Teaching and Educational Methods

Using NetLogo to Build an Agent-Based Model for Teaching Purposes at the Graduate Student Level

Bryan Collins[a] and Chyi-Lyi (Kathleen) Liang[b]
[a]*State University of New York at Oneonta,* [b]*North Carolina Agricultural and Technical State University*

JEL Codes: C0, D0, E0
Keywords: Agent-based model, NetLogo, system analysis

Abstract
Scholars and educators in agricultural economics face changing paradigms moving toward system-wide studies. Complex issues often involve quantitative and qualitative approaches, and it is difficult to access or acquire user-friendly tools that integrate both approaches. Agent-based modeling (ABM) offers a unique supplement to more conventional system-wide modeling frameworks, such as supply chain models, circular economy models, or coupled human and natural system models. The purpose of this paper is to show educators and graduate students about how agent-based models can be used in a graduate program curriculum. The paper shares some insight about the concept and sample applications of ABM, a popular analytic tool to study system-behavior-decision consequences through the interactions of entities. We use an example of simulating a buyer-grower market interaction for poultry products to demonstrate step-by-step strategies of using the NetLogo program to create an agent-based model. The benefits of using agent-based models include flexibilities of generating micro-level assumptions to approximate macro-level activities and outcomes, and the comprehensive integration between quantitative and qualitative analyses. The challenges are at the beginning phase to comprehend the scope and scale of analysis, define proper agents and behavioral characteristics, and generate meaningful interactions among agents in a logical manner.

1 Introduction to Agent-Based Models

The field of agricultural economics has expanded significantly to involve complex factors while examining various systems such as food systems (e.g., Fresco et al. 2021), health systems (e.g., Norton, Alwang, and Masters 2021), environmental systems (e.g., Fezzi and Batemen 2011), and many others. Several analytical tools have been developed to accommodate the needs to incorporate interactions, feedback, decisions, circularity, and consequences across actors within a system or linking multiple systems (Velasco-Munoz et al. 2021; Monti et al. 2023). An agent-based modeling (ABM) system is a computational process for researchers to simulate interactions and dynamics between diverse sets of agents. ABM is generally synonymous with multiagent-based modeling, multi-agent systems modeling, and agent-directed modeling (e.g., Oren et al. 2000). Simulations that take an ABM approach have gained popularity in the social sciences, due to its wide variety of applications that allow for an agent's decision making to be represented in ways that are context-dependent and easy to manipulate (e.g., Gilbert 2019).

The characteristics of an ABM are fundamentally different from other popular modeling structures, such as variable-based approaches, given the nature of using micro-level assumptions to approximate macro-level outcomes (Van Dyke Parunak et al. 1998). Several modeling paradigms exist among ABM specifications, requiring the researcher to choose which option would be the best for specific research purposes. Selecting a paradigm may depend on how the modeling system characterizes time (continuous or discrete), scale (macroscale or microscale), and the desired purpose of study (outcome-oriented or process-oriented). Given the unique strengths of ABM, it would be valuable to provide an overview of this tool for educators and graduate students in order to acquire some information/applications about ABM and how to incorporate them into graduate curricula. In this paper, we will explain the fundamental aspects of ABM, discuss the value in integrating ABM learning into graduate programming, introduce NetLogo (an ABM program) as an example for graduate-level teaching purposes, and outline a demonstration of how to apply NetLogo to simulate producer-buyer market interactions within an ABM concept. Finally, some challenges of this approach are discussed.

2 Background. What Is an ABM?

There are *three primary elements* that compose a typical ABM: (1) the selected agents, (2) the specifications and rules of how those agents interact with each other and with their shared environment, and (3) the specifications of the environment itself (Klügl and Bazzan 2012). Stemming from the world of artificial intelligence (AI) and the belief-desire-intention (BDI) framework, *an agent* is an entity that has its own set of beliefs about itself and its environment, its own set of desires that it wants to preserve, and its own set of intentions for what it is trying to accomplish (Singh, Padgham, and Logan 2016; Abar et al. 2017). The greatest strength of ABM is that it allows the researcher to focus on interactions and behaviors of the agents in the system, rather than solely focusing on overall global outcomes within the environment. This bottom-up, rather than top-down, nature is the ABM system's defining feature (Epstein 2012).

