Agent-based modeling for infectious diseases

Kyung Hee University

Sunmi Lee

2021. January 18-19

Contents

5. The Components of Agent-Based Modeling

Agents

- Agents are the basic units of agent-based modeling. As such, it is important to choose the design of your agents carefully. The two main aspects that define agents are the *properties* that they have and the *actions* (sometimes called behaviors or methods) that they can execute.
- *Agent properties* are the internal and external state of the agents—their data and description. *Agent behaviors or actions* are what the agents can do.
- Besides these two main agent attributes, there are also several issues that are related to agent design. First is the issue of agent "grain-size": which is most effective for the chosen model? For example, if you are modeling a political system, do want to model the individual actors or, instead, the political institutions or even each nation's government as a single entity?
- A second factor to consider is agent cognition. How much capability do the agents have to observe the world around them and make a decision? Do they act in a stimulus-response fashion? Or do they plan out their actions? Finally, we discuss some special types of agents: proto-agents, which are not fully specified agents; and meta-agents, composed of other agents.

Properties

- Agent properties describe an agent's current state, the items that you see when you inspect an agent. In chapter 2, we briefly described how to use a patch monitor to see the current state of a patch, but you can also use monitors to inspect turtles and other types of agents as well. In this example, we are going to explore agent properties using a turtle monitor, but in NetLogo, you can monitor links and patches as well.

Behaviors (Actions)

- Besides items that define the state of an agent (properties), it is also necessary to define how the agent can behave (the actions it can take). An agent's *actions* or *behaviors* are the ways in which the agent can change the state of the environment, other agents, or itself. In NetLogo, there are many behaviors that are predefined for the agents. The list of all these predefined behaviors is too large to iterate through here, but it includes actions like FORWARD, RIGHT, LEFT, HATCH, DIE, and MOVE-TO.

Collections of Agents

- *Types of Agents* Agents are typically divided into three main types: mobile agents that can move about the landscape; stationary agents that cannot move at all; and connecting agents that link two or more other agents.
- In NetLogo, turtles are mobile agents, patches are stationary agents, and links are connecting agents. From a geometric standpoint, turtles are generally treated as shapeless area-less points, although they may be visualized with various shapes and sizes). Thus, even if a turtle appears to be large enough to be on several patches at the same time, it is only contained by the patch at the turtle's center (given by XCOR and YCOR).
- Patches are sometimes used to represent a passive environment and are acted upon by the mobile agents; other times, they can take actions and perform operations. A primary difference between patches and turtles is that patches cannot move. Patches also take up a defined space/area in the world; thus, a single patch can contain multiple turtle agents on it. Links are also unable to move themselves.
- They connect two turtles that are the "end nodes" of the link, but the visual representations of the links do move when their end nodes move. Links are often used to represent the relationship between turtles. They can also represent the environment—e.g., by defining transportation routes that agents can move along, or by representing friendship/communication channels. Despite these differences, all three share the ability to behave, that is, to take actions and perform operations on themselves.

Environments

- Another early and critical decision is how to design the *environment* of the agent-based model. The environment consists of the conditions and habitats surrounding the agents as they act and interact within the model. The environment can affect agent decisions, and, in turn, can be affected by agent decisions.
- Spatial environments in agent-based models generally have two variants: discrete spaces and continuous spaces. In a mathematical representation, in continuous spaces between any pair of points, there exists another point, while in discrete spaces, though each point has a neighboring point, there do exist pairs of points without other points between them, so that each point is separated from every other point. However, when implemented in an ABM, all continuous spaces must be implemented as approximations, so that continuous spaces are represented as discrete spaces where the spaces between the points are very small.

Discrete Spaces

- The most common discrete spaces used in ABM are lattice graphs (also sometimes referred to as mesh graphs or grid graphs), which are environments where every location in the environment is connected to other locations in a regular grid.
- In NetLogo, *patches* are located on a 2D lattice underlying the world of the ABM (see figure 5.9 for a colorful pattern of patches whose code is simply ASK PATCHES [SET PCOLOR PXCOR * PYCOR]). This uniform connectivity makes them different from network-based environments.

Continuous Spaces

- In a continuous space, there is no notion of a cell or a discrete region within the space. Instead, agents within the space are located at points in the space. These points can be as small as the resolution of the number representation allows.
- In a continuous space, agents can move smoothly through the space, passing over any points in between their origin and destination, whereas in a discrete space, the agent moves directly from the center of one cell to another.

Boundary Conditions

- One other factor that comes into play when working with spatial environments is how to deal with boundaries, an issue for Hex and Square lattices as much as for continuous spaces.
- If an agent reaches a border on the far left side of the world and wants to go farther left, what happens?
- There are three standard approaches to this question, referred to as *topologies* of the environment: (1) it reappears on the far right side of the lattice (toroidal topology); (2) it cannot go any farther left (bounded topology); or (3) it can keep going left forever (infinite plane topology).

Network-Based Environments

- In many real-world situations, especially in social contexts, interactions between agents are not defined by physical geography.
- For instance, rumors do not spread between individuals in a strictly geographical manner. If I call a friend of mine in Germany and tell her a rumor, it spreads to Germany without passing through all of the people between Germany and me.
- In many cases, we want to represent the way individuals communicate by using a network-based environment. Using a network-based environment, we can represent the fact that I called my friend in Germany by drawing a link between the agents that represent each of us in our model.
- A *link* is defined by the two ends it connects, which are frequently referred to as *nodes*. These terms are used in the rapidly growing field of *network science* (Barabási, 2002; Newman, 2010; Watts & Strogatz, 1998).
- Note that mathematical graph theory literature uses different to refer to essentially these same objects, whereby a graph (network) consists of vertices (nodes) that are connected by edges (links).
- However, we will use the network/node/link vocabulary throughout this text.

