Complex Systems 530: Computer Modeling of Complex Systems

Lecture 2: Intro to agent-based models

Readings!

- Read
  - “Why Model?”, Epstein 2008
  - Sayama, Chp. 1-2
  - “More is Different,” Anderson 1987
  - Wilensky, Chp. 0-1
- Check-in: Everyone has NetLogo & Python installed?

Top-down vs. bottom-up

- **Top-down**: start with the understanding of the larger system, then break-down or decompose into smaller subsystems

- **Bottom-up**: start with micro-level processes and build up to emergent behaviors at the macro level

Top-down vs. bottom-up

- Related to Hayek's (1973) consideration of two different Greek conceptions of "order":

- **Taxis**: An arranged, top-down order. A "made" or "designed" order, purposefully built and imposed by a part onto the greater whole.

- **Cosmos**: A grown, bottom-up order. An order that arises spontaneously and unintentionally from the interaction of parts within a whole.

Top-down vs. bottom-up

- Many models & modeling frameworks can be built/ thought of from either perspective
- Advantages/disadvantages of each?
- What about in the context of complex systems?
- Top-down/bottom-up often conflated with simple models vs. detailed ones, but not quite true

Agent-based models (ABMs)

Agent-based models have 3 main components

- **Agents** - independent "agents" move, interact, explore environment, etc.

- **Environment** - agents exist in a non-agent environment (can be static or dynamic)

- **Rules/interactions** - to govern agent behavior, how they interact with the environment, etc.

ABM Dynamics

- Hard to build general mathematical theory of ABM dynamics
- Not always so easy to classify
- Not necessarily equilibrium values to calculate
- Phase plane ideas may not be helpful b/c of spatial aspect, etc.

ABM Dynamics

- Wide range of possibilities
- Stable constant steady states
- Repeating patterns (oscillation)
- Organized but non-repeating structure/patterns
- Disorder
- All sorts of things!

Examples & features of ABMs

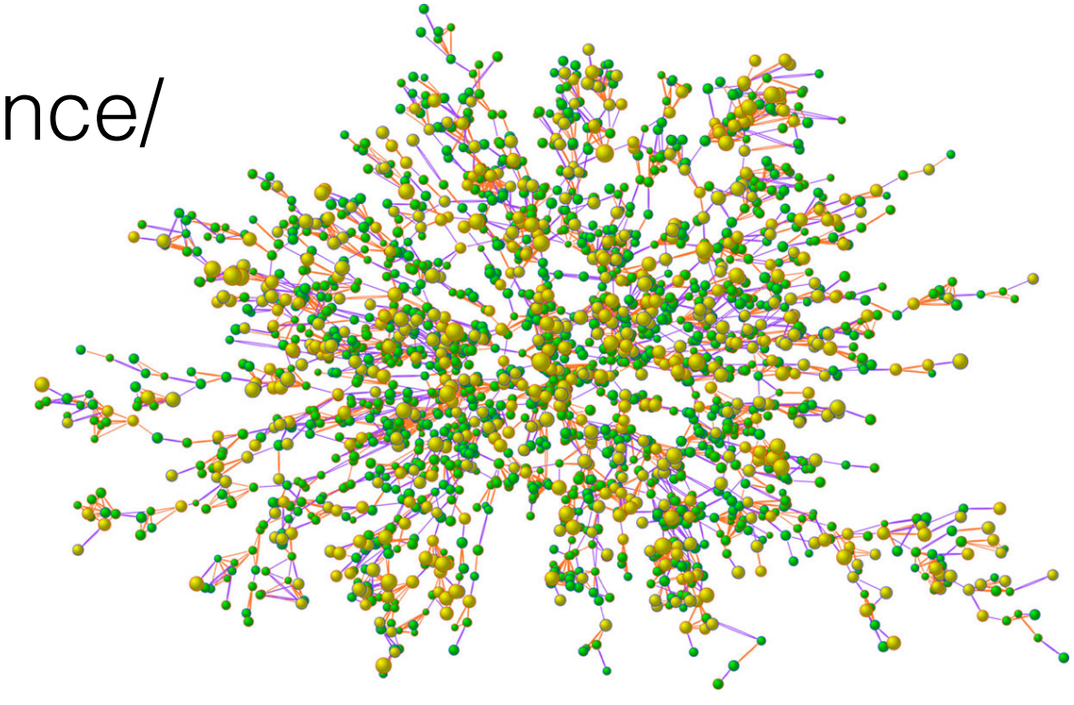
Interactions

- Infectious disease models - account for different mixing patterns, school, work, etc.

- Evolution & invasion of strategies, ideas, memes, species

- Models of markets, voting, violence/ unrest, etc.

