

# Computational Sociology

## Agent-based modeling

Dr. Thomas Davidson

Rutgers University

April 26, 2021

# Plan

1. Course updates
2. What is agent-based modeling?
3. NetLogo and NetLogo Web
4. How to construct a simple ABM using R
5. The future of agent-based modeling

# Course updates

## Presentations

- ▶ In-class presentations next Monday
- ▶ Link to Google Slides posted in Slack
  - ▶ Add your slides in alphabetical order
  - ▶ See example at the beginning
    - ▶ Feel free to customize your slides as much as you want

# Course updates

## Deadlines (according to the syllabus)

- ▶ Homework 4 due today at 4pm
- ▶ Preliminary analyses due next Monday, 5/3
- ▶ Final paper due 5/10

# Course updates

## *Actual deadlines*

- ▶ Ensure to have your presentation ready before class on 5/3
- ▶ Submit all outstanding homework and the final paper by 5pm on 5/10

# What is agent-based modeling?

## Agent-based modeling and quantitative social science

- ▶ Most quantitative social science is variable-centered
  - ▶ e.g. We study the associations and interactions between variables in a linear regression

# What is agent-based modeling?

## Agent-based modeling and quantitative social science

- ▶ As a consequence, many sociologists think about the world in terms of what Andrew Abbott calls “general linear reality”
  - ▶ A social world composed of fixed entities with fixed attributes

# What is agent-based modeling?

## Agent-based modeling and quantitative social science

- ▶ Agent-based modeling is the study of “social life as interactions among adaptive agents who influence one another in response to the influence they receive.” (Macy and Willer 2002)
  - ▶ Rather than interactions between variables, we consider interactions between interdependent individuals



# What is agent-based modeling?

## Agent-based modeling and quantitative social science

- ▶ Often we are interested in the *emergent* properties of local interactions between agents and how they aggregate into system-level processes such as diffusion, polarization, and segregation
  - ▶ These complex system-level patterns can emerge without any centralized coordination
- ▶ Like historical sociology and ethnography, agent-based modeling is a *relational* approach, focusing on the context-dependent and contingent nature of social interaction

# What is agent-based modeling?

## Key assumptions

- ▶ Macy and Willer 2002 outline four key assumptions that underpin many sociological agent-based models
  - ▶ Agents are *autonomous*
    - ▶ There is no system-wide coordination
  - ▶ Agents are *interdependent*
    - ▶ Agents respond to each other and to their environment
  - ▶ Agents follow *simple rules*
    - ▶ Simple local rules can generate global complexity
  - ▶ Agents are *adaptive* and *backwards looking*
    - ▶ Agents can alter their behavior through processes such as imitation and learning

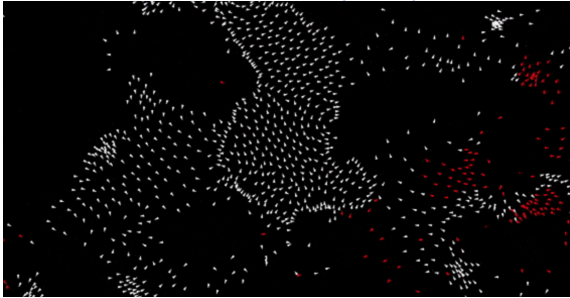
# What is agent-based modeling?

## Advantages of ABMs

- ▶ Virtual experiments to test causal mechanisms
  - ▶ Particularly useful where real-world experimentation is impractical
- ▶ Theory building and testing
  - ▶ Bridging between micro and macro levels of analysis
  - ▶ Varying the social structure *and* the agency of individuals

# What is agent-based modeling?

## Craig Reynolds *Flocking behavior* (1987)



Reynolds, Craig W. 1987. "Flocks, Herds and Schools: A Distributed Behavioral Model." In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, 25–34.