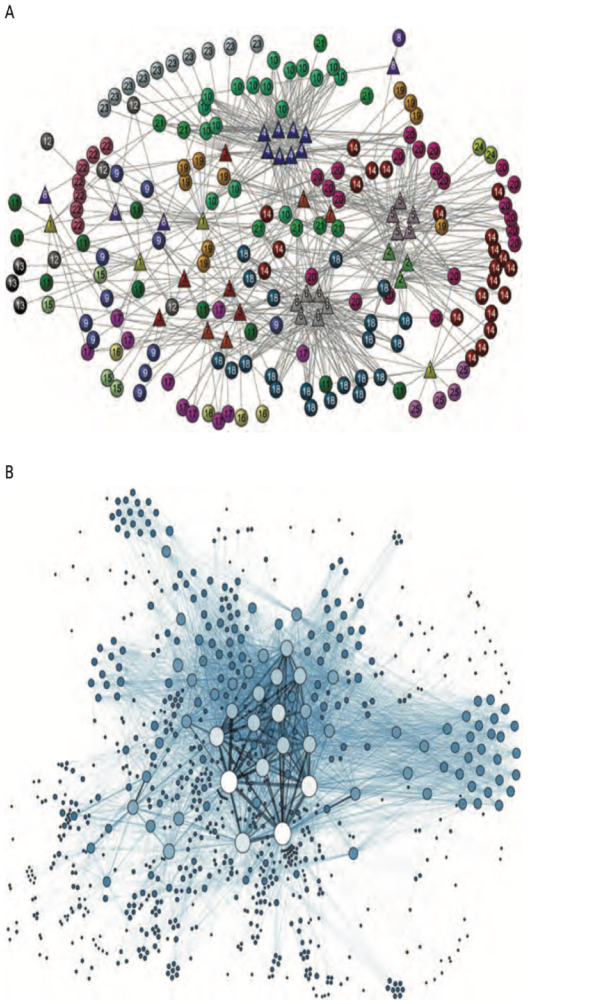


Figure 5.13
(A) Bipartite network of cancers and protein complexes. (B) Network of archive documents shared by League of Nations personnel.

Random Network

- In *random networks*, each individual is randomly connected to other individuals. These networks are created by randomly adding links between agents in the system.
- For example, if you had a model of agents moving around a large room and connected every agent in the room to another agent based on which agent had the next largest last two digits of their social security number, you would probably create a random network.
- The mathematicians Erdös and Rényi (1959) pioneered the study of random networks and described algorithms for generating them.
- We show one simple methods for creating a random network. This code is also in the Random Network model in the chapter 5.

Random Network

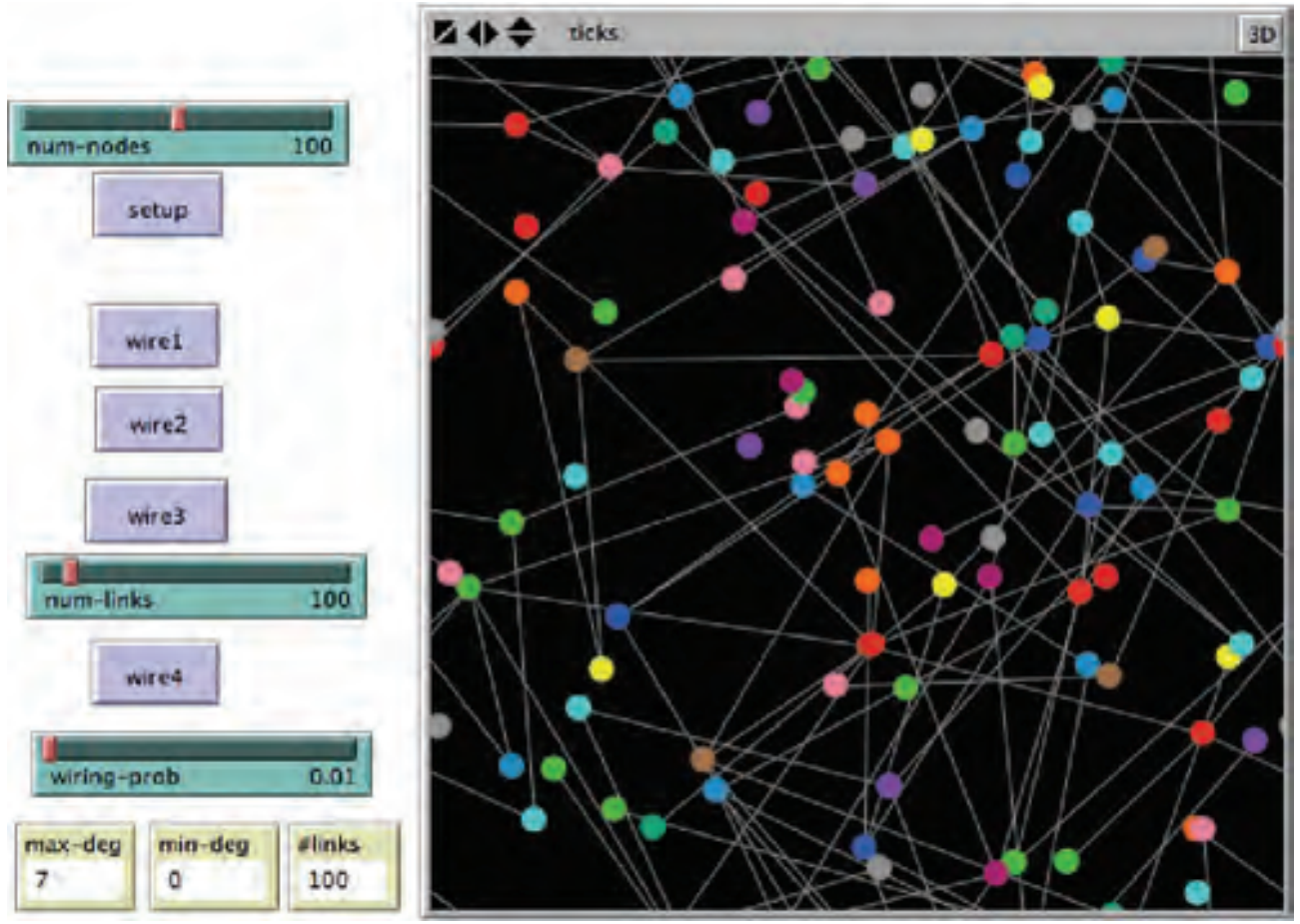
```
to setup
    ca
    crt 100 [
       setxy random-xcor random-ycor
    ]
end

to wire1
    ask turtles [
       create-link-with one-of other turtles
    ]
end
```

