by t_1

$t_0 < t_1$

inferential task: **explain** how t_0 change into t_1

SAOM: longitudinal modelling

We have **observations** on **NETWORK** and **BEHAVIOUR**



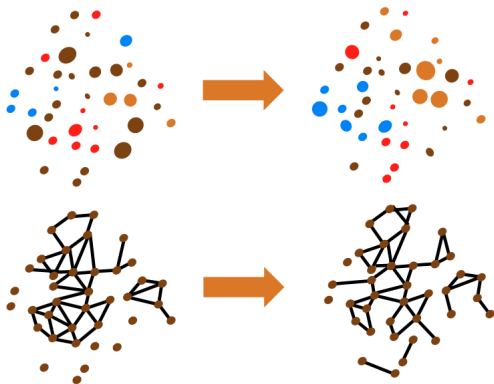
At some fixed points
in time

starting at t_0
followed by t_1
 $t_0 < t_1$

inferential task: **explain** how t_0 change into t_1

SAOM: longitudinal modelling

We have **observations** on **NETWORK** and **BEHAVIOUR**



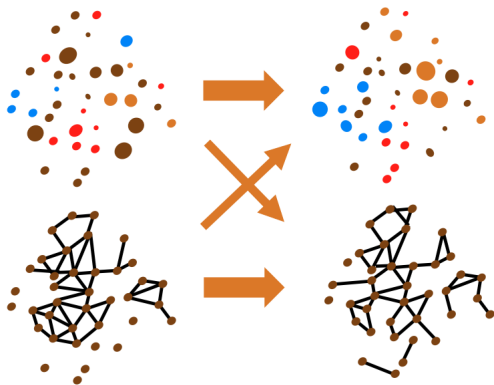
At some fixed points
in time

starting at t_0
followed by t_1
 $t_0 < t_1$

inferential task: **explain** how t_0 change into t_1

SAOM: longitudinal modelling

We have **observations** on **NETWORK** and **BEHAVIOUR**



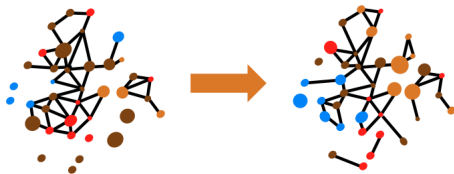
Especially the
co-evolution:

*selection
influence*

inferential task: **explain** how t_0 change into t_1

SAOM: longitudinal modelling

We have **observations** on **NETWORK** and **BEHAVIOUR**



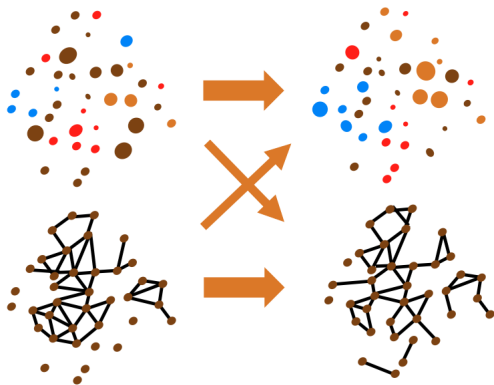
Especially the
co-evolution:

*selection
influence*

inferential task: **explain** how t_0 change into t_1

SAOM: longitudinal modelling

We have **observations** on **NETWORK** and **BEHAVIOUR**



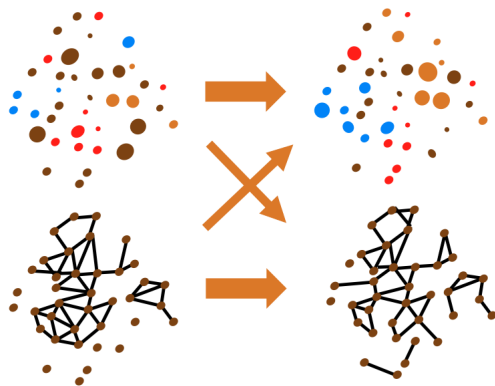
Especially the
co-evolution:

*selection
influence*

inferential task: **explain** how t_0 change into t_1

The SAO Model

We have **observations** on NETWORK and BEHAVIOUR



Especially the
co-evolution:

selection
influence
HOW?

inferential task: **explain** how t_0 change into t_1

The SAO Model

Assume **PARTIAL** observations on a process



observations:

at t_0
and t_1

the rest:
missing

the process **explains** how t_0 change into t_1

The SAO Model

Assume **PARTIAL** observations on a process



observations:

at t_0
and t_1

the rest:
missing

the process **explains** how t_0 change into t_1

What type of data do we want to explain: adjacency matrix

Data represented as adjacency matrices

$$\mathbf{x} = \begin{pmatrix} . & 0 & 0 & 0 & 1 \\ 1 & . & 0 & 0 & 0 \\ 1 & 1 & . & 0 & 0 \\ 0 & 0 & 0 & . & 0 \\ 0 & 0 & 1 & 1 & . \end{pmatrix}$$

where $x_{ij} = 1$ or 0 according to whether $i \rightarrow j$ or not.