# What is agent-based modeling?

## Thomas Schelling *Homophily and segregation*

### DYNAMIC MODELS OF SEGREGATION

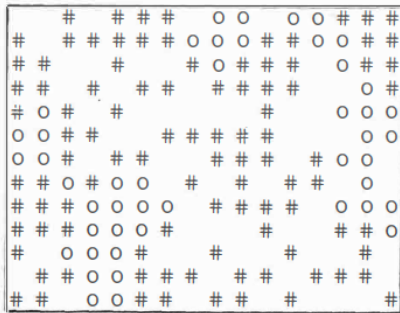
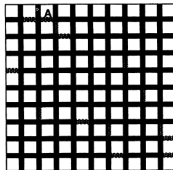


Fig. 13

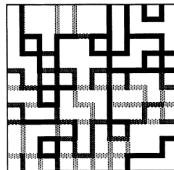
Schelling, Thomas C. 1971. "Dynamic Models of Segregation." *Journal of Mathematical Sociology* 1: 143–86.

# What is agent-based modeling?

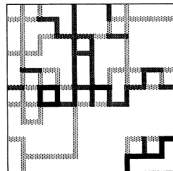
Robert Axelrod *Local convergence and global polarization*  
(1987)



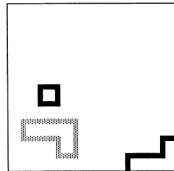
(a) At start



(b) After 20,000 events



(c) After 40,000 events



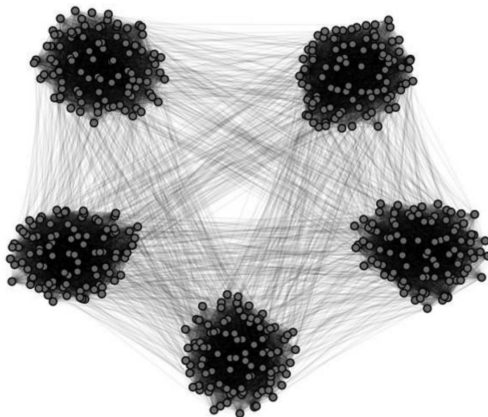
(d) After 80,000 events

Axelrod, Robert. 1997. "The Dissemination of Culture: A Model with Local Convergence and Global Polarization." *Journal of Conflict Resolution* 41 (2): 203–26.

# What is agent-based modeling?

## Testing mechanisms

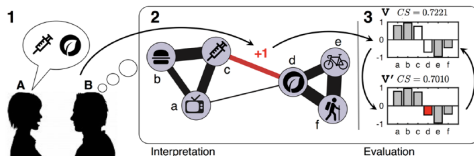
DellaPosta, Shi, and Macy (2015) suggest a mechanism to explain observed correlations between political attitudes and lifestyle choices



# What is agent-based modeling?

## Evaluating competing explanations

Goldberg and Stein (2018) propose an alternative mechanism, arguing that culture does not spread like a virus, but depends on belief structures



**Figure 3.** An Illustration of the Agent-Based Model Sequence

*Note:* (1) Agent *B* observes *A* express support for vaccinations and organic food (practices *c* and *d*); (2) *B* updates the corresponding element in his associative matrix, *R* (the edge connecting nodes *c* and *d* in the network representation of *R*); and (3) randomly updates his preference for organic food (practice *d*, resulting in preference vector *V'*), which is the weaker preference of the pair {*c*,*d*} in his preference vector *V*. Because constraint satisfaction is reduced from .7221 to .7010, this preference update is rejected, and *B*'s preference vector *V* remains unchanged.