- In this code, we create a group of turtles and place them randomly on the screen.
- We then ask each turtle to create a link to another turtle chosen randomly.
- If we want the turtles to have more than one link we can ask each turtle to REPEAT this process several times.
- This code produces a network where each node has at least one link, meaning there are no isolates (i.e., nodes with no links).
- The Random network model shows several other ways to create random networks, including the classic Erdös-Rényi network.

Random Network

http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/IABM%20Textbook/chapter%205/Random%20Network.nlogo

A



Scale-free networks

- *Scale-free networks* have the property that any subnetwork of the global network has the same properties as the global network.
- A common way to create this type of network is by adding new nodes and links to the system so that extant nodes with a large number of links are more likely to receive new connections (Barabási, 2002).
- This technique is sometimes called *preferential attachment*, since nodes with more connections are attached to preferentially.
- This method for network creation tends to produce networks with central nodes that have many radiating links; because of the resemblance to a bicycle wheel, this network structure is sometimes also called hub-and-spoke.
- Many real-world networks such as the Internet, electricity grids, and airline routes have similar properties to a scale-free network.
- To create a scale-free network, we start by creating a couple of nodes and linking them. This code is also in the Preferential Attachment Simple model in the chapter 5.

Scale-free networks

http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/IABM%20Textbook/chapter%205/Preferential%20Attachment%20Simple.nlogo

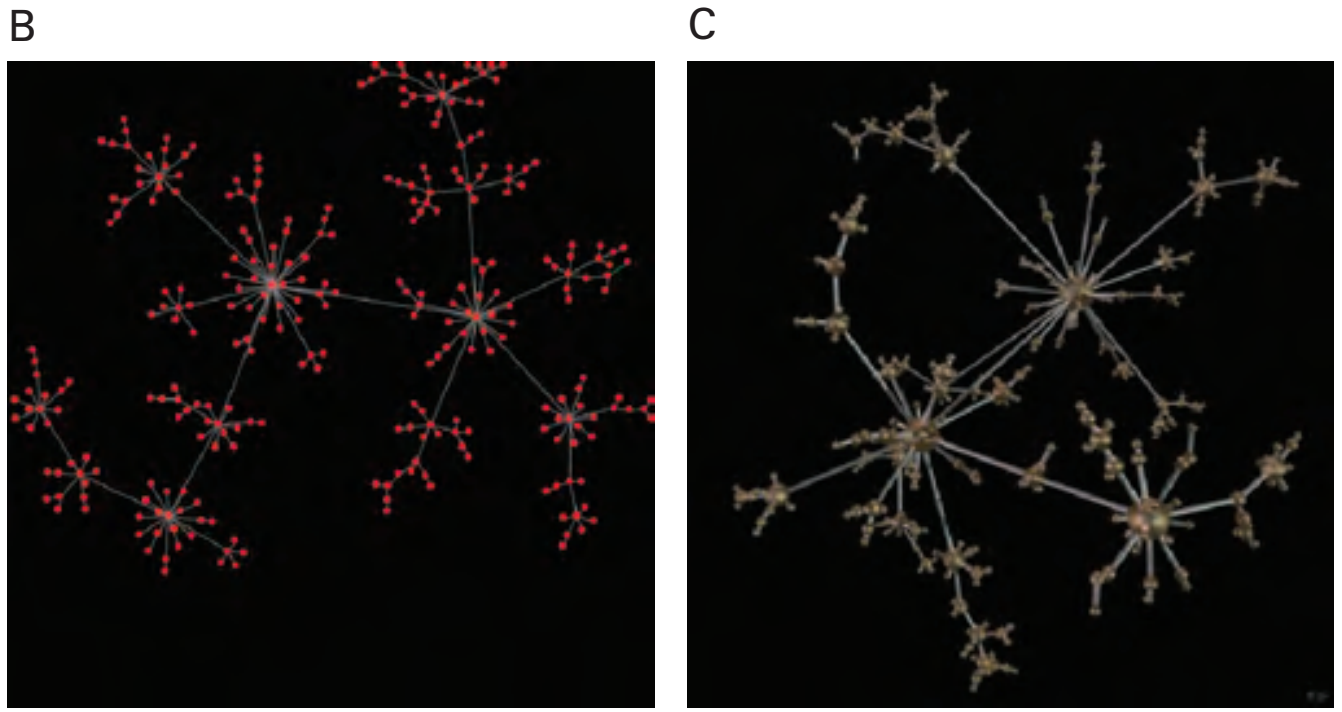


Figure 5.15
(A) Random Network model showing classic Erdös-Rényi random network. (B, C) Scale-Free Network from Preferential Attachment model. http://ccl.northwestern.edu/netlogo/models/PreferentialAttachment (Wilensky, 2001).