While acknowledging the difference between ABM and other modeling systems, ABM typically is most appropriate for analyzing systems with a specific set of characteristics (Hare and Deadman 2004; Klügl and Bazzan 2012). Some key characteristics are as follows: (1) systems that integrate intelligent human behavior, (2) systems that focus on the interactions between individuals and populations, (3) systems that incorporate evolutionary or changing dynamics among the selected agents, (4) systems that have multiple discrete levels where the researcher may want to analyze how each level interrelates, and (5) systems that need heterogeneous rather than homogenous behavioral rules, which is assumed in many model systems. For example, general equilibrium-based models often use homogenous representation and rely on strong, and perhaps unrealistic, assumptions about the uniformity of geographic space, rational decision making and human behavior, and presence of perfect information.

2.1 Impact of ABM on Graduate Student Development

We propose ABM as an excellent teaching support tool for graduate education in fields both in the natural and social sciences. Given the requirement for users to have at least foundational experience and skill in computer science and programming, we recommend introducing ABM materials at the graduate level, despite some claims of benefit to introducing ABM at the undergraduate level (Shiflet and Shiflet 2014). We see value in graduate students being exposed to ABM techniques in the classroom for two primary reasons. First, it will provide students with a domain to engage with interactive materials through active learning. Second, exposure to ABM can act as a starting point for graduate students to enter the world of graduate-level research by using ABM techniques as a focal point in their research projects.

Axtell and Farmer (2022) describe how graduate training in economics and finance typically relies on traditional learning pathways via reading text, whereas students often have different learning styles. Introducing activities in the classroom that make use of ABM programs offers a new learning style for instructors to convey economic ideas in a more illustrative context, while allowing students to interact with the model and examine ideas, such as through "what if" scenario analysis. Agent-based economies, even at a microscale, would offer students chances to tweak parameters and test hypotheses to see how actual economies function. Specific topics, such as supply and demand, can be visualized in real time as the model runs. Several other economic models do not offer the ease-of-use or versality that would be required for implementation in the classroom (Axtell and Farmer 2022). Computable general equilibrium models are more complex and have been critiqued for being “black boxes,” which is not conducive to be presented in a learning environment (Devarajan and Robinson 2002). Dynamic stochastic general equilibrium models often are not accessible to new users due to the steep learning curve for using them (Junior and Garcia-Cintado 2018). The functionality of ABM, paired with its emphasis on visuals, lets instructors show theories in practice in a novel and intuitive way compared to more traditional methods of knowledge transfer.

Exposure to ABM in the classroom can also offer a bridge toward helping graduate students develop their research projects. Modeling and simulation have gained legitimacy as an established research basis in graduate education, even for students with a limited background in quantitative analysis (Mielke et al. 2009; Carsey and Harden 2015). Murphy et al. (2020) praised ABM for graduate research for two reasons. First, ABM development teaches transferable skills, including programming, experimental design, and data management. Second, it provides research autonomy to the student, giving flexibility in designing and fulfilling their research project. In addition to exposure in the classroom, we see supervisors as having a critical role at teaching their students about ABM as a potential focal point of graduate research, given supervisors significant capacity in assisting in their student’s research aims and success (Platow 2012).

2.2 Application Areas of ABM

The first study to directly account for individuals in a simulation was conceived in 1969 when economist Thomas Schelling analyzed ways in which neighborhoods can naturally segregate themselves by assigning characteristics to each household (which are acting as individuals). By assigning rules that households tend to want to be near other households with similar characteristics, the simulation revealed a mechanism for how a phenomenon such as segregation may occur in the natural world. Since Schelling created the first ABM in 1969, the world of ABM has expanded in scope and scale. The valuable properties of taking an agent-based approach started to gain notoriety near the turn of the century, when it was deemed to be a “revolutionary development” within the social sciences (Bankes 2002).

Several subfields within the social sciences, such as economics, political science, epidemiology, and sustainability science, have applied ABM. Economists have noted that conventional models are excellent aids for assessing the economy during times of stability. In these circumstances, equilibriums are normal, and assumptions about human behavior and rationality are usually accurate. However, in times of economic crashes or crises, there is no longer an equilibrium for a model to be based off of, and human behavior and rationality fluctuate. In this case, bottom-up approaches, such as ABM, are more suited to handle this level of complexity than conventional top-down models by being able to capture irrational or non-normal systems behavior (Tesfatsion and Judd 2006).

ABM within the domain of agricultural economics and agricultural policy began to proliferate after 2008 when seminal publications in the field gained distinction (Kremmydas, Athanasiadis, and Rozakis 2018). Two early adopters of ABM in agricultural economics were Balmann (1997), who used a cellular automata approach to observe how farms competed for land and capital in a regional geospatial environment, and Berger (2001), who measured the interactions between farms, the local economy, and regional hydrologic processes to understand resource use changes for irrigation practices and what this means for policy. These papers were innovative for two reasons. First, they explicitly modeled farm’s interactions to explore structural change, such as optimal directions for a farm to scale up or down, or to enter or exit the business. Second, they added a spatial dimension, which was uncommon in traditional modeling techniques at the time.