- Many dynamic social network models are ABMs too

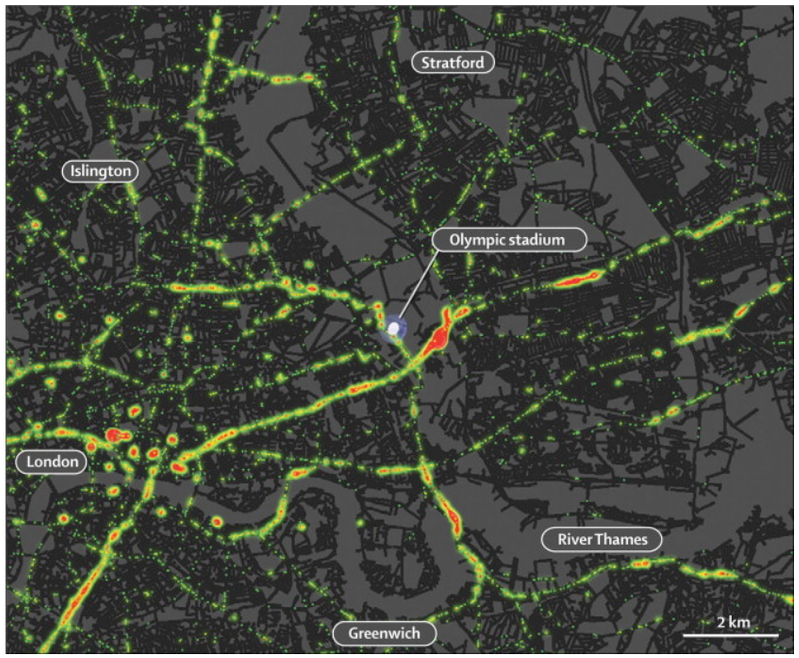


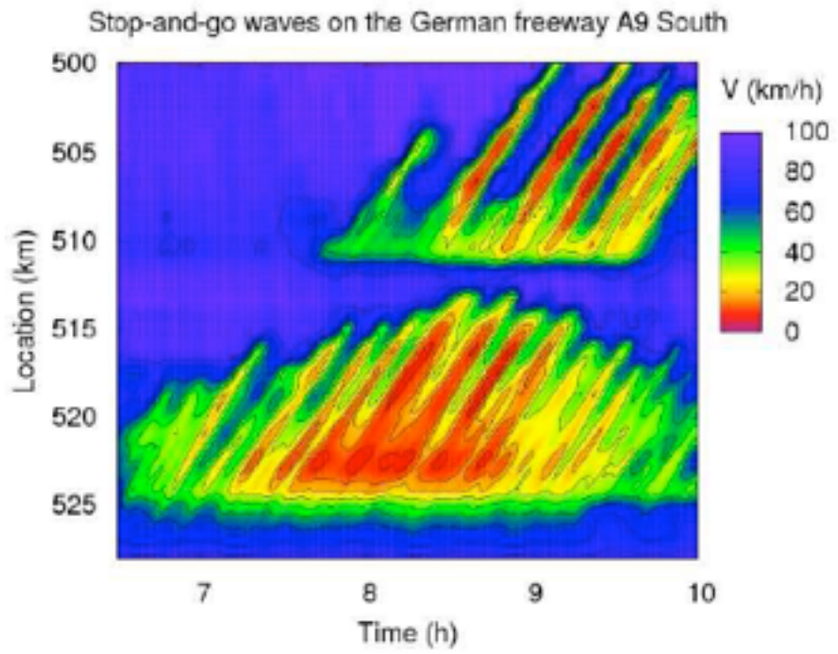
Examples

- Netlogo virus on a network model
- Sugarscape - classic model of interactions between individuals and the environment in a society with spatially distributed resources

Flows & movement

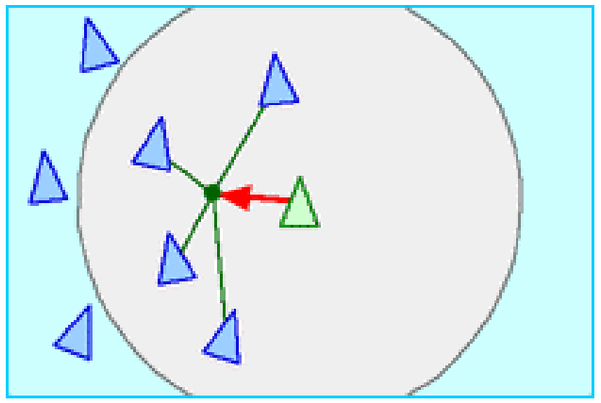
- Swarming, crowd behavior, building evacuation
- Traffic flow - simple rules & shockwaves
- Walking, commuting, migrations, etc.
    - E.g., How does neighborhood structure & distribution of resources affect health?



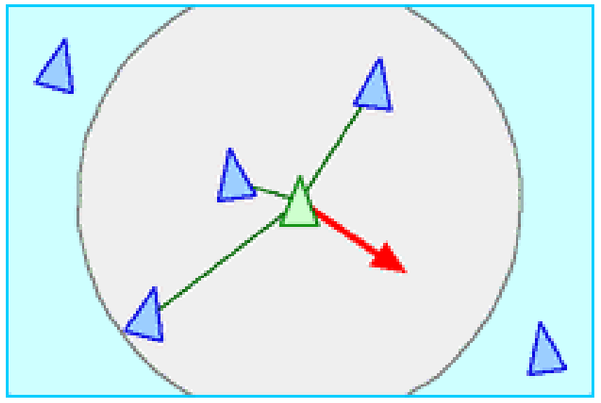


http://www.youtube.com/watch?v=GUkjC-69vaw
http://www.youtube.com/watch?feature=player_embedded&v=PRRJM-Xsg-U

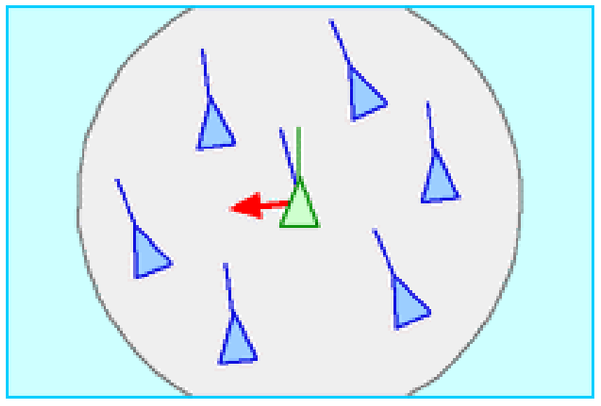
Boids



Cohesion:
Steer to move toward the average position of local flockmates



Separation:
Steer to avoid crowding local flockmates



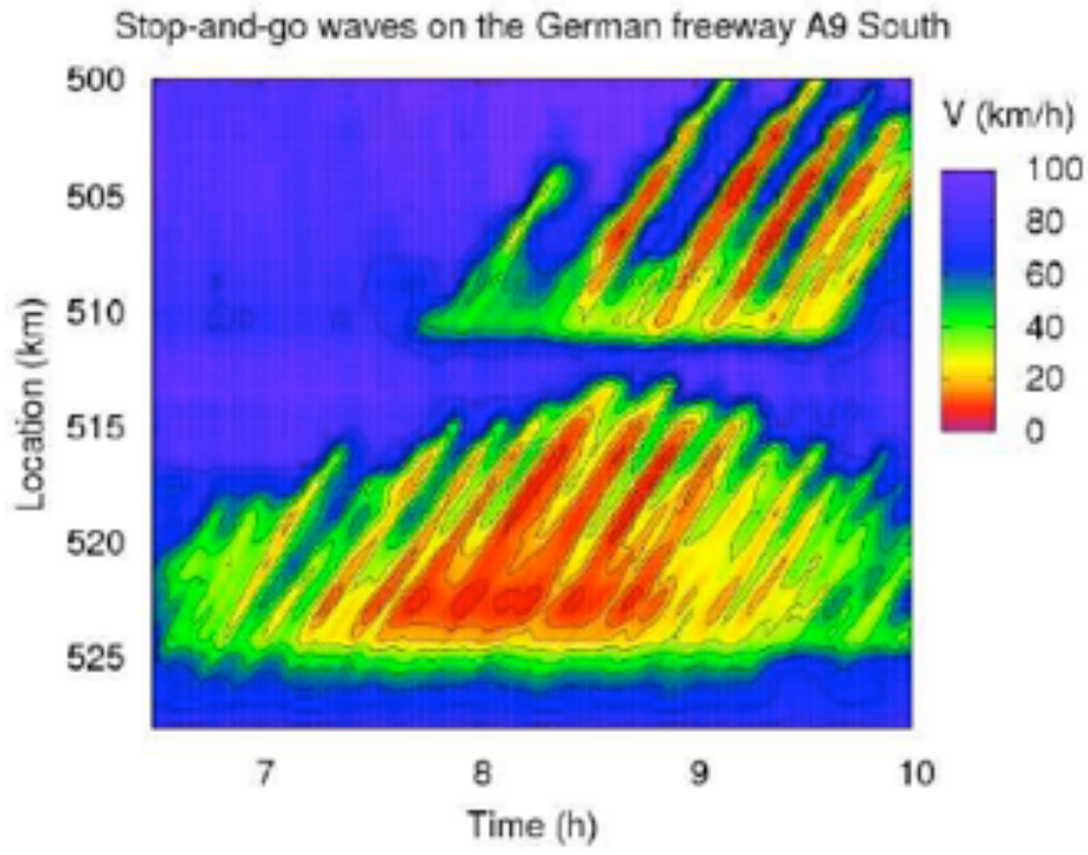
Alignment:
Steer towards the average heading of local flockmates

