Agent-Based Modeling in NetLogo

Conor Mayo-Wilson

University of Washington

Agent-Based Modeling in Philosophy
December 10th, 2014

Today's Tutorials

- **Morning:** Stag Hunt
- **Afternoon:** Stag Hunt on networks
- **Lyon:** ABMs in Python

Outline

1. Why the Stag Hunt?
2. ABMs of Stag Hunt
3. NetLogo
4. References

Stag Hunt

	Stag	Hare
Stag	$\langle 4,4\rangle$	$\langle 0,3\rangle$
Hare	$\langle 3,0\rangle$	$\langle 3,3\rangle$

Importance of Stag Hunt

The Stag Hunt is a popular and influential game in studying cooperation [Skyrms, 2004].

Why not the Prisoner's dilemma?

Prisoner's Dilemma

- Suppose you and a colleague discover a result simultaneously.
- You make a contract to co-author the paper and not claim priority.

	Uphold	Break
Uphold	$\langle 4, 4\rangle$	$\langle 0, 5\rangle$
Break	$\langle 5, 0\rangle$	$\langle 3, 3\rangle$

Solutions to Prisoner's Dilemma

Solutions to Prisoner's Dilemma:

- Punishment
- Altruism
- Correlated play
- Restrict strategy set (e.g., Tit-fot-tat vs. Always Defect) in repeated prisoner's dilemma.
- And more . . .

Claim: Many solutions to the Prisoner's dilemma create stag hunts.

Prisoner's Dilemma

	Uphold	Break Contract
Uphold	⟨4, 4⟩	⟨0, 5⟩
Break	⟨5, 0⟩	⟨3, 3⟩

Suppose an external body punishes those who break contracts (e.g., by inflicting a penalty of 2).

Prisoner's Dilemma

	Uphold	Break Contract
Uphold	⟨4, 4⟩	⟨0, 3⟩
Break	⟨3, 0⟩	⟨3, 3⟩

Suppose an external body punishes those who break contracts (e.g., by inflicting a penalty of 2).

The game is now a stag hunt.

Stag Hunt

	Stag	Hare
Stag	⟨4, 4⟩	⟨0, 3⟩
Hare	⟨3, 0⟩	⟨3, 3⟩

Two Nash Equilibria:

- Both play Stag, or
- Both play Hare.

⇒Equilibrium selection problem.

Outline

1. Why the Stag Hunt?
2. ABMs of Stag Hunt
3. NetLogo
4. References

ABMs vs. Classical Economic Models

Classic Models

- Rational, EU Maximizers
- Homogeneous agents
- Global Interaction
- Equilibria

ABMs

- Boundedly Rational
- Heterogenous Agents
- Local interactions in a network
- Dynamics

And many more . . .

Getting Started

- Download NetLogo: `https://ccl.northwestern.edu/netlogo/5.1.0/`
- Download Code: `http://mayowilson.org/Teaching/ABMP_Tutorial.htm`

Basics

Basics

- Creating agents (turtles, patches, and links)
- Changing built-in attributes
- Creating agent attributes (turtles-own, patches-own, links-own)

Sample Code Explained

- Open the file "Stag_Hunt_Participants.nlogo"
- Play with inputs a bit.
- Then I will explain the code.
  - More details are in the NetLogo Dictionary: `http://ccl.northwestern.edu/netlogo/docs/dictionary.html`

Creating Inputs

Task 1: Create inputs

1. Deleted the declaration of the global variables stag-stag-payoff, stag-hare-payoff, etc. in the code.
2. In the same way I created a slider, create four "Input" fields. Enter "stag-stag-payoff", "stag-hare-payoff" etc. in the global variables field, and choose "Number" as type.
3. These four Inputs will allow the user to enter the payoffs in the game.

Creating Plots

Task 2: Create plots

- In the same way I created a plot that displayed that average payoffs of stag and hare hunters at each stage, create a plot that indicates the number of stag hunters and hare hunters on each stage.

Alter the learning algorithm

Task 3: Change the learning algorithm

- Write a new procedure called "Best-respond" that does the following:
  - After an agent plays a Stag Hunt with her neighbors, she calculates how much they would have earned by playing the other strategy.
  - If that number is strictly greater, she switches strategies.
  - If that number is strictly smaller, she keeps her current strategy.
  - If it is the same, she randomly picks a strategy.
- Create a chooser that declares a global variable "Learning-Algorithm."
  - Make the choices "Imitate-the-best" and "Best-response."
- Alter the play procedure so that it runs the learning algorithm that is chosen by the user.

Reinforcement Learning

Task 4: Change the learning algorithm again

- Create a turtles-own variable called "old-payoffs."
- Initialize old-payoffs to the list [100, 100] in the setup procedure.
- Write a new procedure called "Reinforcement-learning" that does the following.
  - On each stage, after play is called, each stag player adds her payoff to the first item of old-payoffs; each hare player to the second.
  - She then adopts the strategy stag with probability $1 - \frac{\text{First item of old payoffs}}{\text{Sum of old-payoffs}}$.

References I

Skyrms, B. (2004). *The stag hunt and the evolution of social structure.* Cambridge University Press.