Brady et al. (2009) used a spatial ABM approach to understand the effects of reforms to the European Union’s (EU) Common Agricultural Policy in 2003, which altered how farmers received payments and benefited from agricultural support structures. Freeman, Nolan, and Schoney (2009) expanded upon Berger’s use of ABM to explore agricultural structural change by simulating agriculture on a much larger scale over the course of 1960–2000. Happe et al. (2009) re-examined the reforms of the Common Agricultural Policy by creating a model to analyze how changing payment structures can influence succession rates and exit strategies among single-holder farm operators. ABM has been increasingly applied in agricultural studies due to the ability to perform dynamic comparative analysis compared to static equilibrium farm models. This ability allows for modeling of dynamic variables, such as farmer and consumer behavior, changing markets, and resource availability. It also allows for bridging social and environmental elements (Kremmydas et al. 2018). While many more examples of ABM use in agricultural economics exist, we have chosen to highlight a few of the most impactful examples. For a more exhaustive list, see Chapter 86, Section 5.4 in the *Handbook of Agricultural Economics* (Liang and Plakias 2022).

Within political science, knowledge and application of ABM has grown rapidly because the desire to study complex phenomena has increased (de Marchi and Page 2014). Muis (2010) for example, used ABM to research political party stability and change in the Netherlands, simulating how political party popularity fluctuated due to media consumption among the voter base and competition between parties. While political party competition was a dynamic system and very rarely stable, the author concluded that the results from the ABM were comparable to public opinion polls, highlighting their potential as tools to validate or predict political system events.

In epidemiology, scholars have adopted ABM to model patterns of disease outbreak because outbreaks are complex phenomena and are difficult to predict (Miksch et al. 2019). The COVID-19 pandemic, for example, relied heavily on models to forecast trends and assist with resource management, both of which were crucial for policymakers and health agencies. The COVID-19 Agent-Based Simulator (Covasim) was created to assist policymakers and health agencies to effectively manage the crisis, and it was quickly put to use globally (Kerr et al. 2021). An agent-based approach was selected over other modeling systems to best capture the microscale complexities that are necessary for proper mitigation of the pandemic. Being able to run scenarios that test out different policy responses was noted as being immensely valuable. Several other studies introduced similar scoping models to simulate the COVID-19 pandemic (Cuevas 2020; Silva et al. 2020; Shamil et al. 2021).

Within the emerging field of sustainability science, ABM is seen as a great tool for stakeholders and policymakers to improve system sustainability by exploring and testing the effects of different scenarios. Several studies within the last decade have used ABM to discover how proposed interventions could affect food and agricultural systems related to food security, or the interactions between biophysical/climate conditions, supply chain, transportation systems, and spill-over effects from agricultural output (Liang and Plakias 2022).

3 Methods. Choosing the Appropriate ABM Toolkit

Several toolkits exist to design and run ABM. Abar et al. (2017) created an extensive list of 85 different toolkits to create ABM, where each toolkit was briefly described based on its unique properties. Due to the vast number of toolkits available, selecting an appropriate toolkit is essential and often one of the first steps in designing an ABM. ABM toolkits are quite diverse. They range in several functionalities such as: the source code, coding language, type of interfaces available, preferred operating system (OS), necessary experience with coding (novice, intermediate, or expert), modeling strength and capacity for complexity, capacity for 2D or 3D visualization, and typical domain of applications. Additionally, not all ABM toolkits are easily accessible. While many programs are open source and free to download, many need a license and are proprietary.

Five of the most commonly used ABM toolkits are NetLogo, AnyLogic, Repast, MASON, and Swarm. For this project, NetLogo (an open source package) was selected as the ABM toolkit because of its (1) relative ease of use, (2) strong source of educational tools, documentation, and tutorials, (3) free and open source availability, and (4) ability to run on all platforms and operating systems. Developed by Uri Wilensky in 1999, NetLogo is both an agent-based programming language and modeling environment. The program provides modelers with the tools to "give instructions to hundreds or thousands of "agents" all operating independently. This makes it possible to explore the connection between the micro-level behavior of individuals and the macro-level patterns that emerge from their interaction (Wilenski 1999). It was designed to model the complexities of both natural and social systems, and was built in mind for educational and research purposes (Tisue and Wilensky 2004). The creators of the toolkit hoped to create a modeling program that would be capable of running complex simulations, but should be simple enough for students and non-programmers to make use of and create their own sophisticated models. Several computer science and modeling classes have been taught at universities that use NetLogo. The creators have documented that NetLogo has been downloaded tens of thousands of times, and find the discussion group online to be active with group members sharing ideas and advice (Wilenski 1999).