What type of data do we want to explain: longitudinal

Data represented as adjacency matrices
where elements **change**

$$x(t_0) = \begin{pmatrix} . & 0 & 0 & 0 & 1 \\ 1 & . & 0 & 0 & 0 \\ 1 & 1 & . & 0 & 0 \\ 0 & 0 & 0 & . & 0 \\ 0 & 0 & 1 & 1 & . \end{pmatrix}$$



What type of data do we want to explain

Data represented as adjacency matrices
where elements **change**

$$x(t_1) = \begin{pmatrix} . & \textcolor{red}{1} & 0 & 0 & 1 \\ 1 & . & 0 & 0 & 0 \\ 1 & \textcolor{red}{0} & . & 0 & 0 \\ 0 & 0 & 0 & . & 0 \\ \textcolor{red}{1} & 0 & 1 & 1 & . \end{pmatrix}$$

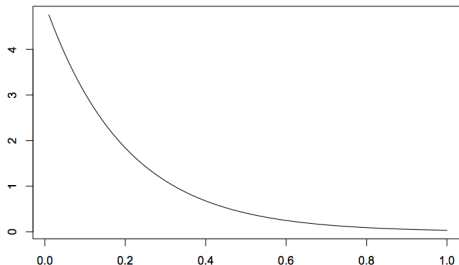
What type of data do we want to explain

Data represented as adjacency matrices
where elements **change**

$$x(t_2) = \begin{pmatrix} . & 1 & 0 & \mathbf{1} & 1 \\ 1 & . & 0 & 0 & \mathbf{1} \\ 1 & \mathbf{1} & . & 0 & 0 \\ 0 & 0 & 0 & . & 0 \\ 1 & 0 & \mathbf{0} & 1 & . \end{pmatrix}$$

SAOM: the rate of change

At random points in time, at rates λ_i



nodes/individuals/**actors** are given **opportunities to change**

SAOM: the *direction* of change

Conditional on an actor having an **opportunity for change**
the probability for each outcome

- ⊙ is modelled like multinomial logistic regression
- ⊙ reflects the attractiveness of the outcome to the actor

Micro-step

When actor i has opportunity to change

They may toggle x_{ij} to $1 - x_{ij}$

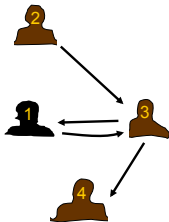
We call the new network

$$X(i \rightsquigarrow j)$$

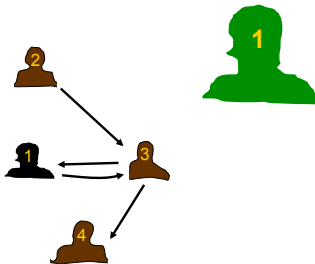
the network X that *differs* in exactly **one** tie-variable x_{ij}



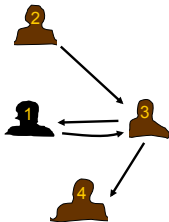
$$x = \begin{array}{|c|c|c|c|} \hline - & 0 & 1 & 0 \\ \hline 0 & - & 1 & 0 \\ \hline 1 & 0 & - & 1 \\ \hline 0 & 0 & 0 & - \\ \hline \end{array}$$



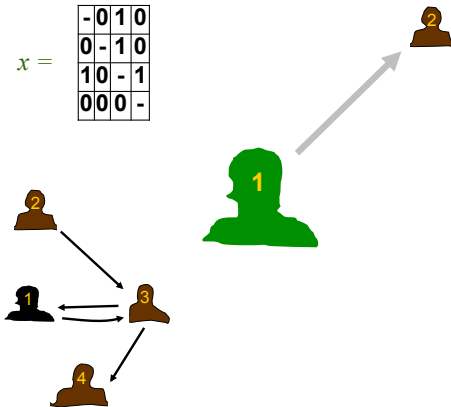
$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



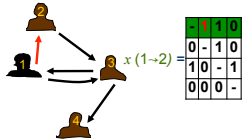
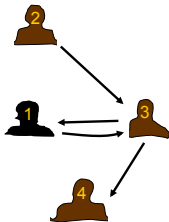
$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$

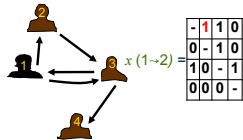
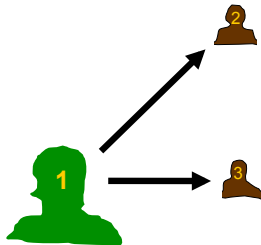
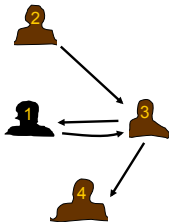


$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



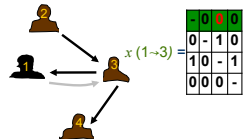
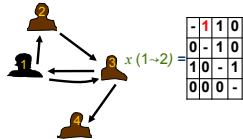
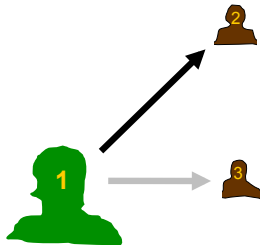
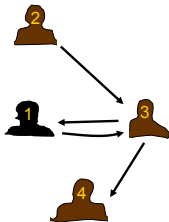
$$x(1 \rightarrow 2) = \begin{bmatrix} - & 1 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$

$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$

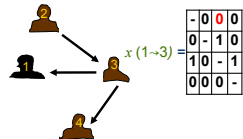
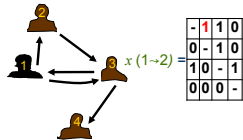
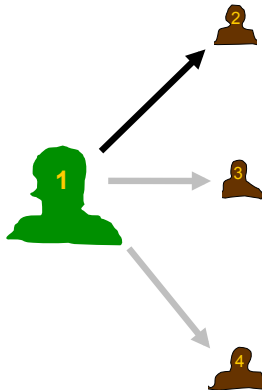
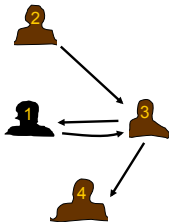


$$x(1 \sim 2) = \begin{bmatrix} - & 1 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$