Source: “Boids” by Craig Reynolds, http://www.red3d.com/cwr/boids/

Traffic Flow

- https://traffic-simulation.de

- Netlogo traffic model



Building ABMs: lots of choices!

- What types of agents? How many? How distributed? What properties do they need?

- What type of environment? Grid? GIS? 1D/2D/3D?

- Major issues/interactions/processes to consider? (movement? agent interactions? etc.)

Building ABMs

- Major Design Decision:
    - Who will be interacting with whom and with what part of the environment?
- Relevant terminology:
    - **Agent's local environment**: local area around the agent
    - **Neighbors**: subset of agents with whom an agent interacts
    - **Neighborhood**: social or spatial areas in the "world" containing an agent's neighbors
    - **Topology**: how agents are connected to one another within the system

Building ABMs

- Agents, Environment, Rules/interactions
- Useful to start by laying out each of these
- Diagrams or tables often useful
- Reproducibility

PARTE Framework

- Properties, Actions, Rules, Time, Environment

- Agents are defined by their properties, actions, and rules

https://www.ncbi.nlm.nih.gov/books/NBK305917/

Features of Agents

- **Discrete, self-contained** - clear boundaries between agents (i.e. they are each distinct elements)

- **Properties or states** - possesses a set of attributes that can change via interactions and which drive its behavior

- **Autonomous** - typically functions independently, takes in local information and executes behaviors based on it

- **Social** - usually interactions of agents with one another influences their behaviors

Features of Agents

- May also be:
  - **Adaptive** - rules that modify behaviors based on accumulated experience (i.e. some form of learning)
  - **Goal-directed** - outcomes can be judged against whether they achieve a goal and future behaviors adjusted accordingly
  - **Heterogeneous** - agents within a model can vary on all previously described dimensions

Agent design is usually broken into two main pieces:

- Agent Properties or Attributes
  - Defines the state of the agent
  - Can be either static or dynamic
  - Captured via “<agent>-owned” variables (NetLogo) or via entries in a data structure (e.g. list, array, dictionary) or class attributes (Python)

Agent design is usually broken into two main pieces:

- Agent Actions or Methods
    - How state and interaction information translates into the behavior of the agent
    - Also used to update attributes
    - Captured via agent executed procedures (NetLogo) or via functions on agent-related data structures or class methods (Python)

Agent design

- Rules of the ABM and its agents govern how and when these properties (attributes) and actions (methods) occur

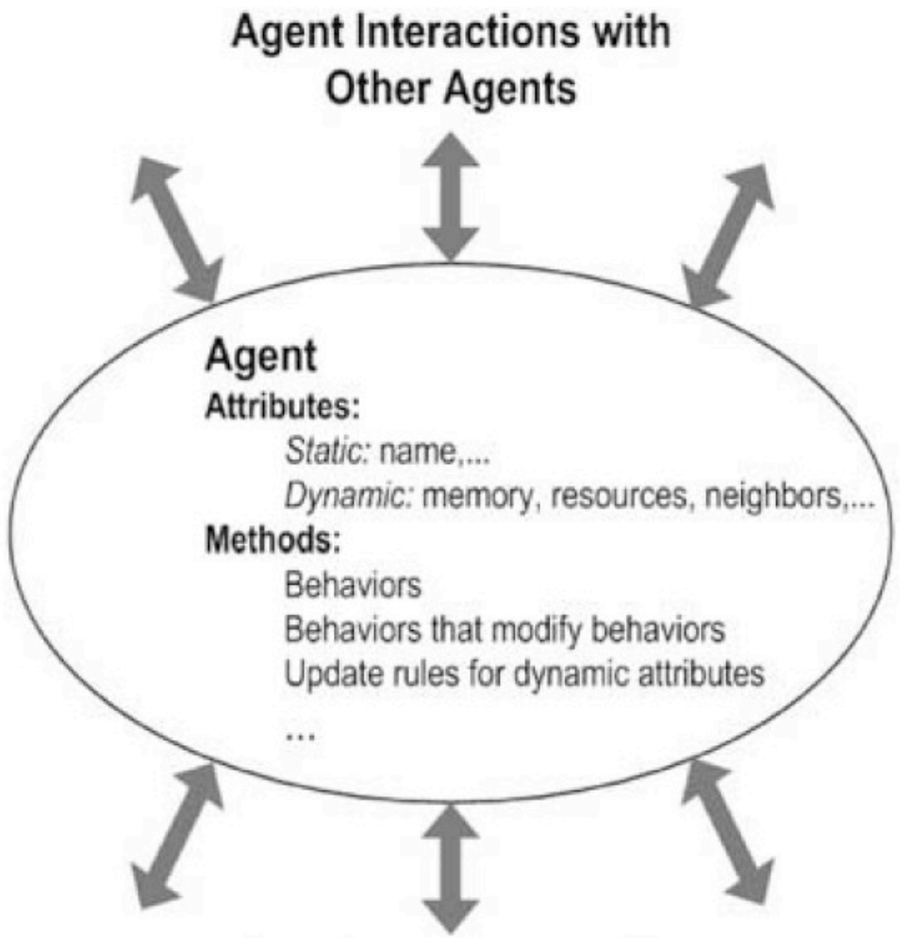
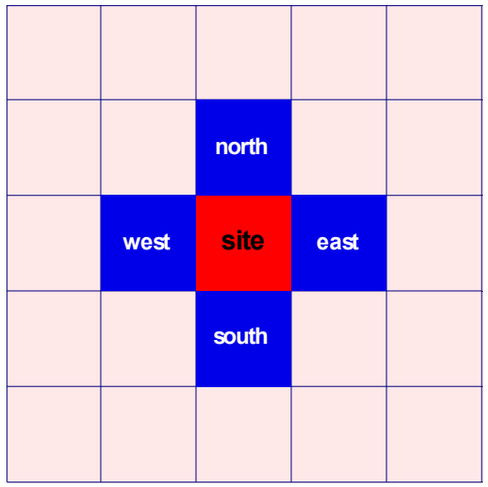