3.1 What Is NetLogo?

NetLogo has three interfaces for the user to interact with. The *first interface* is the programming environment to write and edit NetLogo code. The *second interface* is the visualization environment, where the user can see how agents are interacting and see how relevant metrics and statistics change as the model runs. Also, in this interface, the user can manually toggle model parameters through means of sliders and input buttons (features of the programming) that link back to the NetLogo code. The *third interface* is the documentation environment, where users are encouraged to document the background information about the model as well as step-by-step information for running the model correctly. In NetLogo, the agents can be used to represent any conceivable entity, such as airline passengers, cars, solar panels, molecules, buyers, sellers, batteries, etc. The environment that the agents inhabit is referred to as *a network of patches*. Patches can be used to represent any environmental entity, such as farmland, a street network, a grocery store, a microscopic cell, etc. See Figure 2 for labels of each of these interfaces within the NetLogo environment.

3.2 Applications of NetLogo

NetLogo has been a key research tool in many studies, ranging from applications in biology, logistics, economics, and sustainability. For example, it has been the selected toolkit for modeling green transportation potential in cities. Emergence of systems thinking in urban studies has integrated the food-energy-water nexus within urban agriculture networks, prompting a methodology to simulate how this integration is sensitive toward policy changes to enhance efforts toward green transportation. ABM was used to help understand this coupled system by locating food desserts and deficiencies within green transportation efforts (Elkamel et al. 2023). Gebrehiwot et al. (2022) applied the ABM approach to simulate factors influencing grower (farmer)–buyer (household) decisions while considering fresh food availability and farmland transitions to alleviate food desert challenges (Gebrehiwot et al. 2022). In these studies, the agents were individual farmers or households, green transport variables (energy capacity, mileage, and electric charger type), urban agriculture microgrid variables (technologies including solar panels and wind turbines), other urban grid facilities like stormwater treatment plants and public utilities, and farmland.

Putting these agents into an ABM structure and seeing the resulting dynamics when implementing scenarios commonly associated with public policies, such as increasing renewable energy or inflating food costs, revealed the underlying complexities of coupled systems in an urban agriculture content. Under some scenarios, household income increased and food security decreased, suggesting a relationship between the scenarios introduced and food security levels across the modeling environment. The authors found that these methods could be used to assess sustainability strategies by simulating anticipated effects of new policy or technology (Gebrehiwot et al. 2022). Because ABM relies on local data and local knowledge, results from ABM that are context specific are not always interpretable, scalable, or generalizable (Sun et al. 2016).

Kowalska-Pyzalska (2017) used NetLogo to study sustainable development by modeling consumer willingness to pay for green energy. Consumer agents are first assigned a variable of their valuation of renewable energy, called a reservation price, and then assigned a variable of their willingness to adopt renewable energy practices, called an adoption threshold. Agents with a greater reservation price than the adoption threshold are considered potential adopters of renewable energy due to their greater willingness to pay and change their behavior toward favoring adoption of renewable energy practices. Because some of the variables are dynamic and can be specified by the modeler as parameters within the model change, different scenarios were applied to see how the population can best be incentivized to determine those that result in the greatest adoption rates. The study found that external incentives, such as financial support for adoption, offered positive outcomes to increase consumers' support for renewable energy projects.

Delcea, Cotfas, and Paun (2018) explored several scenarios within NetLogo to model the most efficient strategies to increase turnaround time among commercial aircraft by seeking different methods of loading and unloading passengers. Twenty-four different boarding patterns, such as random open seating, seat assignment by group, and seat assignment by seat, were all compared to see which pattern had the quickest flow and movement of agents. Agents were given variables related to speed, seat location, and how much luggage they carried on. For full flights, the model showed that the "by-row-back-to-front" method was the quickest, which could be helpful for airlines to adjust turnaround times and reduce conflicts.

NetLogo has been applied in studying cell biology, such as modeling of interactions between the sensory system of an organism and its surrounding environment. Dalle Nogare and Chitnis (2020) explored how the cells of a zebrafish could be influenced by its environment to modify its organ systems and biological development. NetLogo was chosen to study this phenomenon because of its visualization capabilities, as well as the relative difficulty to study such phenomena in a wet laboratory setting. This relative advantage also allows for study in biomedical research to understand complex biological systems and to investigate new hypotheses for research and development, such as the immune system (Chiacchio et al. 2014).