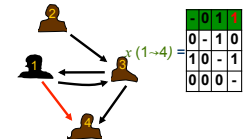
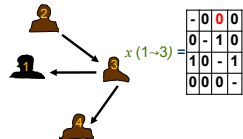
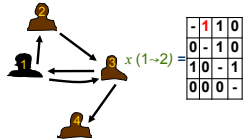
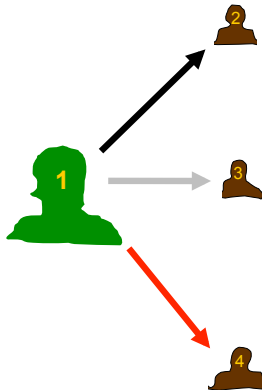
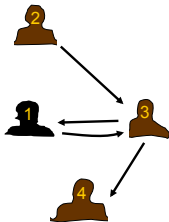
$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



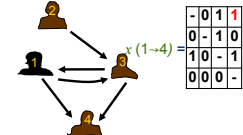
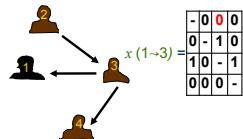
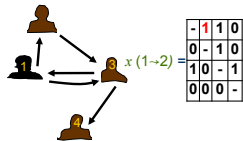
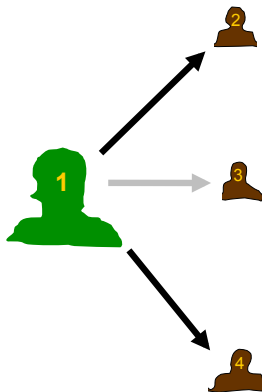
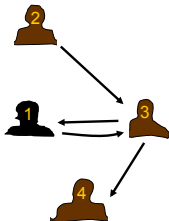
$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



$$x = \begin{bmatrix} - & 0 & 1 & 0 \\ 0 & - & 1 & 0 \\ 1 & 0 & - & 1 \\ 0 & 0 & 0 & - \end{bmatrix}$$



Probability of a change

Of the three changes (for $j = 2, 3, 4$) available to i (here 1) the probability that i toggles the tie $i \rightarrow j$ is given by

One-step jump probability

$$p_{ij}(\beta, \underbrace{X}_{\text{current network}}) = \frac{\exp(f_i(\beta, \underbrace{X(i \rightsquigarrow j)}_{\text{new network}}))}{\underbrace{\sum_{h=1}^n \exp(f_i(\beta, X(i \rightsquigarrow h)))}_{\text{all possible changes}}},$$

where

- $X(i \rightsquigarrow j)$ is the network resulting from the change
- β are **statistical parameters**
- f_i describes the attractiveness of $X(i \rightsquigarrow j)$ to i



Probability of a change: utility

One-step jump probability: *can* be derived as:

- Each network X has **utility** $U_i(X, t)$ for i
- Actor i chooses network X that maximises $U_i(X, t)$

If (random) utility has form

$$U_i(X, t) = \underbrace{f_i(\beta, X)}_{\text{objective function}} + \epsilon_{it}.$$

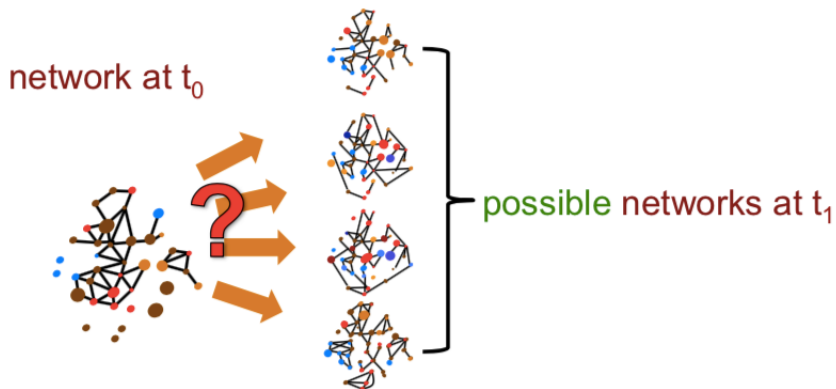
\Uparrow

random component

Actors are (*myopically*) maximising the utility of their network ties



Agent-based: Change driven by incremental updates



Markdown: <https://raw.githubusercontent.com/johankoskinen/CHDH-SNA/main/Markdowns/CHDH-SNA-2.Rmd>

```
library('RSiena')  
library('network')  
library('sna')  
tmp3[is.na(tmp3)] <- 0 # remove missing  
tmp4[is.na(tmp4)] <- 0 # remove missing  
par(mfrow = c(1,2))  
coordin <- plot(as.network(tmp3))  
plot(as.network(tmp4),coord=coordin)
```

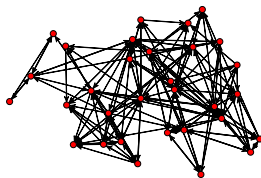
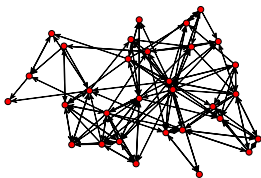


Figure: Network at t_0 and t_1

Let us assume that i **ONLY** cares about not having too many or too few ties:

$$f_i(\beta, X) = \exp \left\{ \beta \sum_j x_{ij} \right\}$$

meaning that

$$p_{ij}(\beta, X) = \frac{\exp \{ \beta (1 - 2x_{ij}) \}}{\sum_{h=1}^n \exp \{ \beta (1 - 2x_{ih}) \}} ,$$

because if

- currently $x_{ij} = 1$, then
- the number of ties for i in $X(i \rightsquigarrow j)$ will be one less (-1) ,
- and if currently $x_{ij} = 0 \dots$