3D world There is a version of NetLogo called NetLogo 3D (it is a separate application in the NetLogo folder) that allows modelers to explore ABMs in three dimensions.

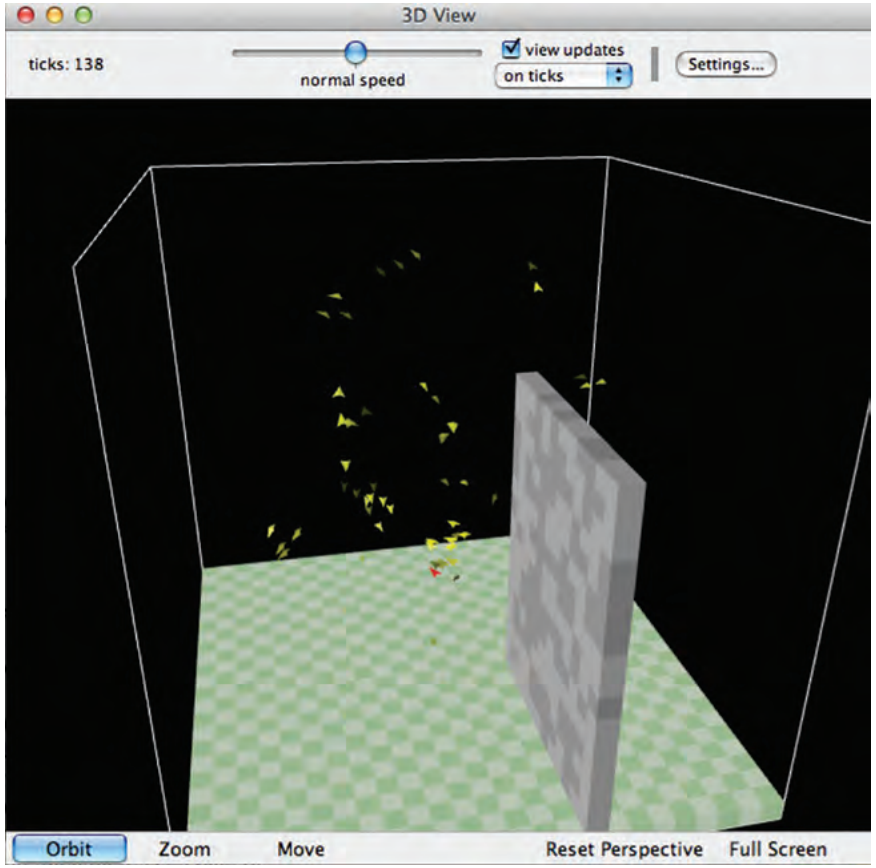


Figure 5.19
The 3D model, Flocking 3D Alternate, in which a flock of birds move around a wall. http://ccl.northwestern.edu/netlogo/models/Flocking3DAlternate (*Wilensky, 2005*).

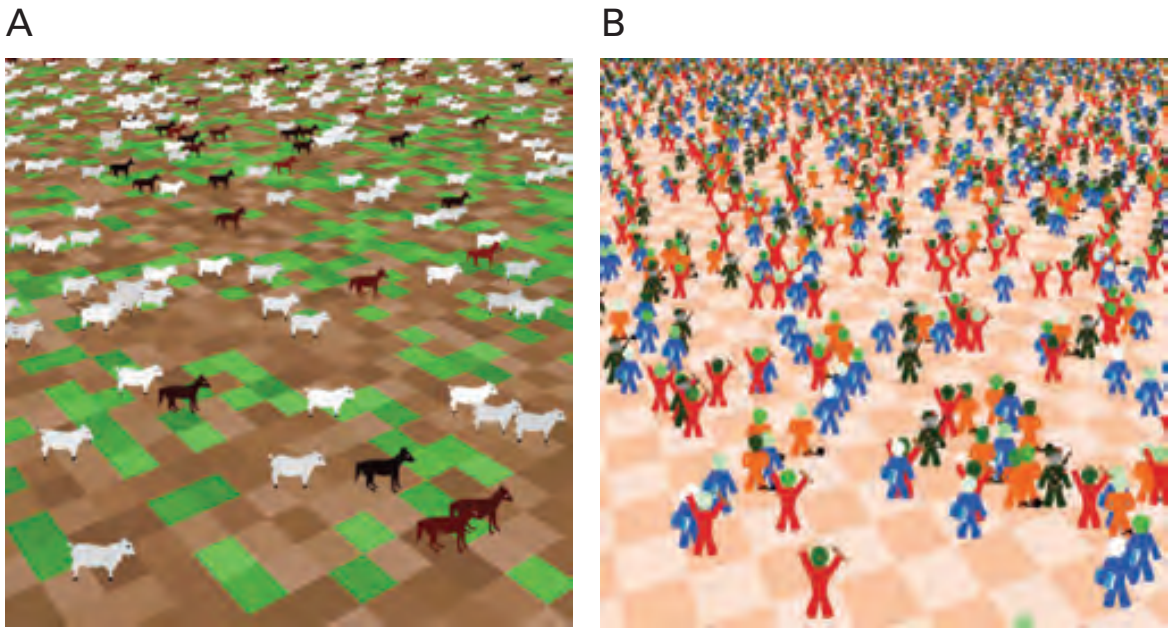


Figure 5.22
3D views of 2D models. (A) Wolf Sheep Predation model. (B) Rebellion model.