Figure 2 A typical agent.

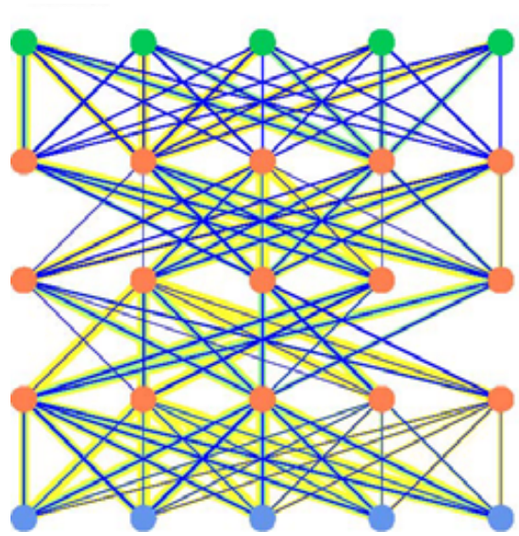
(Macal and North, 2010)

Environments

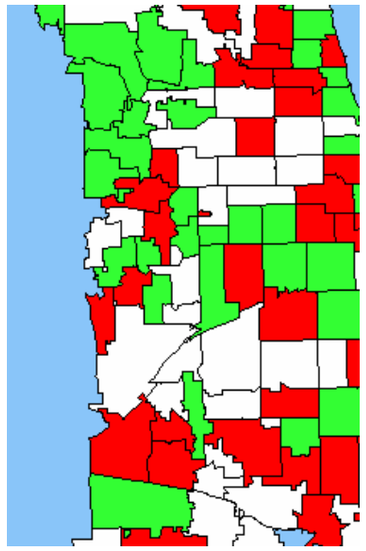
Euclidean Space: *2D, 3D*



Grid: *von Neumann neighborhood*



Network

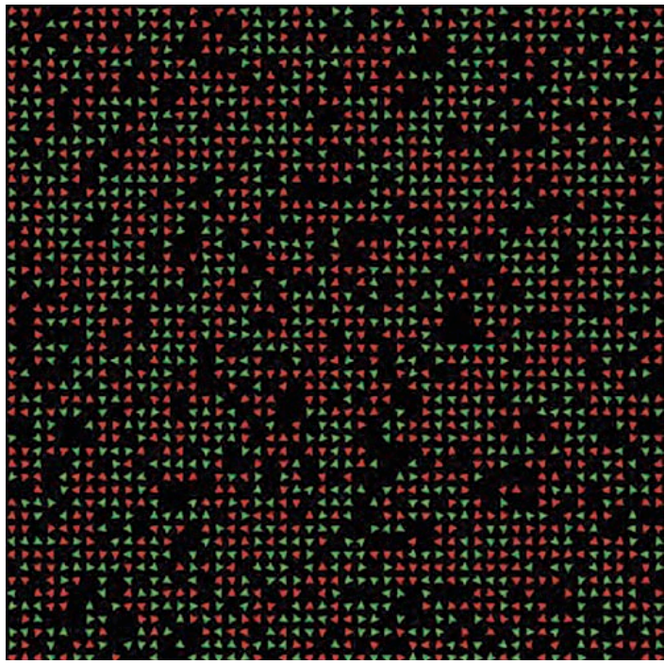


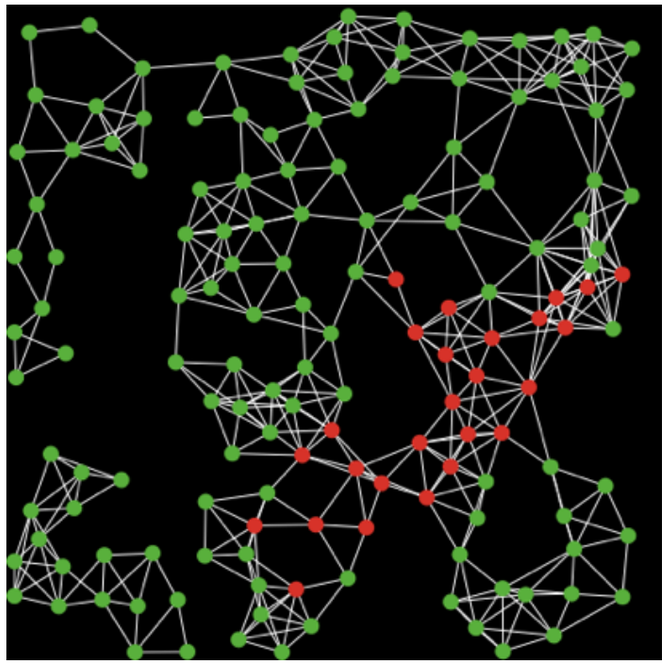
GIS: Geographic Information System

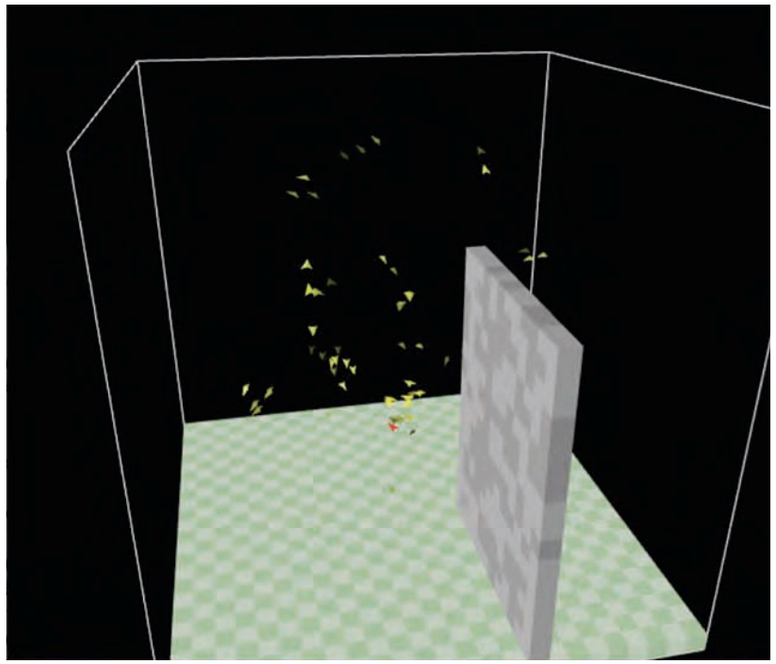
Introduction to Agent-based Modeling and Simulation