# What is agent-based modeling?

## Integrating real-world data

DiMaggio and Garip (2011) construct agent with attributes based on the General Social Survey

Network Externalities, Intergroup Inequality

TABLE 2  
LINEAR REGRESSION OF ADOPTION LEVELS ON EXPERIMENTAL CONDITIONS

	RACE			INCOME		EDUCATION	
	ALL	Whites	Blacks	High	Low	BA	Less than High School
No network externalities .....	-.516**	-.536**	-.399**	-.685**	-.238**	-.611**	-.351**
General network externalities .....	.030**	.028**	.043**	.032**	.017**	.023**	.030**
Homophily = .25 .....	-.003**	-.001	-.012**	.009**	-.014**	.005**	-.011**
Homophily = .5 .....	-.005**	-.002**	-.024**	.017**	-.028**	.010**	-.024**
Homophily = .75 .....	-.011**	-.006**	-.040**	.024**	-.046**	.012**	-.043**
Homophily = 1 .....	-.019**	-.012**	-.061**	.029**	-.067**	.015**	-.068**
Intercept .....	.618**	.647**	.454**	.925**	.249**	.788**	.392**
R <sup>2</sup> .....	.99	.99	.97	.99	.96	.99	.96

NOTE.—All independent variables are binary. Both dependent and independent variables are measured on the final period of simulations ( $t = 100$ ). Reference: homophily = 0;  $N = 7,000$ .

\*  $P < .05$ .

\*\*  $P < .01$ .

# What is agent-based modeling?

## Realism

- ▶ Bruch and Atwell (2015) distinguish between two types of realism in ABMs
  - ▶ *Low-dimensional realism*: simple, parsimonious models
  - ▶ *High-dimensional realism*: complex, complicated models
- ▶ Trade-offs:
  - ▶ The latter might be more realistic, but involve more parameters and may be less intelligible

# What is agent-based modeling?

## Parameters and sensitivity

- ▶ It can be difficult to decide which parameters and which to fix
- ▶ How do system-wide outcomes vary as we adjust parameters?
- ▶ Models can be extremely sensitive to small variations in parameters
  - ▶ Be careful to check for coding errors!
- ▶ Timing matters
  - ▶ Constant time vs. discrete-time
  - ▶ Asynchronous vs. synchronous updating

# NetLogo and NetLogoWeb

## Running agent-based models

- ▶ NetLogo is a widely used environment for constructing agent-based models, storing, and visualizing results
- ▶ NetLogoWeb is a browser version with many examples (<https://www.netlogoweb.org/launch>)
- ▶ There are various interfaces with R to run NetLogo, but I have not used them (e.g. <https://cran.r-project.org/web/packages/RNetLogo/RNetLogo.pdf>)

# NetLogo and NetLogoWeb

## Flocking behavior in NetLogo

<http://www.netlogoweb.org/launch#http://ccl.northwestern.edu/netlogo/models/models/Sample%20Models/Biology/Flocking.nlogo>

# NetLogo and NetLogoWeb

## Schelling's segregation model in NetLogo

<http://www.netlogoweb.org/launch#http://ccl.northwestern.edu/netlogo/models/models/IABM%20Textbook/chapter%203/Segregation%20Extensions/Segregation%20Simple.nlogo>

# NetLogo and NetLogoWeb

## A simple voting model

<http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/Sample%20Models/Social%20Science/Voting.nlogo>

# Building an agent-based model

## A simple contagion model in R

- ▶ Let's simulate a contagion among a population of agents
- ▶ Assumptions
  - ▶ Agents interact at random
  - ▶ Transmission probability is constant for all agents
  - ▶ No agent is immune



# Building an agent-based model

## Generating agents

I use the `setClass` option to define a new class called `agent` with two different numeric properties, `id` and `infected`. I then use `new` to create two different instances of the class.

```
setClass("agent", slots=list(  
  id="numeric",  
  infected="numeric"  
))  
  
a <- new("agent", id=100, infected=0)  
b <- new("agent", id=101, infected=1)
```

# Building an agent-based model

## Generating agents

The agents are what are known as S4 classes in R. This means that all slots must be of the correct type. e.g. We cannot set `id` to be characters.

```
print(a@id)
```

```
## [1] 100
```

```
print(a@infected)
```

```
## [1] 0
```

```
#a@id <- 'a' # uncomment and run to produce error
```