4. A Demonstration of Using NetLogo to Simulate Producer-Buyer Interactions within an ABM Concept

The following section elaborates how to design and create an ABM using NetLogo to analyze how *poultry producers and buyers* interact, including (1) ways to define various entities or agents, (2) ways agents interact among the patches, and (3) methods to explore complex themes such as positive and negative feedback loops, network dynamics, population dynamics, market dynamics, optimization, and self-organization. The rationale for choosing the poultry market is due to the nature of this project as a component of a large collaboration funded by the U.S. Department of Agriculture (USDA) Sustainable Agriculture Systems, focusing on an integrated system approach to examine the poultry industry from production to consumption, while taking account of environmental impacts.

To ensure the readers understand how we identify actors, factors, scenarios, and interactions in the examples described in the sections below, we need to set the stage by providing some background information about the poultry industry in the United States. The poultry industry in the United States has grown into a $77 billion industry (U.S. Department of Agriculture, National Agriculture Statistics Service 2022). Broilers made up 65 percent of the total poultry industry production, while eggs made up 25 percent and turkeys made up 10 percent. Consumption of chicken per capita has increased significantly from 2000 (76 pounds) to 2022 (99 pounds) partly due to health recommendations and cheaper prices when compared with red meat (U.S. Department of Agriculture, Economic Research

Service 2022). Improved efficiency of feeding and poultry supply chain mechanisms have boosted the prosperity of poultry operations over the years, as well (U.S. Department of Agriculture, Economic Research Service 2022).

Despite the prosperity of the U.S. poultry industry, there remains ongoing challenges. For example, Highly Pathogenic Avian Influenza (HPAI) has increased restrictions on production and exports, which threatens to raise prices and decrease demand in several markets (Chai et al. 2017; Choe 2023). Clostridium perfringens and salmonella enterica were identified as the most common pathogens, both of which are still impacting public health (Velasquez et al. 2018). Issues within the conventional poultry industry have spurred new methods toward poultry production and marketing. Supermarkets are increasingly promoting antibiotic-free and organic products due to higher demand. Organic production of broilers has continued to rise, with the value of organic broilers sold being 1.51 billion, a jump of 35 percent from 2020 (U.S. Department of Agriculture, National Agriculture Statistics Service 2023). On the consumer side, demand for organic, fresh, free-range, and antibiotic-free products is growing (Vander Mey 2004; Martínez Michel, Punter, and Wismer 2011; Van Loo et al. 2011). Increased demand leads to increased production of these products, allowing prices to be more affordable for the average consumer (Adamski, Kuzniacka, and Milczewska 2017; Schipmann-Schwarze and Hamm 2020).