Simulation settings: actors only care about degree

Let the rate be equal for all $\lambda_i = \lambda = 5.7288$

- ✓ in each iteration, actor with shortest waiting time 'wins' (and gets to change)
- ✓ on average every actor gets 5.7 opportunities to change

and set $\beta = -0.7349$

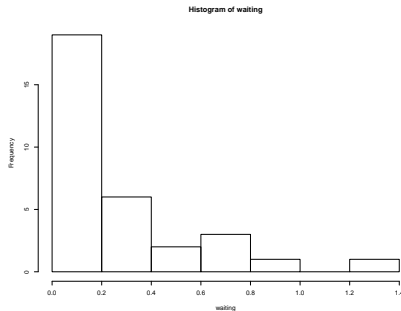
- ✓ if $\beta = 0$ actor would not care if tie was added or deleted
- ✓ here $\beta < 0$ meaning that actor wants less than half of the possible ties



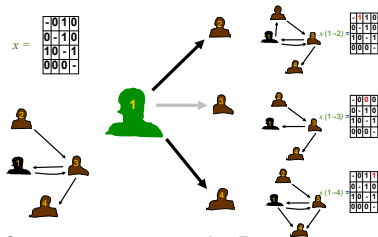
```
mynet1 <- sienaDependent(array(c(tmp3, tmp4),  
                                dim=c(32, 32,2)))  
mydata <- sienaDataCreate(mynet1)  
myeff <- getEffects(mydata)  
myeff <- includeEffects(myeff, recip,include=FALSE)  
myeff$initialValue[  
    myeff$shortName == 'Rate'] <- 5.7288  
myeff$initialValue[  
    myeff$shortName=='density'][1] <- -0.7349
```

Model: rate

```
waiting <- rexp(32,5.7288)
hist(waiting)
which( waiting == min(waiting)) ← the winner
```



Model: conditional one-step change probability



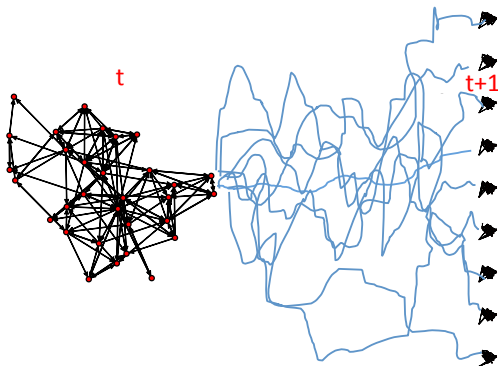
$$\Pr(1 \leadsto 2) = \frac{e^{-0.7349}}{e^{-0.7349} + e^{1.1059} + e^{-0.7349} + 1} = 0.1185$$

$$\Pr(1 \leadsto 3) = \frac{e^{1.1059}}{e^{-0.7349} + e^{1.1059} + e^{-0.7349} + 1} = 0.5156$$

$$\Pr(1 \leadsto 4) = \frac{e^{-0.7349}}{e^{-0.7349} + e^{1.1059} + e^{-0.7349} + 1} = 0.1185$$

Of course, in van de Bunt every actor has 31+1 choices for change

Now let us simulate



Simulate from t_0 to t_1 (`simOnly = TRUE`)

```
sim_model <- sienaAlgorithmCreate(  
  projname = 'sim_model',  
  cond = FALSE,  
  useStdInits = FALSE, nsub = 0 ,  
  simOnly = TRUE)  
sim_ans <- siena07( sim_model, data = mydata,  
  effects = myeff,  
  returnDeps = TRUE, batch=TRUE )
```

The object `sim_ans` will now contain 1000 simulated networks



Extract networks

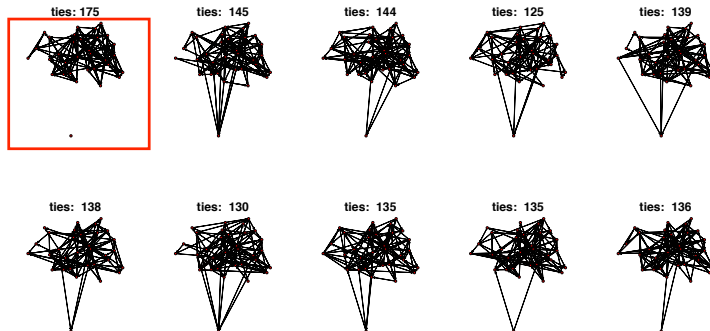
```
n <- dim(tmp4)[1]  
mySimNets <- reshapeRSienaDeps( sim_ans , n )
```

The object `mySimNets` is a 1000 by n by n array of adjacency matrices

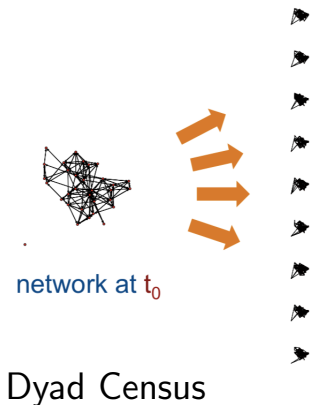
Plot observed network and 9 simulated

```
pdf(file='simnets1.pdf', width = 9,height =4.5)
par(mfrow=c(2,5), oma = c(0,4,0,0) + 0.1,
    mar = c(5,0,1,1) + 0.1)
plot(as.network(tmp4),coord=coordin,
     main=paste('ties:',sum(tmp4) ) )
apply(mySimNets[1:9,,],1,function(x)
     plot(as.network(x),
          coord=coordin,
          main=paste('ties: ',sum(x))) )
dev.off()
```

The **observed** at t_1 and possible networks at t_1



Simulated networks v t_1 obs



```
> dyad.census(tmp4)
      Mut Asym Null
[1,]  46   83  367
> dyad.census(mySimNets[1:9,,])
      Mut Asym Null
[1,]  23  128  345
[2,]  25  133  338
[3,]  17  136  343
[4,]  20  134  342
[5,]  16  143  337
[6,]  21  136  339
[7,]  26  118  352
[8,]  23  128  345
[9,]  30  122  344
```