Read more about S4 classes here: <http://adv-r.had.co.nz/S4.html>

# Building an agent-based model

## Generating agents

We can use a function to generate a set of  $N$  agents and store them in a list.

```
agent.generator <- function(N){  
  agents <- list()  
  for (i in 1:N) {  
    agents[[i]] <- new("agent", id=i,infected=0)  
  }  
  return(agents)  
}
```

# Building an agent-based model

## Interaction protocols

Next we want to define how agents interact. This function takes a focal agent, indexed by  $i$ , and randomly selects another agent  $j$ , where  $i$  is not equal to  $j$ .

```
select.partner <- function(i, N){ # i is the focal agent
  ids <- c(1:N) # define list of IDs
  ids <- ids[-i] # remove ith id
  j <- sample(ids, 1) # pick j at random
  return(j)
}
```

# Building an agent-based model

## Interaction protocols

The next function, `interact`, defines how agents `i` and `j` interact, in this case, whether the virus spreads. Parameter `P` denotes the probability of transmission. Note the function takes and modifies the entire list of agents.

```
interact <- function(agents, i, j, P){  
  if (agents[[i]]@infected == agents[[j]]@infected) {} # no action if s  
  else if (agents[[i]]@infected == 1) {  
    # infect j with P  
    agents[[j]]@infected <- rbinom(n=1, size=1, prob=P)  
  } else {  
    # infect i with P  
    agents[[i]]@infected <- rbinom(n=1, size=1, prob=P)  
  }  
  return(agents)  
}
```

# Building an agent-based model

## Putting together a simulation

```
simulator <- function(N, t, P, agents){  
  results <- numeric(t) # 0 vector of length t  
  agents[[sample(1:N, 1)]]@infected <- 1 # randomly infect 1 agent  
  for (timestep in 1:t) { # for each timestep  
    for (i in sample(1:N)) { # for each agent  
      j <- select.partner(i, N) # select a partner  
      agents <- interact(agents, i, j, P) # interact  
    }  
    statuses <- numeric(N)  
    for (i in 1:N) {statuses[[i]] <- agents[[i]]@infected}  
    results[[timestep]] <- sum(statuses)/N # prop infected at timestep  
  }  
  return(list("results"=results,  
             "agents"=agents))  
}
```

# Building an agent-based model

## Running a single simulation

Here we define the relevant parameters, generate a set of agents, and run simulator.

```
N = 50 # agents
P = .1 # transmission probability
t= 10 # timesteps

set.seed(478437) # set randomization seed

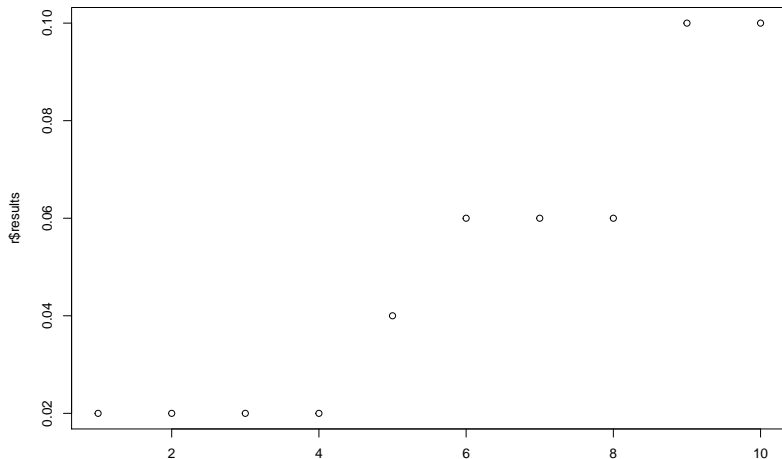
agents <- agent.generator(N) # gen N agents
r <- simulator(N, t, P, agents) # run sim

print(r$results)

## [1] 0.02 0.02 0.02 0.02 0.04 0.06 0.06 0.06 0.10 0.10
```

# Building an agent-based model

The graphic shows the proportion infected at each timestep.