4.1 Learning Model for ABM Applications

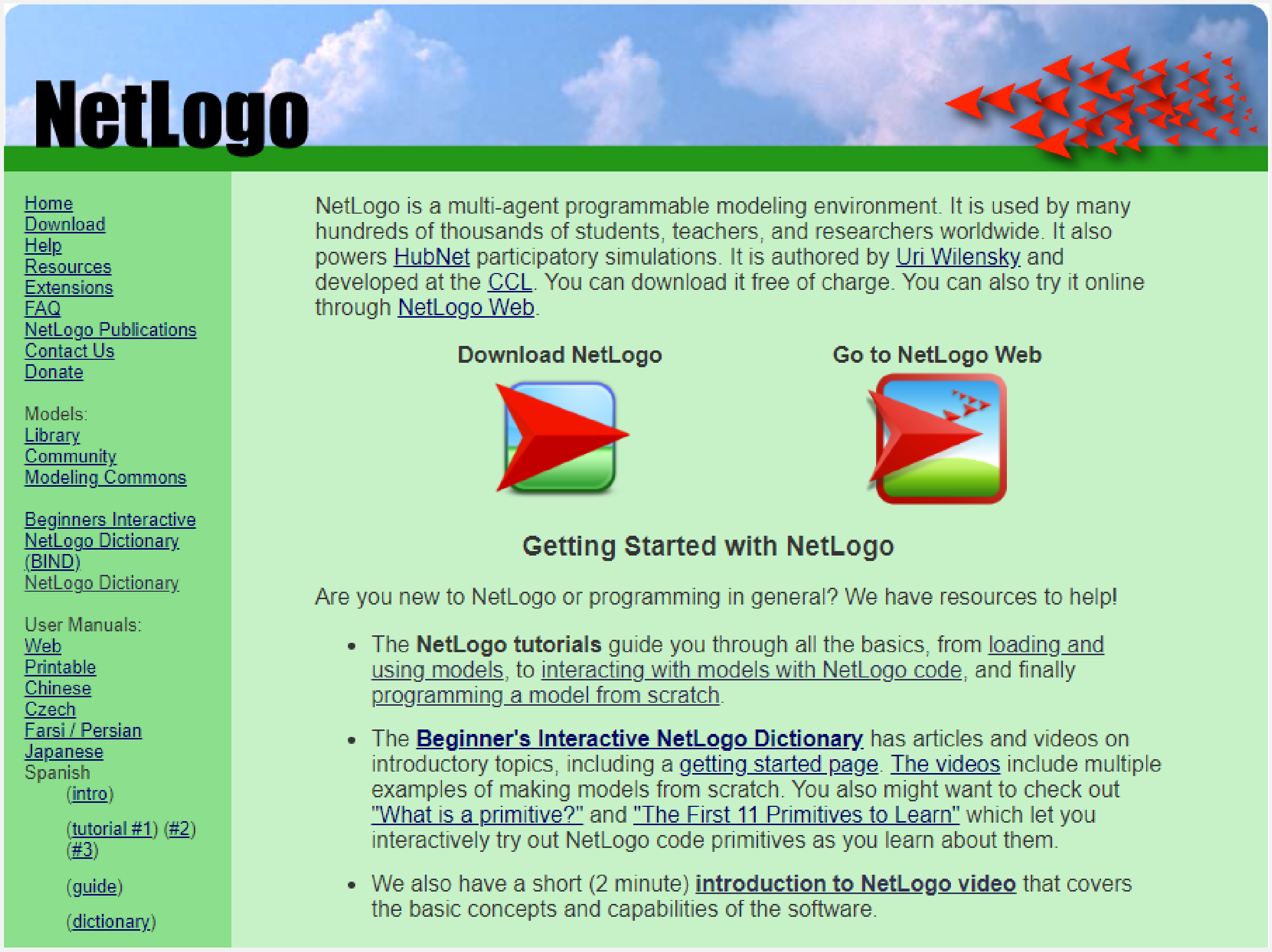
We will begin with a model with two different sets of agents—buyers and sellers—and explore how they interact in a poultry market. This is a simplified case study that proposes a methodology to use NetLogo for simulating market dynamics within an ABM analysis. *Buyer agents* are consumers who can make decisions whether to pay a premium for specific types of products, such as organic or pasture-raised chicken meat. *Seller agents* represent different scales of farms, such as a typical large farm that offers cheaper products and at greater quantities, or a typical small farm that offer products at a premium price with attributes that are appealing to a certain demographic of buyers.

4.2 Classroom Activity: Getting Started Learning NetLogo by Completing Tutorials

Upon selecting an appropriate ABM toolkit, the first step is to seek out which educational resources are available. For use in the classroom, we recommend priming students by directing them to the NetLogo website and having them complete the provided tutorials. The NetLogo website (https://ccl.northwestern.edu/netlogo/) features many resources to help a new user, including links to tutorials, dictionaries, and videos (Figure 1 below). The website provides clear instructions on how to download NetLogo, or if preferred, use the NetLogo Web application, which does not require any installation. Once downloaded, three key tutorials are worth exploring. *The first tutorial* introduces the user to the model's library, a set of several pre-made models that the user can experiment with, without having to create any code. This introduces how to control the parameters of the selected model by reviewing how different interface elements interact with the model, as well as actually running the model under different parameters.

The second tutorial focuses more on editing models as opposed to simply observing them. This is done through introducing commands and reviewing how commands can alter the model's properties. For example, the command center is where the user can change the color of an agent, such as changing the color of a house from red to blue. *The third tutorial* teaches the user how to start building a model from the beginning. The example given is how to build an ecosystem, where turtles roam around the environment and eat grass, represented by patches, which in turn provides the turtles with energy to further roam.

For further information, the programming guide and interface guide are both lengthy documents that describe the functions possible within the ABM toolkit. The *NetLogo Dictionary* provides definitions for each of the hundreds of primitives that exist as foundational language elements of the NetLogo



Home
Download
Help
Resources
Extensions
FAQ
NetLogo Publications
Contact Us
Donate

Models:
Library
Community
Modeling Commons

Beginners Interactive NetLogo Dictionary (BIND)
NetLogo Dictionary

User Manuals:
Web
Printable
Chinese
Czech
Farsi / Persian
Japanese
Spanish
(intro)
(tutorial #1) (#2) (#3)
(guide)
(dictionary)

NetLogo is a multi-agent programmable modeling environment. It is used by many hundreds of thousands of students, teachers, and researchers worldwide. It also powers HubNet participatory simulations. It is authored by Uri Wilensky and developed at the CCL. You can download it free of charge. You can also try it online through NetLogo Web.