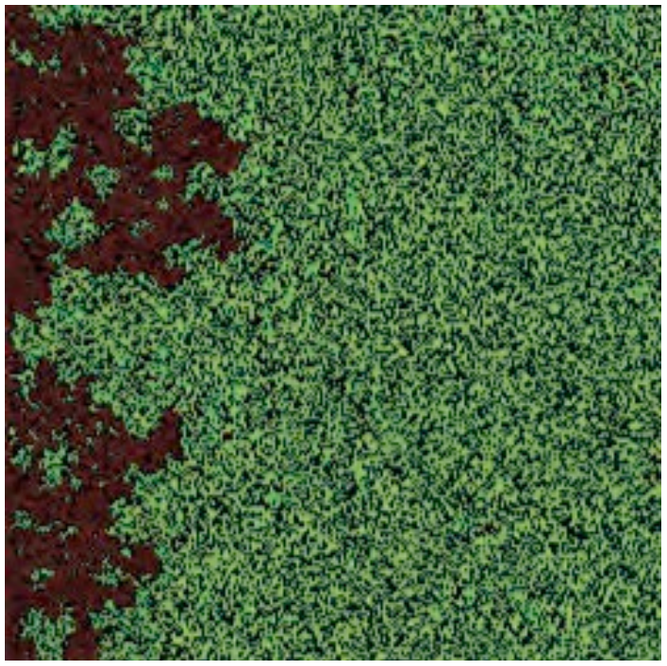
Charles M. Macal and Michael J. North

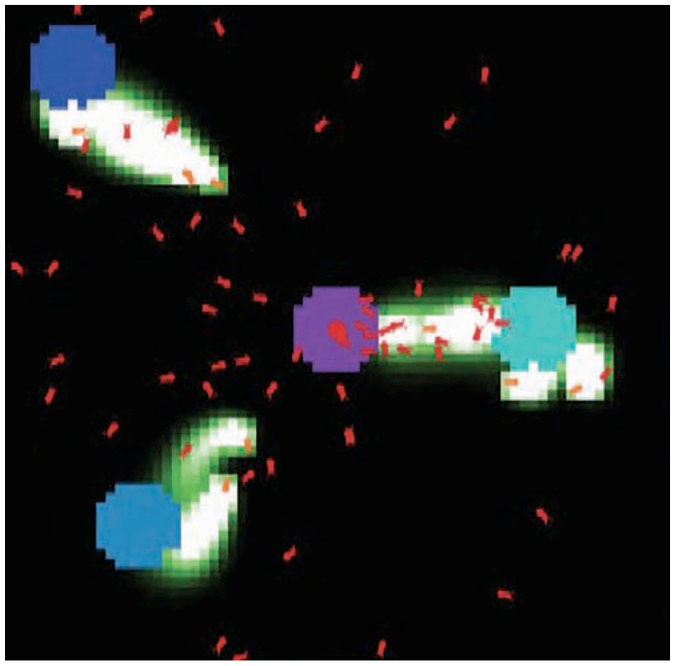
Environments





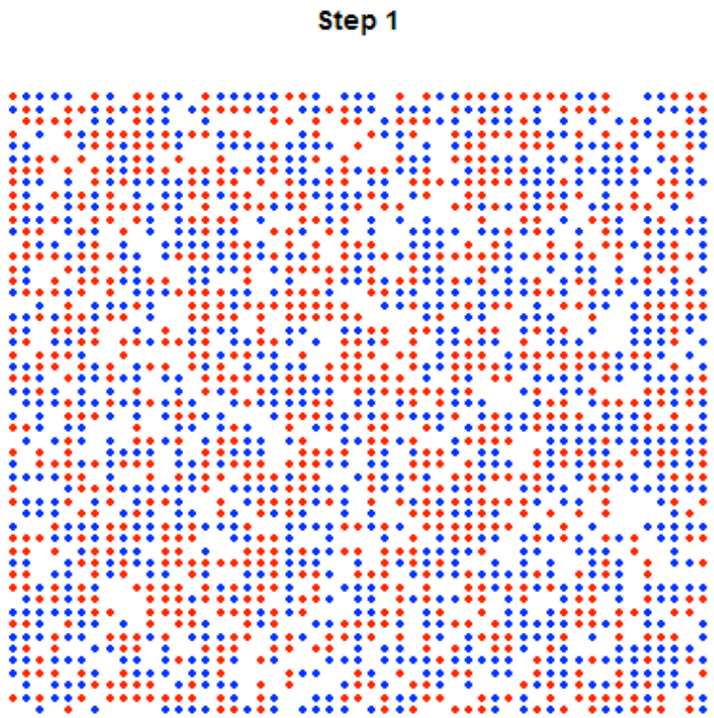




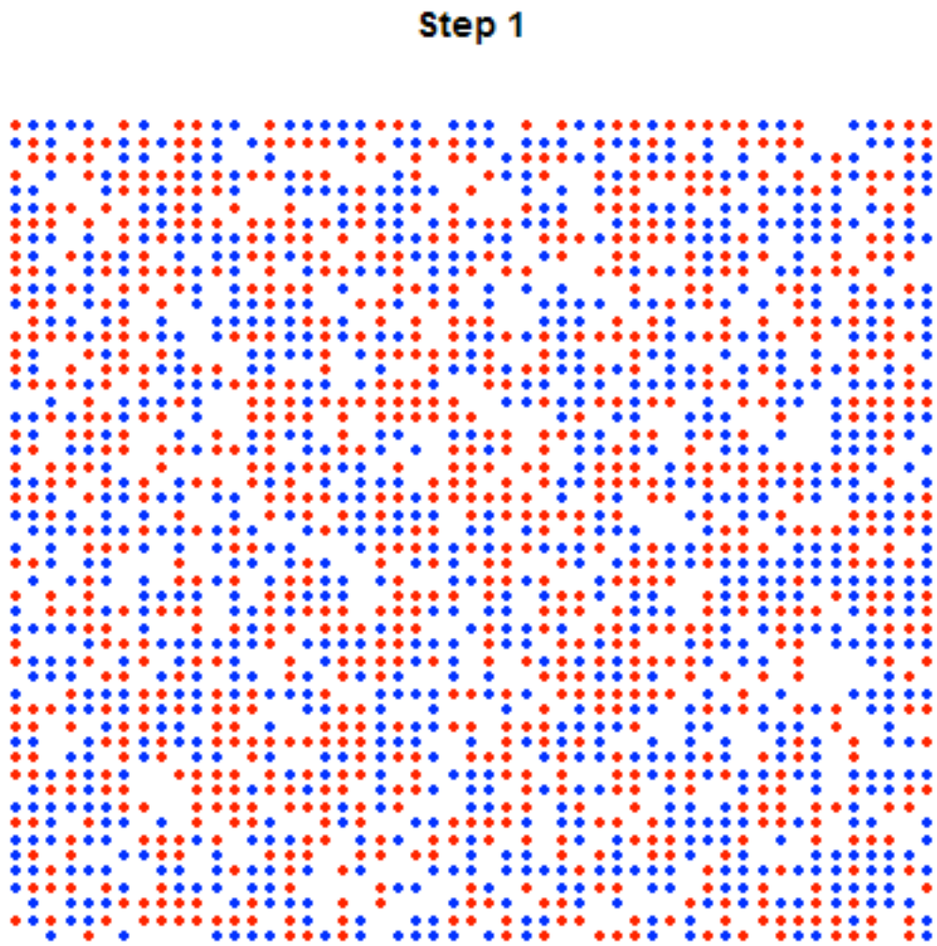


Example: Schelling model

- Thomas Schelling developed to understand neighborhood segregation
- Basic rules
  - Two kinds of agents (red/blue), initially placed randomly
  - Each wants at least X% of neighbors to be the same type as them
  - If not, move to a new location