# Building an agent-based model

## Running multiple simulations

```
K = 100 # trials
results.matrix <- matrix(nrow=K*t, ncol=3)

i <- 1 # iterator
for (k in 1:K) {
  agents <- agent.generator(N)
  results <- simulator(N, t, P, agents)
  timestep <- 1
  for (r in results$results) {
    results.matrix[i,] <- c(r,timestep,k)
    timestep <- timestep + 1
    i <- i + 1
  }
}
```

# Building an agent-based model

## Running multiple simulations

```
df <- data.frame(results.matrix)
colnames(df) <- c("prop", "time", "id")
print(head(df))
```

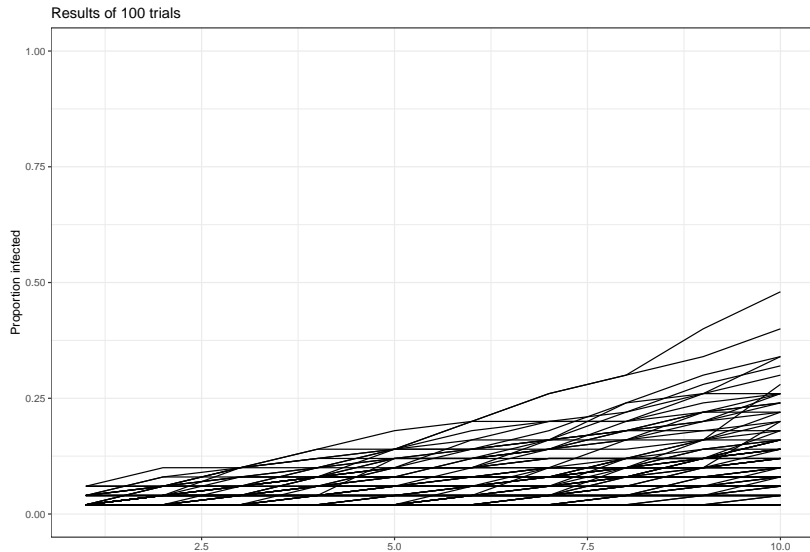
```
##   prop time id
## 1 0.04    1  1
## 2 0.08    2  1
## 3 0.10    3  1
## 4 0.10    4  1
## 5 0.14    5  1
## 6 0.14    6  1
```

# Building an agent-based model

## Running multiple simulations

```
library(ggplot2)
library(viridis)
library(tidyverse)
```

# Building an agent-based model



# Building an agent-based model

## Varying P

Now we want to examine how the results vary across different transmission probabilities.

```
P.vals <- c(0.1,0.2,0.3) # added three different variations of P  
results.matrix <- matrix(nrow=K*t*length(P.vals), ncol=4) # Define a new
```

# Building an agent-based model

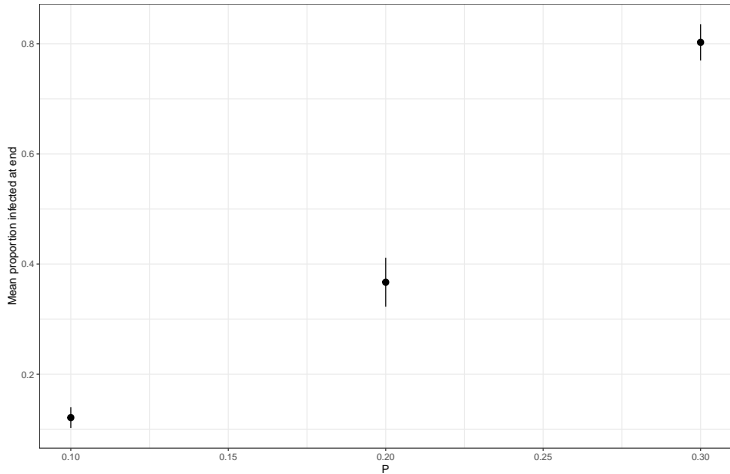
## Varying P

```
i <- 1
for (P in P.vals) {
  for (k in 1:K) {
    agents <- agent.generator(N)
    results <- simulator(N, t, P, agents)
    timestep <- 1
    for (r in results$results) {
      results.matrix[i,] <- c(r,timestep,P,k)
      timestep <- timestep + 1
      i <- i + 1
    }
  }
}
```