Download NetLogo

Go to NetLogo Web

Getting Started with NetLogo

Are you new to NetLogo or programming in general? We have resources to help!

- The **NetLogo tutorials** guide you through all the basics, from loading and using models, to interacting with models with NetLogo code, and finally programming a model from scratch.
- The **Beginner's Interactive NetLogo Dictionary** has articles and videos on introductory topics, including a getting started page. The videos include multiple examples of making models from scratch. You also might want to check out "What is a primitive?" and "The First 11 Primitives to Learn" which let you interactively try out NetLogo code primitives as you learn about them.
- We also have a short (2 minute) **introduction to NetLogo video** that covers the basic concepts and capabilities of the software.

Figure 1: NetLogo webpage (https://ccl.northwestern.edu/netlogo/), which houses download instructions, as well as a host of learning resources and contact information.

programming language. All of these resources can be found on the ribbon on the left of the NetLogo website. If assistance is required when working through development of a model, the *NetLogo User's Group* (https://groups.google.com/g/netlogo-users?pli=1) is an active community where members may pose their questions and receive valuable feedback from other members.

4.3 Creating a Buyer-Seller Model for the Poultry Industry: Step by Step Instructions and Notes

In the poultry industry, there are several important actors. Within an ABM schema, each set of actors will be represented by a set of agents. A schema is the logical and organizational structure of a model. Create an initial schema for the project by considering all the agents necessary. This schema will be tied to the end goal of the project. In our case, it is to create a simulation of the poultry industry to test out different scenarios and find potential pathways for increased sustainability within the industry.

Step 1: Conceptualize Agents by Assigning Definitions and Assumptions

The following types of agents would be considered in a complete poultry industry model:

1. Farmers and farm workers.
2. Consumers—both direct consumers that buy meat to consume, but also intermediate consumers that purchase by-products of chicken for use in other things, such as animal feed or fertilizer.

3. Processors—butchers, inspectors, packaging, labeling, transportation to market, and other actors within the supply chain.
4. Agencies—USDA, Food and Drug Administration, Centers for Disease Control and Prevention, etc.
5. Chickens—How they interact with each other and maintain a healthy lifestyle.
6. Environment—Including variables of environmental health mostly at the farm level, but also including environmental impacts of waste and transportation.

For the purposes of this initial simplified model demonstration, only two agents will be selected—farmers and consumers.

Step 2: Identify Interactions Between Agents at the Most Fundamental Level

For initial model development, start simple and expand rather than introducing all possible agents into the model. First, pick two agents that interact at the most fundamental level. *Begin with one farmer (seller) and one consumer (buyer).* The simplest interaction is if the buyer decides to make a purchase from the seller. In this case, the decision a buyer makes is to either buy the seller's product or not. *At the least complex level, there are no clear assumptions that the modeler needs to make*. Writing the code to model this transaction requires adding relevant variables, such as assigning the seller an amount of price they are willing to sell their product for and assigning the buyer an amount of money they are willing to buy the same product for. In this case, a simple logical operator can establish the link between each agent's variable and determine if there is a successful purchase. If the buyer's amount they will spend is greater than the seller's amount they will sell for, then a purchase or transaction can be made. If the opposite is true, a purchase will not be made. Figure 2 below shows this simple dynamic, where one triangle is representing the buyer and the other represents the seller. The buyer is willing to spend $10 and the seller is willing to sell for $5; therefore a successful transaction occurs, represented by the yellow line between them. If the buyer is willing to spend $5 and the seller is willing to sell for $10, there will be no transaction and no yellow line will appear.

Step 3: Increase the Number of Buyers, Sellers, and Behaviors That Drive Market Exchanges

Two sellers and one buyer will be introduced in this step of the poultry market interaction model. The first seller, Seller A, represents a large commercial producer. Seller B represents a small local farm. Seller A and B can be distinguished by many characteristics, such as the number of birds they have sold, the total pounds they have sold, their gross earnings per year, their market channel and methods for selling products, their keywords and company mission, and their target demographics of consumers they sell to. In the earliest iterations of the model, the sole separating feature between the sellers is their selling price.