GIS-based Geographic Information Systems (GIS) are environments that record large amounts of data that are related to physical locations in the world.

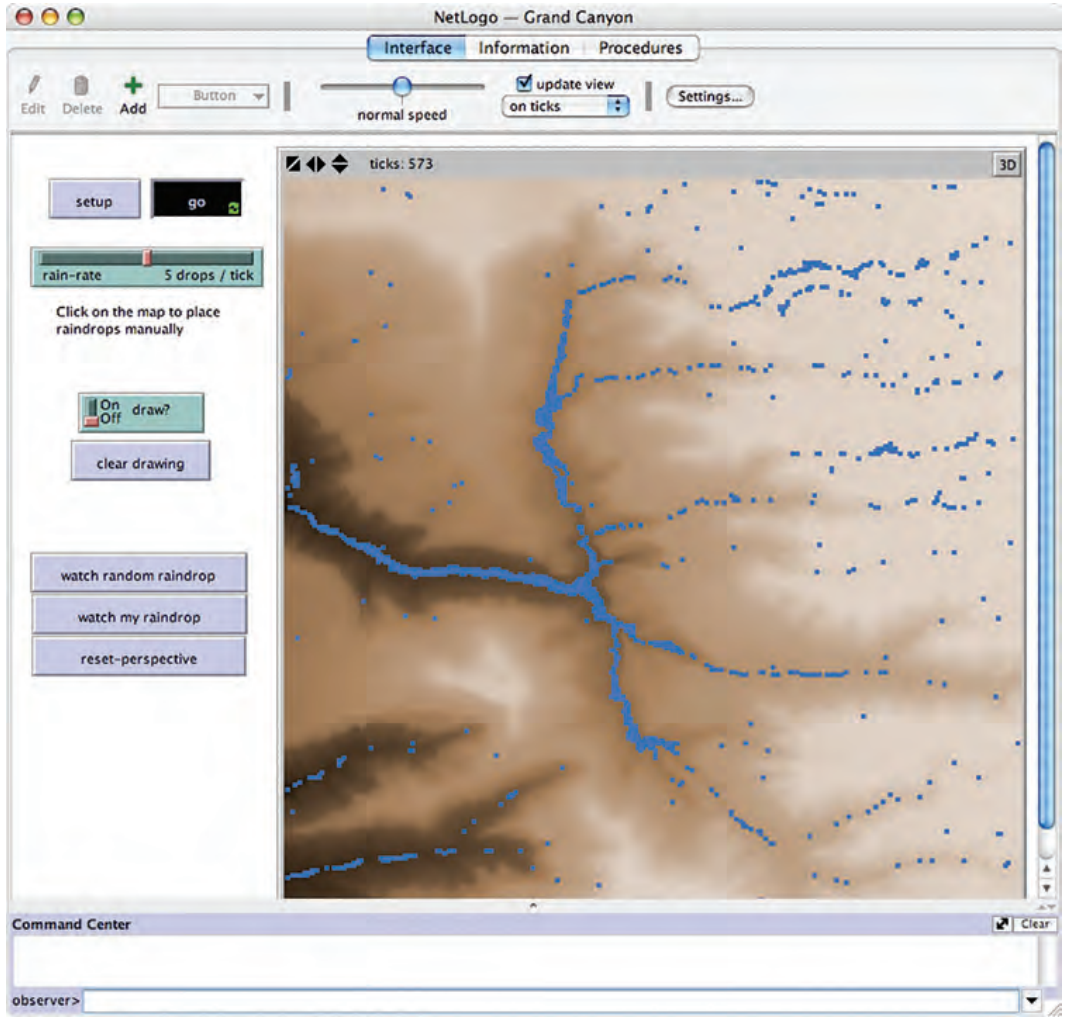


Figure 5.20
Grand Canyon model. http://ccl.northwestern.edu/netlogo/models/GrandCanyon (*Wilensky, 2006*).

Summary

We have described five classes of ABM components: agents, environments, interactions, observer/user interface, and schedule. These five classes of components of ABMs are described at a conceptual level. When creating your own models, you will find that you will need many instances of components, but they will all fall within these five general categories. Agents can have many different properties and behaviors, and there are many kinds of agents, but they are the basic component of an agent-based model, no agents, no ABM. The environment is where agents reside, and thus adequately describing the environment is important to an agent-based model. Interactions are how the dynamics of the model evolve, and therefore are critical to the operation of an agent-based model. The observer/interface is how the model is controlled and how data is extracted from the model.

6. Analyzing Agent-Based Models

- We will learn to employ ABM to produce new and interesting results about the domain that we are investigating.
- What kinds of results can ABMs produce? There are many different ways of examining and analyzing ABM data.
- Choosing just one of these techniques can be limiting; therefore, it is important to know the advantages and disadvantages of a variety of tools and techniques.
- It is often useful to consider your analysis methods *before* building the ABM, to enable you to design output that is conducive to your analysis.

Modeling the Spread of Disease

- If someone catches a cold and is coughing up a storm, he might infect others. Those that he comes into contact with—his friends, co-workers, and even strangers—may catch the cold.
- If a cold virus infects someone, that person might spread that disease to five other people (six now infected) before they recover. In turn, those five other people might spread the cold to five more people each (thirty-one are now infected), and those twenty-five people might spread the cold to five additional people (a hundred and fifty-six people are now infected).
- In fact, the rate of infection initially rises exponentially.