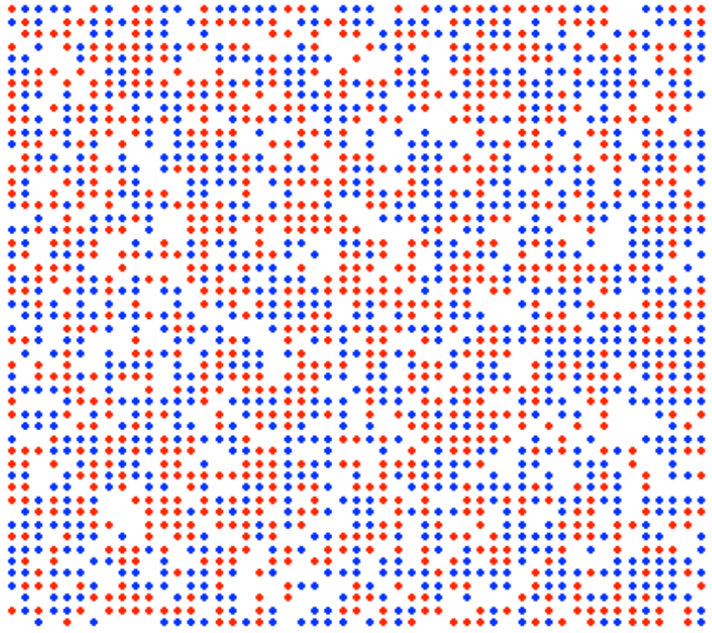


Example: Schelling Model

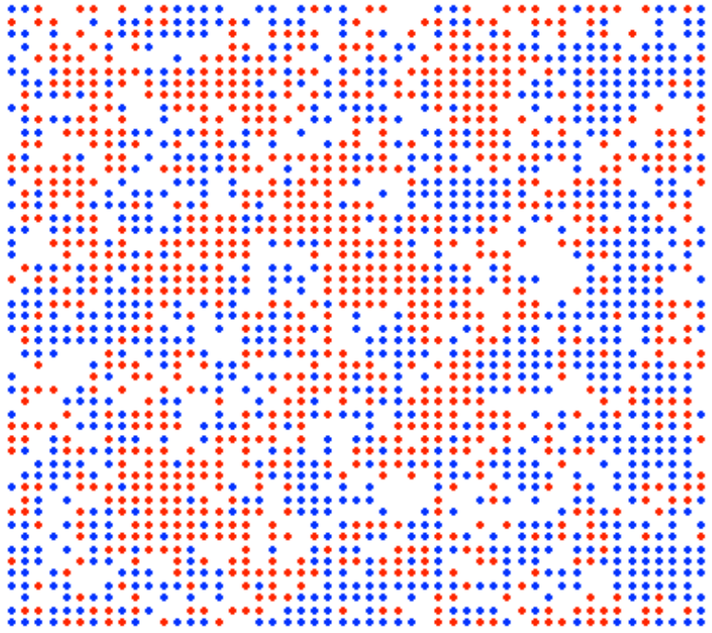


Example: Schelling Model

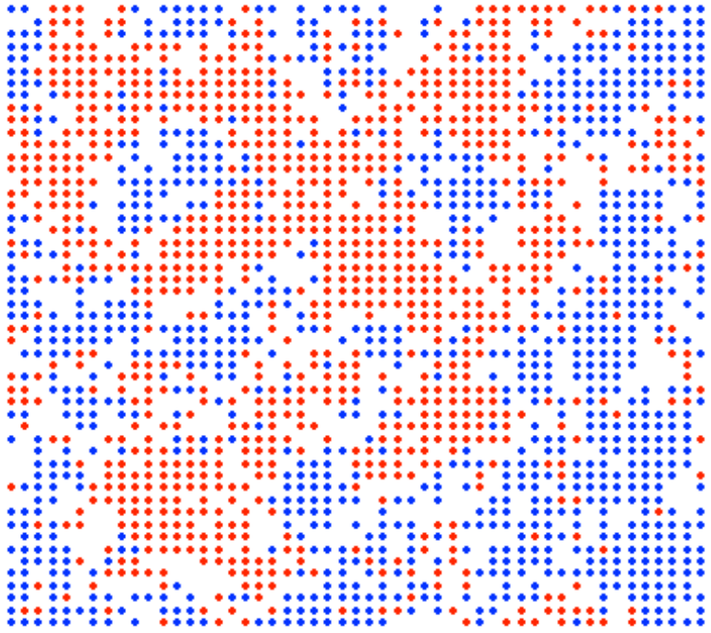
Step 1



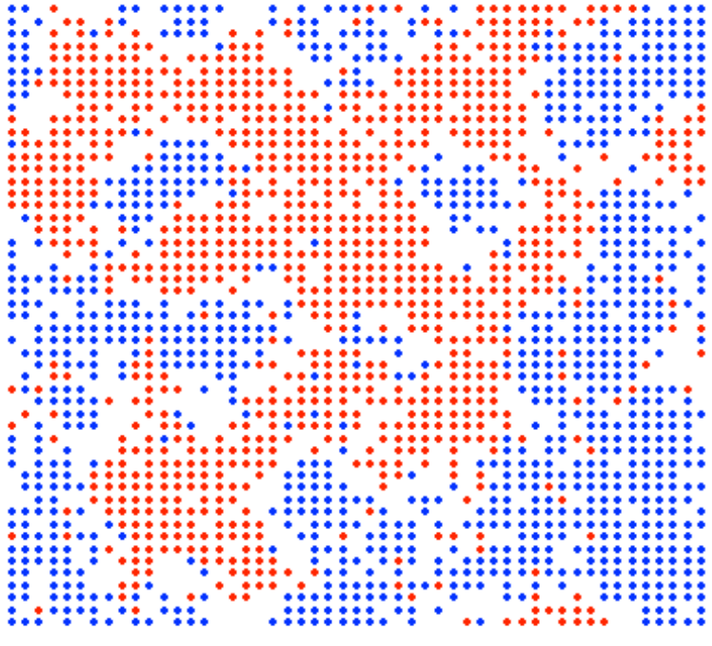
Step 3



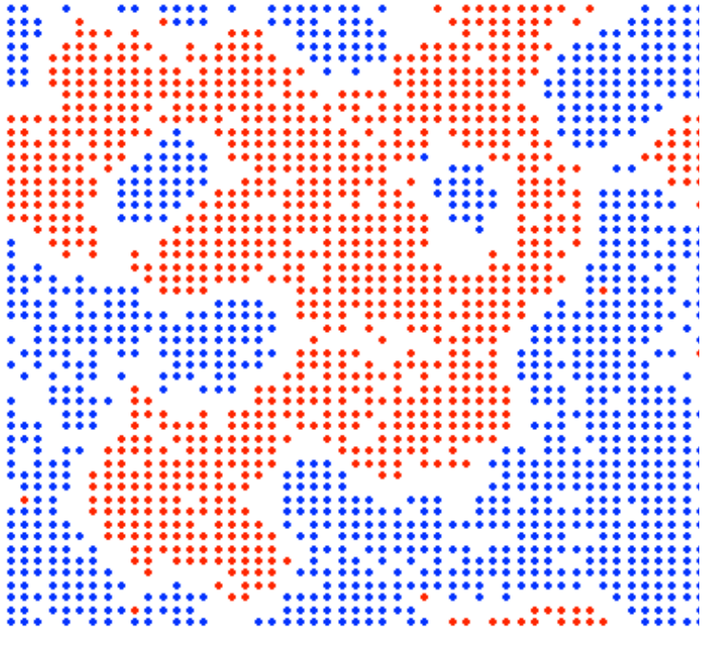
Step 5



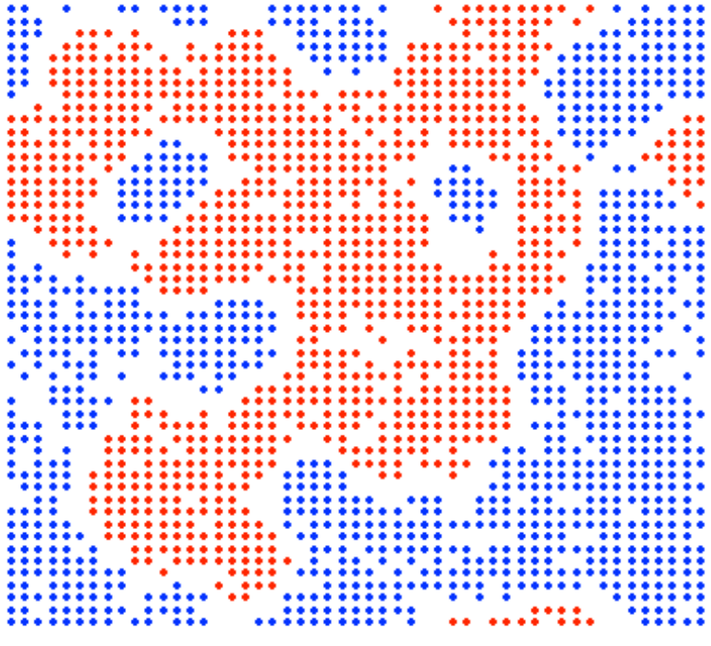
Step 7



Step 14



Step 26



Example: Schelling Model

- What processes need to happen?
  - Each individual needs to evaluate its neighborhood, decide if happy
  - Each individual needs to move if unhappy

Example: Schelling Model

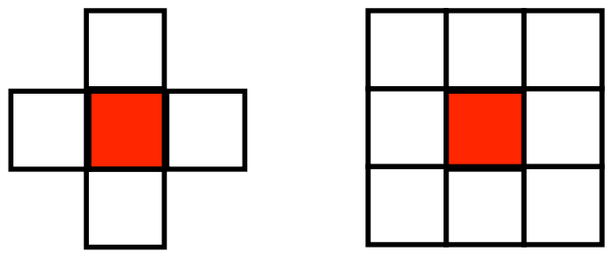
- **Agents**: people
  - **Properties**
    - Red or Blue
    - Fraction of neighbors they want to be same
    - Location
    - Happiness Level

Example: Schelling Model

- **Actions**: move or stay
- **Rules**
  - How to decide to move/stay?
  - What order to evaluate agent movements?
- **Time**: usually discrete steps
  - Continuous time?
- **Environment**: grid (size? boundaries?)

Example: Schelling Model

- What order to do rules in?
- Where to move them if unhappy?
- Count diagonal neighbors? What about grid edges?