# Building an agent-based model

## Varying P

Comparing different transmission probabilities

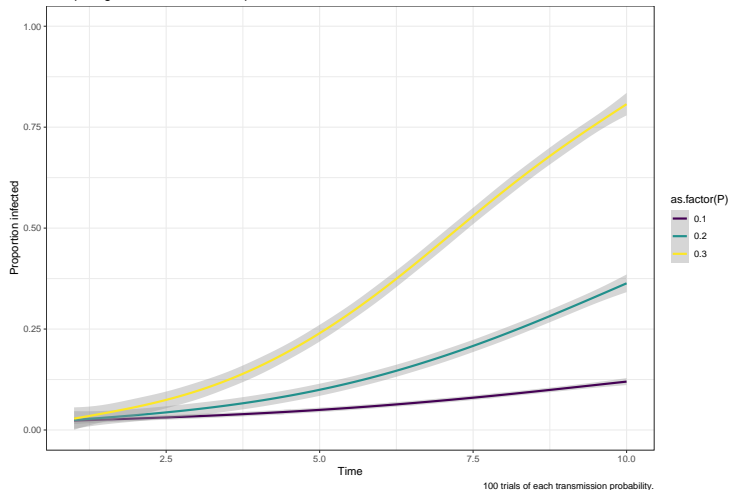


100 trials of each transmission probability.

# Building an agent-based model

## Varying P

Comparing different transmission probabilities





# Building an agent-based model

## Adding a parameter

```
setClass("agent", slots=list(
  id="numeric",
  infected="numeric",
  shape="character" # Adding an extra attribute
))

agent.generator <- function(N){
  agents <- list()
  for (i in 1:N) {
    agents[[i]] <- new("agent", id=i, infected=0, shape=sample(c("square"
  })
  return(agents)
}
```

# Building an agent-based model

## Adding a parameter

```
# Defining a helper function to return a list of ids of agents with a g
ids.by.shape <- function(shape, agents){
  agent.ids <- c()
  for (i in 1:length(agents)) {
    if (agents[[i]]@shape == shape)
    {
      agent.ids <- append(agent.ids, c(agents[[i]]@id))
    }
    else {}
  }
  return(agent.ids)
}
```

# Building an agent-based model

## Updating select.partner to induce homophily

```
select.partner <- function(i, agents, H){  
  i.shape <- agents[[i]]@shape # get i shape  
  agents <- agents[-i] # remove ith id  
  if (i.shape == "circle") {  
    alter.shape <- sample(c("square","circle"), size=1, prob=c(1-H,H))  
  }  
  else {  
    alter.shape <- sample(c("square","circle"), size=1, prob=c(H,1-H))  
  }  
  ids <- ids.by.shape(alter.shape, agents)  
  j <- sample(ids, 1) # pick j at random  
  return(j)  
}
```

# Building an agent-based model

## Updating the simulator function

```
simulator.2 <- function(N, t, P, agents, H){  
  results <- numeric(t) # 0 vector of length t  
  agents[[sample(1:N, 1)]]@infected <- 1 # randomly infect 1 agent  
  for (timestep in 1:t) { # for each timestep  
    for (i in sample(1:N)) { # for each agent  
      j <- select.partner(i, agents, H) # selected a partner  
      agents <- interact(agents, i, j, P) # interact  
    }  
    statuses <- numeric(N) # get prop infected at t  
    for (i in 1:N) {statuses[[i]] <- agents[[i]]@infected}  
    results[[timestep]] <- sum(statuses)/N  
  }  
  return(list("results"=results,  
             "agents"=agents))  
}
```