Conclusions

A process where i ONLY cares about not having too many or too few ties does to replicate the reciprocity at t_1

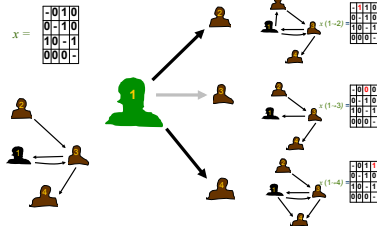
Assume that i ALSO cares about having ties $i \rightarrow j$ reciprocated $j \rightarrow i$

$$f_i(\beta, X) = \exp \left\{ \beta_d \sum_j x_{ij} + \beta_r \sum_j x_{ij} x_{ji} \right\}$$

meaning that probability that i toggles relationship to j

$$p_{ij}(\beta, X) = \frac{\exp \{ \beta_d (1 - 2x_{ij}) + \beta_r (1 - 2x_{ij}) x_{ji} \}}{\sum_{h=1}^n \exp \{ \beta_d (1 - 2x_{ih}) + \beta_r (1 - 2x_{ih}) x_{hi} \}},$$





Objective function:

$$\beta_d \sum_j x_{ij} + \beta_r \sum_j x_{ij} x_{ji}$$

- adding $1 \rightarrow 2$: β_d
- deleting $1 \rightarrow 3$: $\beta_d - \beta_r$
- adding $1 \rightarrow 4$: β_d

Our simulated networks had too **few** reciprocated dyads so we need to set $\beta_r \dots$

Simulation settings: actors care about degree and reciprocity

Let the rate be equal for all $\lambda_i = \lambda = 6.3477$

✓ on average every actor gets 6.3 opportunities to change

and set $\beta_d = -1.1046$

✓ here $\beta_d < 0$ - actors do not want too many ties

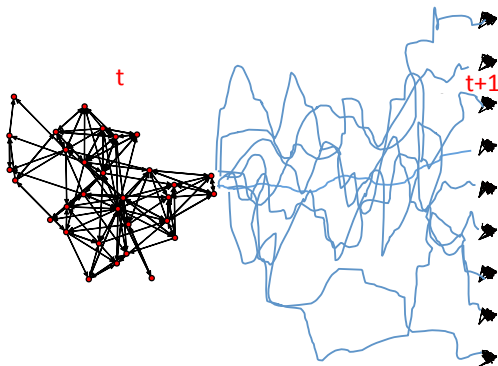
and set $\beta_r = 1.2608$

✓ here $\beta_r > 0$ - actors prefer reciprocated to asymmetric ties




```
myeff <- includeEffects(myeff, recip, include=TRUE)
myeff$initialValue[
    myeff$shortName == 'Rate'] <- 6.3477
myeff$initialValue[
    myeff$shortName == 'density'][1] <- -1.1046
myeff$initialValue[
    myeff$shortName == 'recip'][1] <- 1.2608
```

Now let us simulate



Simulate from t_0 to t_1 now with reciprocity (`simOnly = TRUE`)

```
sim_model <- sienaAlgorithmCreate(
  projname = 'sim_model',
  cond = FALSE,
  useStdInits = FALSE, nsub = 0 ,
  simOnly = TRUE)
sim_ans <- siena07( sim_model, data = mydata,
  effects = myeff,
  returnDeps = TRUE, batch=TRUE )
```

The object `sim_ans` will now contain 1000 simulated networks

NOTE: this piece of code is unchanged

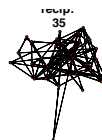
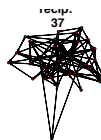
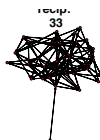
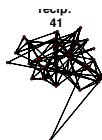
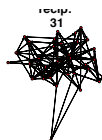
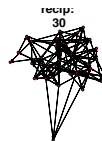
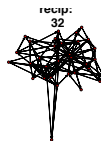
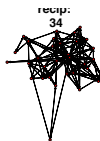
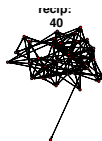


Plot observed network and 9 simulated

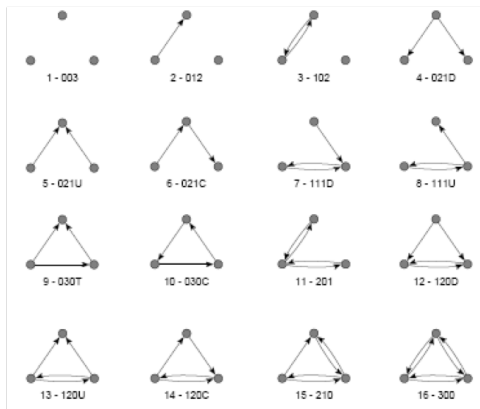
```
mySimNets <- reshapeRSienaDeps(sim_ans,n)
plot(as.network(tmp4),coord=coordin,
      main=paste('recip:',dyad.census(tmp4)[1] ) )
apply(mySimNets[1:9,,],1,function(x)
      plot(as.network(x),
            coord=coordin,
            main=paste('recip:
',dyad.census(x)[1] ) ) )
```