Modeling the Spread of Disease

http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/IABM%20Textbook/chapter%206/Spread%20of%20Disease.nlogo

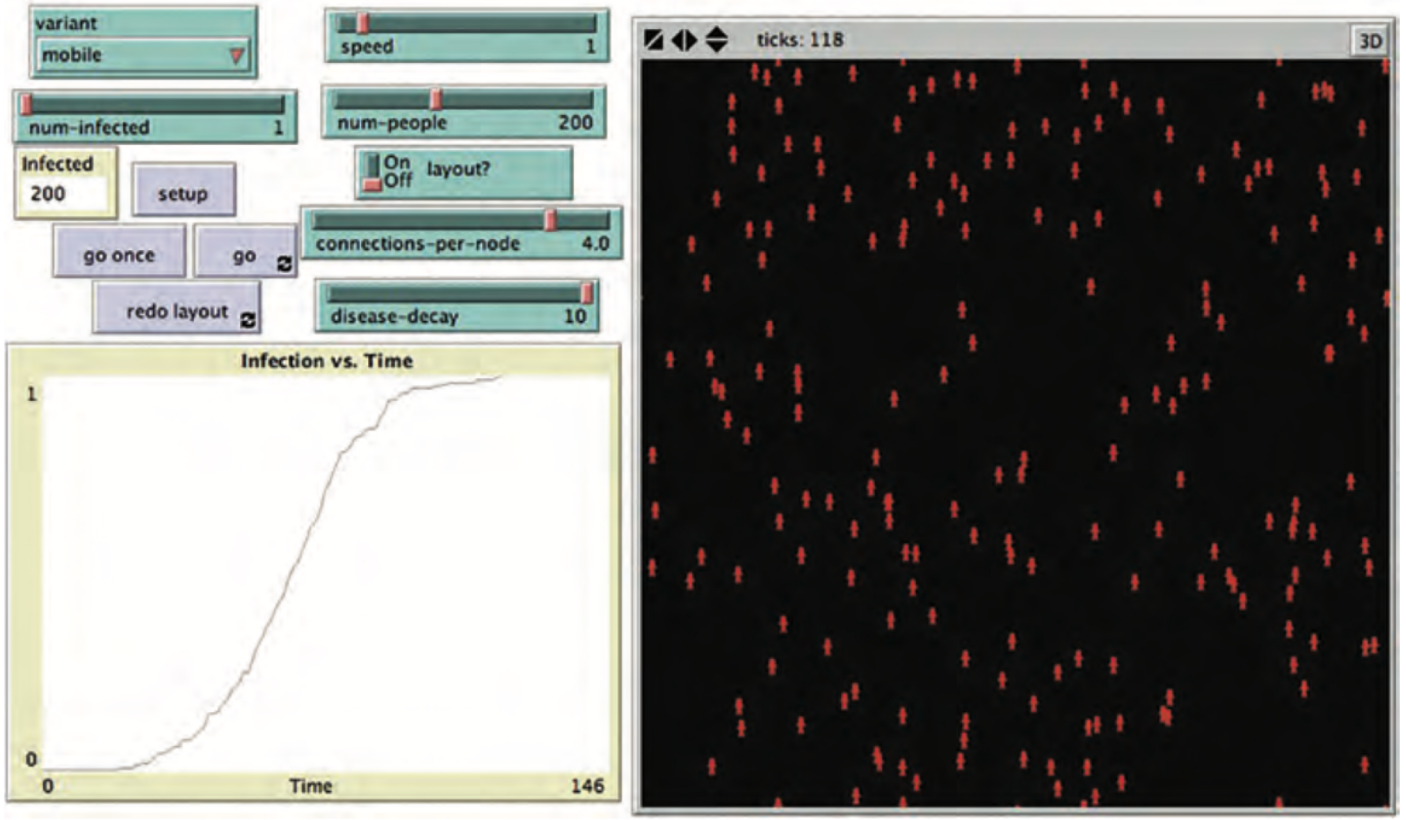


Figure 6.1
Spread of Disease model.

Modeling the Spread of Disease

WHAT IS IT?

- This model explores the spread of disease in a number of different conditions and environments. In particular, it explores how making assumptions about the interactions of agents can drastically affect the results of the model.

HOW IT WORKS

- The SETUP procedure creates a population of agents. Depending on the value of the VARIANT chooser, these agents have different properties. In the NETWORK variant they are linked to each other through a social network. In the MOBILE and ENVIRONMENTAL variants they move around the landscape. At the start of the simulation, NUM-INFECTED of the agents are infected with a disease.
- The GO procedure spreads the disease among the agents. In the case of the NETWORK variant this is along the social network links. In the case of the MOBILE or ENVIRONMENTAL variant, the disease is spread to nearby neighbors in the physical space. In the case of the ENVIRONMENTAL variant the disease is also spread via the environment. Finally, if the variant is either the MOBILE or ENVIRONMENTAL variant then the agents move.

Modeling the Spread of Disease

HOW TO USE IT

- The NUM-PEOPLE slider controls the number of people in the world.
- The VARIANT chooser controls how the infection spreads.
- NUM-INFECTED controls how many individuals are initially infected with the disease.
- The CONNECTIONS-PER-NODE slider controls how many connections to other nodes each node tries to make in the NETWORK variant.
- The DISEASE-DECAY slider controls how quickly the disease leaves the current environment.
- To use the model, set these parameters and then press SETUP.
- Pressing the GO ONCE button spreads the disease for one tick. You can press the GO button to make the simulation run until all agents are infected.
- The REDO LAYOUT button runs the layout-step procedure continuously to improve the layout of the network.