# Building an agent-based model

## Defining new parameters

```
H.vals <- c(0.5, 0.6, 0.7, 0.8, 0.9, 1.0)
```

```
P.vals <- c(0.1,0.2,0.3,0.4,0.5,0.6)
```

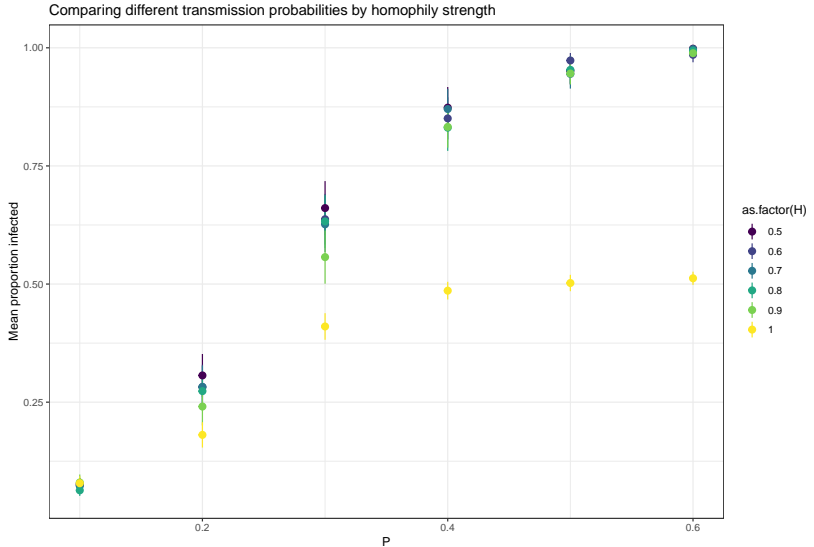
```
results.matrix <- matrix(nrow=K*t*length(H.vals)*length(P.vals), ncol=5
```

# Building an agent-based model

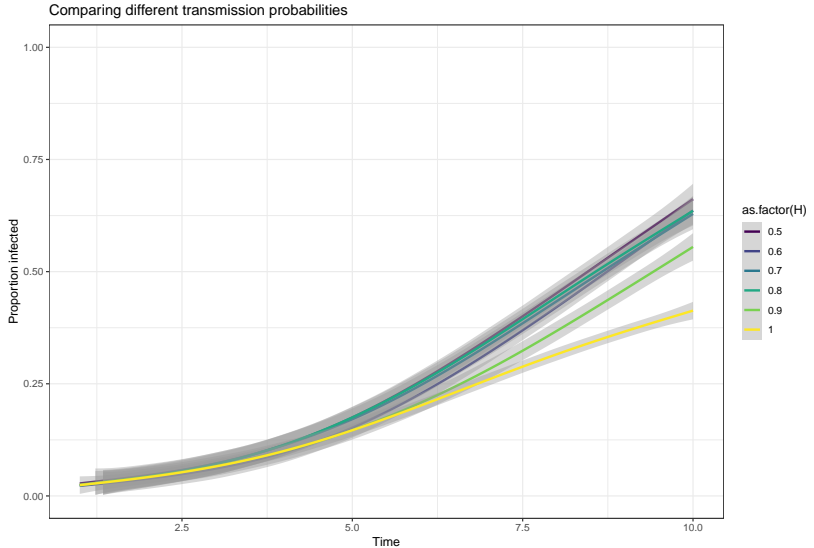
## Running the new simulations

```
i <- 1
for (H in H.vals) {
  for (P in P.vals) {
    for (k in 1:K) {
      agents <- agent.generator(N)
      results <- simulator.2(N, t, P, agents, H)
      timestep <- 1
      for (r in results$results) {
        results.matrix[i,] <- c(r,timestep,P,H,k)
        timestep <- timestep + 1
        i <- i + 1
      }
    }
  }
}
```

# Building an agent-based model



# Building an agent-based model





# Building an agent-based model

## Back to our assumptions

- So far this model is very simple. What are some of the assumptions it makes?