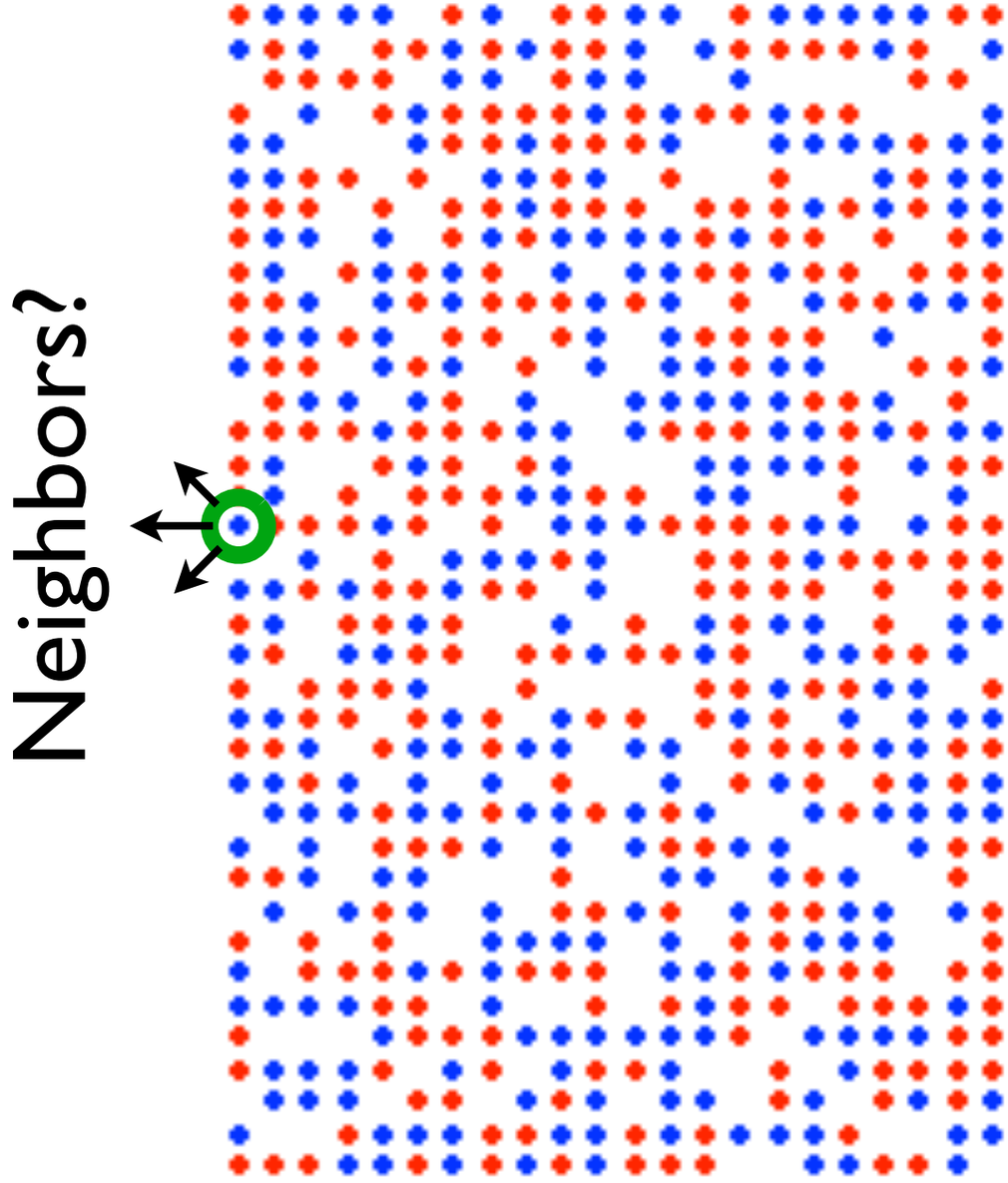
Example: Schelling Model

- For each individual (choose randomly)
  - 1) Evaluate happiness
  - 2) Relocate if needed

- 1) For each individual evaluate happiness
- 2) Randomly relocate individuals as needed

Example: Schelling Model

- Environment: gridded
- Size?
- Borders? What to do about neighbors on edges - should it wrap or..?



Example: Schelling Model

- These choices can affect the outcome!
- Need to think carefully about assumptions and potentially do a lot of sensitivity analyses to determine if the observed behavior is robust to these different decisions about how to build the model
- We will explore this more next time!

Common Issues with ABMs (and models in general)

- May seem easy to set up
- Allows for a lot of details to be added
- Frequently used by beginning modelers for these reasons
- But can be very subtle!
- Details of implementation can greatly change the behavior

- "You should assume that, no matter how carefully you have designed and built your simulation, it will contain bugs (code that does something different to what you wanted and expected)." (Gilbert 2007)

- "Achieving internal validity is harder than it might seem. The problem is knowing whether an unexpected result is a reflection of a mistake in the programming, or a surprising consequence of the model itself. … Confirming that the model was correctly programmed was substantially more work than programming the model in the first place." (Axelrod 1997)

Common Issues with ABMs (and models in general)

- Complexity of implementing rules
  - Rule order for the SIR model - Recover vs transmit first?
- Numerical issues (e.g. grid issues, floating point arithmetic)
- Environment issues - e.g. boundary choices (wrap boundary, closed edge, or..?)

Common Issues with ABMs (and models in general)

- ‘Emergent’ behavior can be a strength & weakness
  - Interaction of simple rules can lead to complex, organized behavior
  - But this extends to often arbitrary rules of how you set the model up, e.g. how you make the grid for the model, or order in which you implement rules

Common Issues with ABMs (and models in general)

- Importance of documenting, verifying, and validating code

- Replicability of ABMs and models in general is key to testing these issues—allows authors and others to change assumptions, underlying setup, etc.

- ODD (Overview, Design concepts, Details) Protocol - https://www.comses.net/resources/standards/

Parameters & Sensitivity Analysis

- Sensitivity analysis essential
- Parameter values are almost certainly wrong
- Sensitivity of behavior to parameters can mean predictions or results are less certain
- Simple approach - often know a reasonable range for parameters

For next time…

- Read
  - “Why Model?”, Epstein 2008
  - Sayama, Chp. 1-2
  - “More is Different,” Anderson 1987
  - Wilensky, Chp. 0-1
  - **PARTE framework reading by Ross Hammond (on website)**
- Do
  - Download and Install NetLogo and Python (e.g. Anaconda)