Modeling the Spread of Disease

THINGS TO NOTICE

- How do the different variants affect the spread of the disease?
- In particular, look at how the different parameters of the model influence the speed at which the disease spreads through the population. For example, in the "mobile" variant, the population (NUM-PEOPLE) clearly seem to be the main driving force for the speed of infection. Is that the case for the other two variants as well? Some suggestions of parameters to vary are given below under THINGS TO TRY.
- Another thing that you may have noticed is that, in the "network" variant, there are cases where the disease will not spread to all people. This happens when the network has more than one components (isolated nodes, or groups of nodes that are not connected with the rest of the network) and that not all components get infected with the disease right from the start. NetLogo's network extension has a primitive that can help you identify the components of a network.

Modeling the Spread of Disease

THINGS TO TRY

- Set different values for the DISEASE-DECAY slider and run the ENVIRONMENTAL variant. How does the DISEASE-DECAY slider affect the results?
- Similarly, set different values for the CONNECTIONS-PER-NODE slider and run the NETWORK variant. How does the CONNECTIONS-PER-NODE slider affect the results?
- Though this model is simple, it exhibits interesting and complex behavior. For instance, what happens if we increase the number of people in the model? Does the disease spread quicker throughout the population, or does it take a longer time because there are more people? Let us run the model at population sizes of 50, 100, 150, and 200, and examine the results.

Modeling the Spread of Disease

EXTENDING THE MODEL

- Can you think of additional variants and parameters that could affect the spread of a disease?
- At the moment, in the environmental variant of the model, patches are either infected or not. DISEASE-DECAY allows you to set how long they stay infected, but they are fully contagious until they suddenly stop being infected. Do you think it would be more realistic to have their infection level decline gradually? The probability of a person catching the disease from a patch could become smaller as the infection level decreases on the patch. If you want to make the model look really nice, you could vary the color of the patch using the scale-color primitive.

NETLOGO FEATURES

- One particularity of this model is that it combines three different "variants" in the same model. The way this is accomplished in the code of the model is fairly simple: we have a few if-statements making the model behave slightly different, depending on the value of the VARIANT chooser.
- A more interesting element is the **Infection vs. Time** plot. In the "mobile" and "network" variants, the plot is the same: we simply plot the number of infected persons. In the "environmental" variant, however, we want to plot an additional quantity: the number of infected patches. To achieve that, we use the "Plot update commands" field of our plot definition. Just like the "Pen update commands", these commands run every time a plot is updated (usually when calling tick). In this case, we use the create-temporary-plot-pen primitive to make sure that we have a pen for the number of infected patches, and actually plot that number:
- if variant = "environmental" [create-temporary-plot-pen "patches" plotxy ticks count patches with [p-infected?] / count patches]
- One nice thing about this NetLogo feature is that the temporary plot pen that we create is automatically added to the plot's legend (and removed from the legend when the plot is cleared, when calling clear-all).
- If you open the BehaviorSpace tool, you will see that we have a defined a few experiments that can be used to explore the behavior of the model more systematically. Try these out, and look at the data in the resulting CSV file. Are those results similar to what you obtained by manually playing with the model parameters? Can you confirm that using your favorite external analysis tool?

Though this model is simple, it exhibits interesting and complex behavior. For instance, what happens if we increase the number of people in the model?

Table 6.1
Infection Data

Population	50	100	150	200
Time to 100% Infection	419	188	169	127

Table 6.2
Your Friend's Data

Population	50	100	150	200
Time to 100% Infection	305	263	118	126

Table 6.3
Raw Data

Population	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Run 8	Run 9	Run 10
50	419	365	305	318	323	337	432	380	430	359
100	188	263	256	205	206	205	201	181	202	231
150	169	118	163	146	143	167	137	121	140	140
200	127	126	113	111	133	129	109	101	105	133

The Necessity of Multiple Runs within ABM

- As illustrated earlier, when you are trying to collect statistical results from an ABM you should run the model multiple times and collect different results at different points.
- Most ABM toolkits will provide you with a way to collect the data from these runs automatically.

Statistical Analysis of ABM: Moving beyond Raw Data

- Statistical results are the most common way of looking at any kind of scientific data whether it is computational models, physical experiments, sociological surveys, or other methods that generate data.
- The general methodology behind descriptive statistics is to provide numerical measures that summarize a large data set and describe the data set in such a way that it is not necessary to examine every single value.

Sensitivity Analysis and Robustness

- After we have finished building a model and found some interesting results, it is important to explore these results to determine how sensitive our model is to the particular set of initial conditions that we are using.
- Sometimes, this just means varying a group of the parameters that we already have within our model, but other times, this entails adding new parameters to the model. This process, called *sensitivity analysis*, creates an understanding of how sensitive (or robust) the model is to various conditions.

Table 6.4
Summary Statistics

Population	Mean	Std. Dev.
50	366.8	47.39385802
100	213.8	27.40154091
150	144.4	17.65219533
200	118.7	12.12939497