# Building an agent-based model

## Back to our assumptions

- ▶ So far this model is very simple. What are some of the assumptions I make?
  - ▶ Only groups, square and circles
    - ▶ Each group has the same tendency towards homophily
    - ▶ Each group is the same size
  - ▶ Homophily and transmission probability are constant
  - ▶ Within-group, interactions are random
    - ▶ All relationships are possible, there are no structural holes
  - ▶ No agent is immune / non-compliant

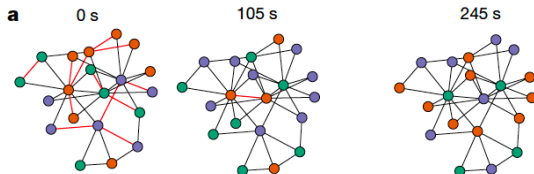
# Building an agent-based model

## Back to our assumptions

- ▶ The main challenge when constructing an ABM is to determine which parameters are theoretically relevant and how to operationalize them
- ▶ For example, if we considered this as a model of cultural transmission it is important to recognize that culture does not spread like a virus (Goldberg and Stein 2018)
  - ▶ But how does culture diffuse? The onus is on the modeler to develop a parsimonious mechanism and implement it in code
- ▶ This is difficult, but it forces us to think carefully about our theories and our assumptions

# The future of agent-based modeling

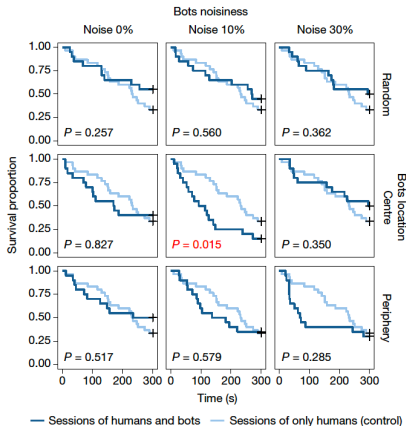
## Human-agent interactions



Shirado, Hirokazu, and Nicholas A. Christakis. 2017. "Locally Noisy Autonomous Agents Improve Global Human Coordination in Network Experiments." *Nature* 545 (7654): 370–74.

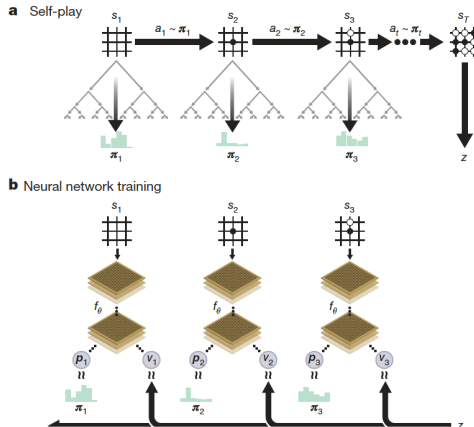
# The future of agent-based modeling

## Human-agent interactions



# The future of agent-based modeling

## Reinforcement learning and autonomous agents



Silver, David et al. 2017. "Mastering the Game of Go without Human Knowledge." *Nature* 550 (7676): 354–59.

# Summary

- ▶ Agent-based modeling allows us to simulate complex social systems
  - ▶ Interdependent, emergent, relational
- ▶ The technique has been used by sociologists to study a range of different processes and to develop and test theories
  - ▶ But there are difficult trade-offs between parsimony (low-dimensional realism) and complexity (high-dimensional realism)
- ▶ NetLogo provides a suite of tools for agent-based modeling
- ▶ R's object-oriented functionality can be used to create models