The **observed** at t_1 and possible networks at t_1



Simulated networks v t_1 obs: triad census



Simulated networks v t_1 obs: triad census

```
> triad.census(tmp4)
      003  012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210 300
[1,] 2078 1329 745 146  80  52  65 217  37  0 68  16  65  10 30 22
> triad.census(mySimNets[1:9,,])
      003  012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210 300
[1,] 1968 1381 718  95  84 160 148 224  16  4 77  13  13  17 33  9
[2,] 2067 1348 703  85  82 154 150 191  16  8 75  9  11  25 27  9
[3,] 2073 1397 687 102  75 158 129 181  18  2 68  14  13  23 13  7
[4,] 2185 1313 733  78  60 132 102 172  20  7 89  7  10  20 28  4
[5,] 2040 1340 766  89  64 155 129 189  18  7 82  12  11  19 27 12
[6,] 2206 1403 669  76  68 135 122 143  17  6 64  8  12  9 17  5
[7,] 1788 1357 760 113  89 169 195 238  30 11 98  14  16  34 37 11
[8,] 2164 1301 681  70  65 136 168 174  12  5 91  15  8  30 28 12
[9,] 1988 1383 729 111  67 151 148 173  26 10 82  13  22  19 32  6
```

Reciprocity is clearly not enough to explain the incidence of *transitive triangles* and *simmelian ties* (3 Mutual 0, Assymmetric, 0 Null)

Assume that i ALSO cares about *closure*

$$f_i(\beta, X) = \exp \left\{ \beta_d \sum_j x_{ij} + \beta_r \sum_j x_{ij} x_{ji} + \beta_t s_{i,t}(x) \right\}$$

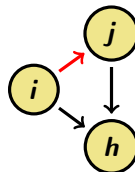
Modelled through, e.g.

transitive triplets effect,

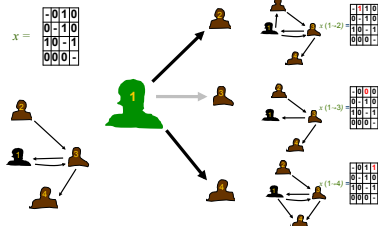
number of transitive patterns in i 's ties

$(i \rightarrow j, j \rightarrow h, i \rightarrow h)$

$$s_{i,t}(x) = \sum_{j,h} x_{ij} x_{jh} x_{ih}$$



transitive triplet



Objective function including $s_{i,t}(x)$:

- adding $1 \rightarrow 2$: $\dots + \beta_t$
- deleting $1 \rightarrow 3$: no change in closure
- adding $1 \rightarrow 4$: $\dots + \beta_t$

Our simulated networks had too **few** 030T and 300 so we need to set $\beta_t \dots$

Assume actors care about degree, reciprocity, and closure

Let the rate be equal for all $\lambda_i = \lambda = 7.0959$

- ✓ on average every actor gets 7 opportunities to change

and set $\beta_d = -1.6468$

- ✓ here $\beta_d < 0$ - actors do not want too many ties

and set $\beta_r = 0.8932$

- ✓ here $\beta_r > 0$ - actors prefer reciprocated to asymmetric ties

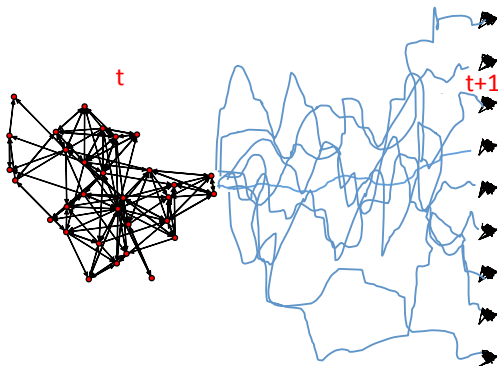
and set $\beta_t = 0.2772$

- ✓ here $\beta_t > 0$ - actors prefer ties that close open triads



```
myeff <- includeEffects(myeff, recip, include=TRUE)
myeff <- includeEffects(myeff, transTrip, include=TRUE)
myeff$initialValue[
    myeff$shortName == 'Rate'] <- 7.0959
myeff$initialValue[
    myeff$shortName == 'density'][1] <- 1.6468
myeff$initialValue[
    myeff$shortName == 'recip'][1] <- 0.8932
myeff$initialValue[
    myeff$shortName == 'transTrip'][1] <- 0.2772
```

Now let us simulate



Simulate from t_0 to t_1 now with transitivity (`simOnly = TRUE`)

```
sim_model <- sienaAlgorithmCreate(
  projname = 'sim_model',
  cond = FALSE,
  useStdInits = FALSE, nsub = 0 ,
  simOnly = TRUE)
sim_ans <- siena07( sim_model, data = mydata,
  effects = myeff,
  returnDeps = TRUE, batch=TRUE )
```

NOTE: this piece of code is unchanged

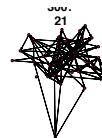
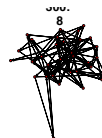
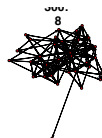
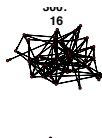
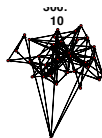
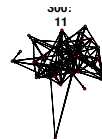
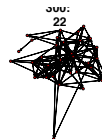
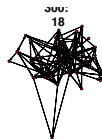
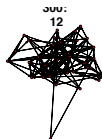
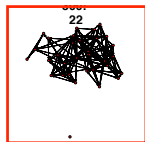


Plot observed network and 9 simulated

```
mySimNets <- reshapeRSienaDeps(sim_ans,n)
plot(as.network(tmp4),coord=coordin,
      main=paste('recip:',triad.census(tmp4)[16] ) )
apply(mySimNets[1:9,,],1,function(x)
      plot(as.network(x),
            coord=coordin,
            main=paste('300:
',triad.census(x)[16] ) ) )
```