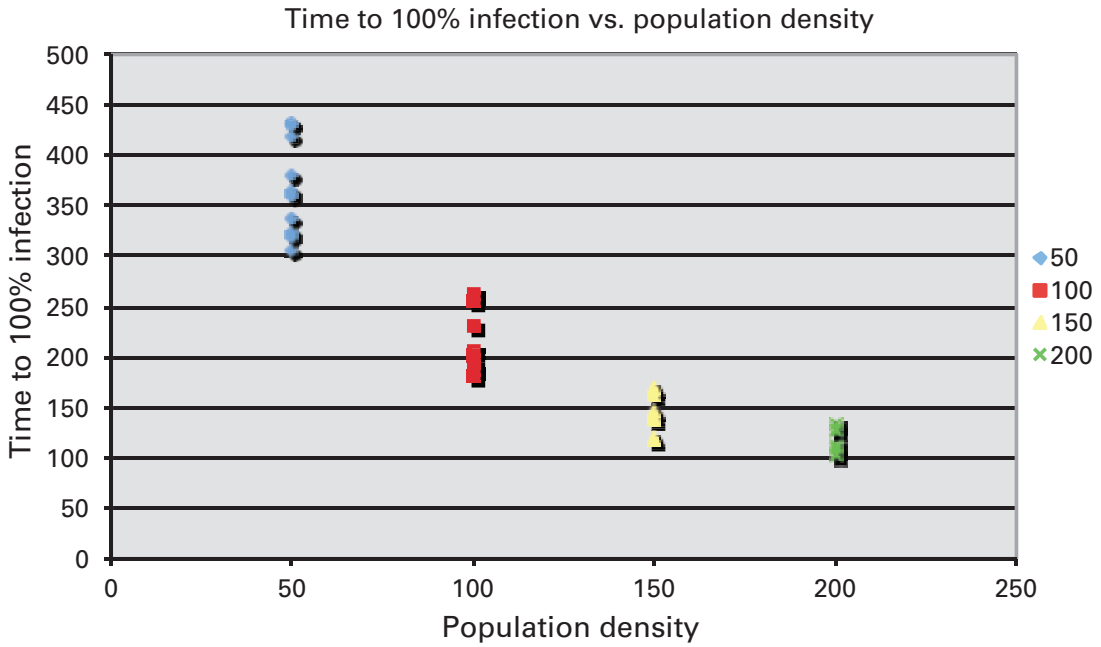


Figure 6.4
All raw data in a graph-based form.

Analyzing Networks within ABM

- As we mentioned in chapter 5, having agents walking around and interacting is useful if you want to examine physically proximate interactions, but many types of interactions do not occur in physical space, but rather across social networks.
- The Spread of Disease model we have explored relied on contact between moving agents to spread the infection. Diseases do indeed spread in part by people walking around and infecting other people, but the spread of diseases relies heavily on the social network of individuals.
- In fact, some diseases such as sexually transmitted diseases are not spread by casual contact at all but only through certain kinds of social networks. The Spread of Disease model was designed to explore that possibility as well.
- There is an interface element, the chooser, which allows you to select different variants of this model. When the chooser is set to “network,” instead of agents moving around on the plane, they are connected via a network, and the disease spreads over the network. The network that is created in this model is a particular type of network, a random graph,5 which we saw in the previous chapter (in this case an Erdös-Rényi random graph [1959]).

Table 6.6
Number of Agents Infected after 50 Ticks

connections-per-node	mean # infected	std. dev.
0.5	1.8	1.229272594
1	15.1	21.75852323
1.5	68	59.11946474
2	145.3	50.64922946
2.5	181.1	3.348299734
3	189.3	3.093002856
3.5	174.5	61.03050239
4	196.4	2.170509413

Environmental Data and ABM

- After looking at the model for some time, we might realize that for the diseases we are interested in studying, such as the common cold, the mobile agent model makes more sense. However, the mobile agent model did not go far enough.
- Besides agent-to-agent transmission of cold germs, there can also be agent-to-environment and environment-to- agent transmission.
- For instance, if someone has a bad cold and wipes his nose with his hand, and then opens a door, someone else who comes through that door shortly after the person with the disease might catch the cold.
- Of course, germs do not live very long outside the body so these environmental infections might decrease after time; so we would want the model to reflect this.

Table 6.7
Time to 100% Infection, Environmental Variant

disease-decay	Average	Std. Dev
0	126.4	12.2854928
1	71	4.988876516
2	62	7.363574011
3	51	4.242640687
4	51.2	2.780887149
5	49.4	2.716206505
6	49.9	2.643650675
7	46.5	2.758824226
8	48.5	3.341656276
9	47.4	3.062315754
10	47.3	2.213594362

RELATED MODELS

- NetLogo is very good at simulating the spread of epidemics, so there are a few disease transmission model in the library:
- HIV
- Disease Solo
- Disease HubNet
- Disease Doctors HubNet
- epiDEM Basic
- epiDEM Travel and Control
- Virus on a Network

7. Verification, Validation, and Replication

Essentially, all models are wrong, but some are useful. —George Box

Correctness of a Model

- If a model is to be useful for answering real-world questions, it is important that the model provides outputs that address the relevant issues and that the outputs are accurate.
- The model must provide outputs that are useful to the model user. Model accuracy can be evaluated through three different modeling processes: validation, verification, and replication.
- *Model validation* is the process of determining whether the implemented model corresponds to, and explains, some phenomenon in the real world.
- *Model verification* is the process of determining whether an implemented model corresponds to the target conceptual model. This process is equivalent to making sure that the model has been correctly implemented.
- Last, *model replication* is the implementation by one researcher or group of researchers of a conceptual model previously implemented by someone else.

Validation

- *Validation* is the process of ensuring that there is a correspondence between the implemented model and reality. Validation, by its nature, is complex, multilevel, and rela- tive. Models are simplifications of reality; it is impossible for a model to exhibit all of the same characteristics and patterns that exist in reality. It is important to keep the conceptual model questions in mind and validate aspects of the model that relate to these questions.
- There are two different axes along which to consider validation issues (Rand & Rust, 2011). The first axis is the level at which the validation process is occurring. *Microvalidation* is making sure the behaviors and mechanisms encoded into the agents in the model match up with their real world analogs. *Macrovalidation* is the process of ensuring that the aggregate, emergent properties of the model correspond to aggregate properties in the real world.
- The second axis of validation is the level of detail of the validation process. *Face validation* is the process of showing that the mechanisms and properties of the model look like mechanisms and properties of the real world. *Empirical validation* makes sure that the model generates data that can be demonstrated to correspond to similar patterns of data in the real world.

THANK you