The **observed** at t_1 and possible networks at t_1



Simulated networks v t_1 obs: triad census

```
> triad.census(tmp4)
  003  012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210 300
[1,] 2078 1329 745 146  80  52  65 217  37  0 68 16 65 10 30 22
> triad.census(mySimNets[1:9,,])
  003  012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210 300
[1,] 2029 1223 662 108  79 159 138 257  36  6 82 19 35 29 68 30
[2,] 2163 1292 727  89  62 125 136 160 15  4 61  9 23 25 29 40
[3,] 2551 1191 657  44  34  84  88 143  5  0 78  6 16 14 27 22
[4,] 1990 1430 576 100  84 173 128 200 23 10 103 12 38 22 45 26
[5,] 2219 1390 631 103  83 121  84 175 25  5 45  8 21  9 31 10
[6,] 2031 1218 799  75  56 109 136 213 15  9 114 15 25 25 61 59
[7,] 2079 1443 597 104  72 160  97 206 24  4 54 17 24 24 38 17
[8,] 2105 1256 764  77  55 114  99 212 12  5 98 11 33 16 50 53
[9,] 2405 1260 569 103  55 126  78 184 22  6 41 12 25  8 46 20
```

Reciprocity together with transitivity seems enough to explain the incidence of *transitive triangles* and *simmelian ties* (3 Mutual 0, Assymmetric, 0 Null)

What effects are there?

- RSiena Manual
http://www.stats.ox.ac.uk/~snijders/siena/RSiena_Manual.pdf - check for shortName
- scroll through the effects available to you for your data `myeff` - check for shortName
- also `effectsDocumentation(myeff)`

Where did I get these numbers?



Estimation by Method of Moments: data

Basics for data

- You need at least 2 observations on $X(t)$ for waves t_0, t_1
- First observations is fixed and contains no information about θ
- No assumption of a stationary network distribution



How to estimate $\theta = (\lambda, \beta)$?

- pick starting values for θ
- simulate from $X(t_0)$ until t_1 - call the simulated network (-s) X_{rep}
- if statistic $Z_k(X_{\text{rep}})$ for parameter k is different to $Z_k(X_{\text{obs}})$, adjust accordingly

Estimation by Method of Moments: aim

For suitable statistic $Z = (Z_1, \dots, Z_K)$,
i.e., K variables which can be calculated from the network;
the statistic Z_k must be *sensitive* to the parameter θ_k
e.g. number of mutual dyads is sensitive to the reciprocity parameter (as
we have seen)

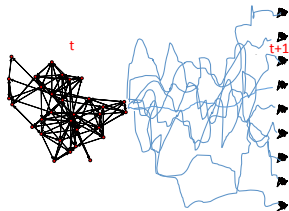
The MoM estimate is a value: $\hat{\theta}$ of θ such that for

- observed stats $Z(X_{\text{obs}})$
- and the expected value $E_{\theta}(Z(X_{\text{rep}}))$

$$E_{\hat{\theta}} \{Z(X_{\text{rep}})\} = Z(X_{\text{obs}}) .$$



Method of Moments matches the moments



Do we have to do this for every update of the parameter θ ?

Robbins-Monro algorithm

The moment equation $E_{\hat{\theta}}\{Z\} = z$ cannot be solved by analytical or the usual numerical procedures

Stochastic approximation (Robbins-Monro, 1951)

Iteration step:

$$\hat{\theta}_{N+1} = \hat{\theta}_N - a_N D^{-1}(z_N - z) , \quad (1)$$

where z_N is a simulation of Z with parameter $\hat{\theta}_N$,

D is a suitable matrix, and $a_N \rightarrow 0$.



Computer algorithm has 3 phases:

- 1 brief phase for preliminary estimation of $\partial E_{\theta} \{Z\} / \partial \theta$ for defining D ;
- 2 estimation phase with Robbins-Monro updates, where a_N remains constant in *subphases* and decreases between subphases;
- 3 final phase where θ remains constant at estimated value; this phase is for checking that $E_{\hat{\theta}} \{Z\} \approx z$, and for estimating D_{θ} and Σ_{θ} to calculate standard errors.



We say that $E_{\hat{\theta}}\{Z\} = z$ is approximately satisfied if, for each statistic $Z_k(X_{\text{obs}})$ is within 0.1 standard deviation of $E_{\theta}(Z(X_{\text{rep}}))$.
This is provided in the output as the *convergence t-ratio* (and the overall maximum convergence ratio is less than 0.25)

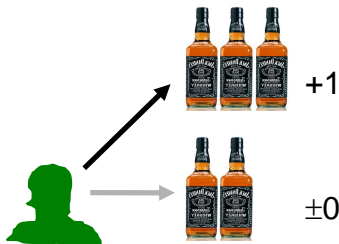


Change to BEHAVIOUR



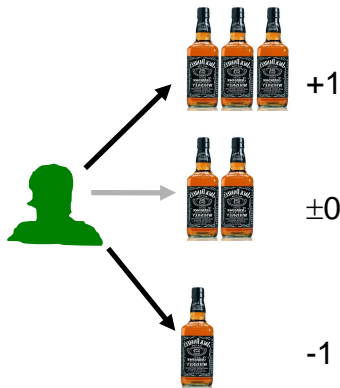
Satisfaction with new state: $f_i +$ random component

Change to BEHAVIOUR



Satisfaction with new state: $f_i +$ random component

Change to BEHAVIOUR



Satisfaction with new state: $f_i + \text{random component}$

For the behaviours, the formula of the change probabilities is

$$p_{ihv}(\beta, z) = \frac{\exp(f(i, h, v))}{\sum_{k, u} \exp(f(i, k, u))} \text{ where } f(i, h, v) \text{ is the objective function}$$

calculated for the potential new situation after a behaviour change,
 $f(i, h, v) = f_i^Z(\beta, z(i, h \rightsquigarrow v))$.

Again, multinomial logit form.



Things that go into the objective functions - selection

Homophily effects:

counts of the number of ties to people that are “like me”



Things that go into the objective functions - influence

Controls:

- 1 Gender
- 2 Age
- 3 Education

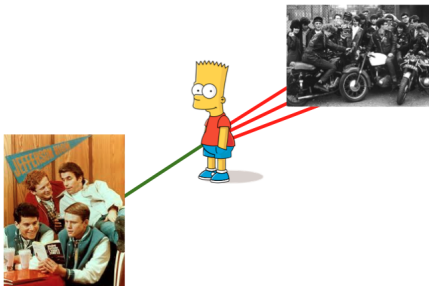
For influence
effects:
imitation
persuasion
etc



Things that go into the objective functions - influence

Controls:

- ① Gender
- ② Age
- ③ Education

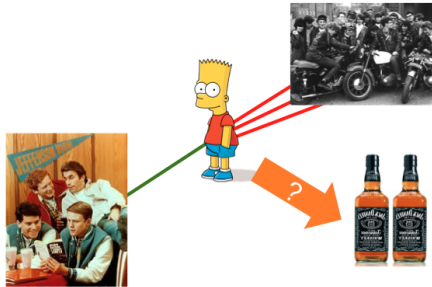


For **influence**
effects:
imitation
persuasion
etc

Things that go into the objective functions - influence

Controls:

- 1 Gender
- 2 Age
- 3 Education



For **influence** effects:
imitation
persuasion
etc

What is the purpose of having the embedded Markov Chain in continuous time?

DYNAMICS

can model change of *tie* as dependent on current ties AND behaviour
can model change in *behaviour* as dependent on current behaviour AND
the behavior of those you are tied to



What is the purpose of having the embedded Markov Chain in continuous time?

STATISTICAL

This is a statistical model that has **estimable parameters** for selection and influence

This is a **generative model** from which we can also generate replicate data AND assess GOF

What is the purpose of having the embedded Markov Chain in continuous time?

STATISTICAL

This is a statistical model that has **estimable parameters** for selection and influence

This is a **generative model** from which we can also generate replicate data AND assess GOF

Compare

- Generalized Estimation Equations
- Regressing behaviour wave 1 on wave 0

Example: 50 girls in a Scottish secondary school

Study of smoking initiation and friendship (starting age 12-13 years)
(following up on earlier work by P. West, M. Pearson & others).
with sociometric & behavior questionnaires at three moments, at appr. 1
year intervals.

Smoking: values 1–3;

drinking: values 1–5;

covariates:

gender, smoking of parents and siblings (binary),

money available (range 0–40 pounds/week).



Rename data that was automatically loaded

```
friend.data.w1 <- s501
friend.data.w2 <- s502
friend.data.w3 <- s503
drink <- s50a
smoke <- s50s
friendshipData <- array( c( friend.data.w1,
                             friend.data.w2,
                             friend.data.w3 ),
                         dim = c( 50, 50, 3 ) )
```



Define dependent/independent data

```
friendship <- sienaDependent(friendshipData)
drinking <- sienaDependent( drink, type = "behavior" )
smoke1 <- coCovar( smoke[ , 1 ] )
```

Join data and get effects

```
NBdata <- sienaDataCreate( friendship,  
                           smoke1,  
                           drinking )  
NBeff <- getEffects( NBdata )
```

Define structural network effects

```
NBeff <- includeEffects( NBeff, transTrip, transRecTrip )
```


Define covariate effects on the network (selection)

```
NBeff <- includeEffects( NBeff,  
                        egoX, egoSqX, altX, altSqX,  
diffSqX,  
                        interaction1 = "drinking" )  
NBeff <- includeEffects( NBeff, egoX, altX, simX,  
                        interaction1 = "smoke1" )
```

Define effects on drinking (influence)

```
NBeff <- includeEffects( NBeff, avAlt, name="drinking",  
                        interaction1 = "friendship" )
```

Define estimation settings and estimate

```
myalgorithm1 <- sienaAlgorithmCreate( projname = 's50_NB' )  
NBans <- siena07( myalgorithm1,  
                  data = NBdata, effects = NBeff,  
                  returnDeps = TRUE )
```



Result selection

Effect	par.	(s.e.)	t stat.
constant friendship rate (period 1)	6.21	(1.08)	-0.0037
constant friendship rate (period 2)	5.01	(0.87)	0.0042
outdegree (density)	-2.82	(0.27)	-0.0809
reciprocity	2.82	(0.35)	0.0559
transitive triplets	0.90	(0.16)	0.0741
transitive recipr. triplets	-0.52	(0.24)	0.0695
smoke1 alter	0.07	(0.17)	0.0343
smoke1 ego	-0.00	(0.15)	0.0747
smoke1 similarity	0.25	(0.24)	0.0158
drinking alter	-0.06	(0.15)	0.0158
drinking squared alter	-0.11	(0.14)	0.0704
drinking ego	0.04	(0.13)	0.0496
drinking squared ego	0.22	(0.12)	0.0874
drinking diff. squared	-0.10	(0.05)	0.0583

convergence t ratios all < 0.09 .

Overall maximum convergence ratio 0.19.

Result Influence

Effect	par.	(s.e.)	<i>t</i> stat.
rate drinking (period 1)	1.31	(0.34)	−0.0692
rate drinking (period 2)	1.82	(0.54)	0.0337
drinking linear shape	0.42	(0.24)	0.0301
drinking quadratic shape	−0.56	(0.33)	0.0368
drinking average alter	1.24	(0.81)	0.0181

convergence *t* ratios all < 0.09 .

Overall maximum convergence ratio 0.19.

Everything you need to know (including scripts for all kinds of data) is available at <http://www.stats.ox.ac.uk/~snijders/siena/>