The model, shown in Figure 3 below, now has two sellers, which are represented by the blue and green bodies. The buyer, now represented by the orange smiley face, has the option of buying from either seller. At this second level of complexity in the model, the assumption is that the buyer will purchase only one product from the seller who is selling at the cheapest price if the seller's selling price is less than what the buyer is willing to pay. The yellow link between the buyer and the green seller indicates a successful purchase, while the blue link between the buyer and blue seller indicates that there is no purchase. As the model develops, there are now more interface elements and visual plots and windows. The turquois buttons in Figure 3 allows the user to change behaviors of the buyers and sellers, as well as determine the total amount of money the buyer has available to spend. The large plots on the right track the seller's prices they will sell at and the buyer's price they will pay, which will change after every tick. These will also update after every tick (A tick is the discrete time component of a NetLogo model). Each tick, in this case, represents a new

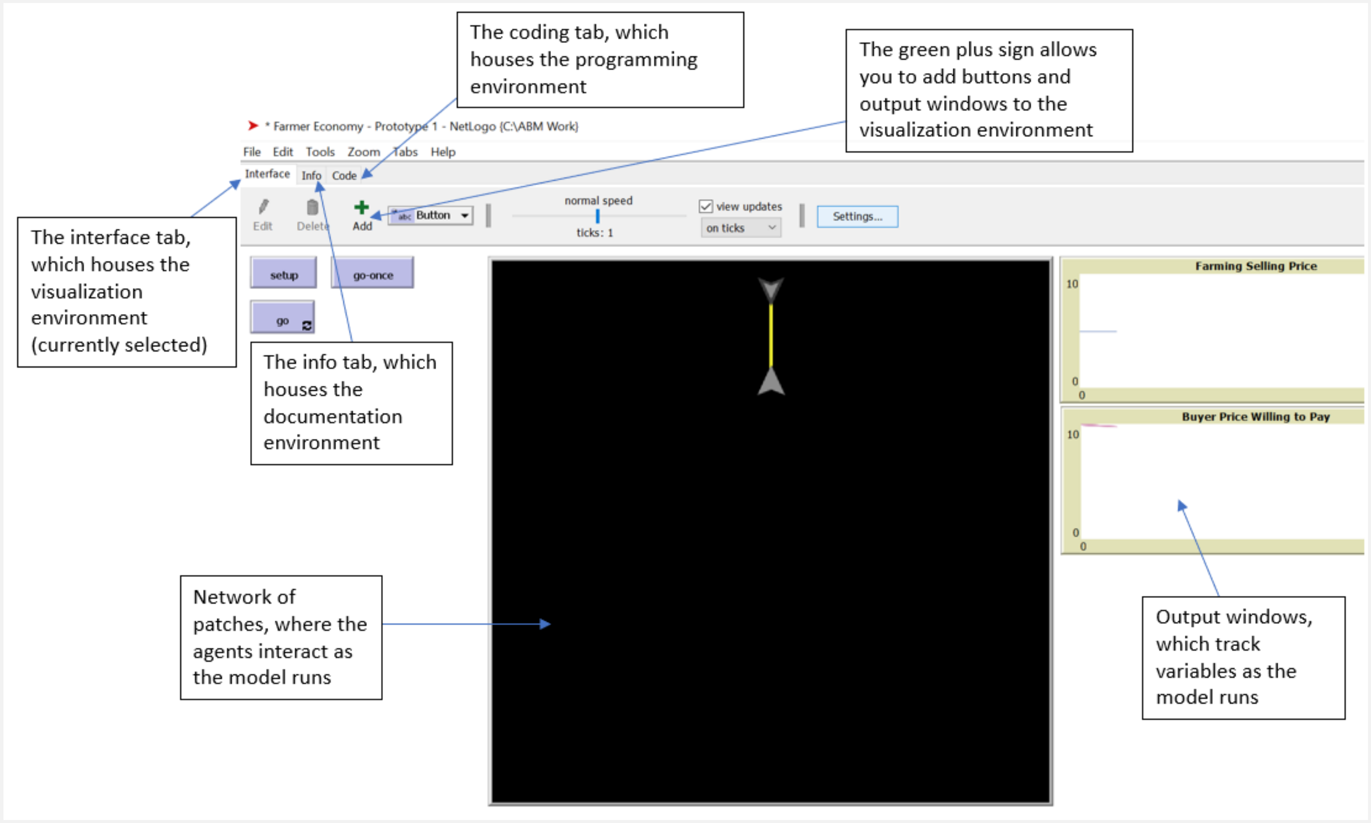
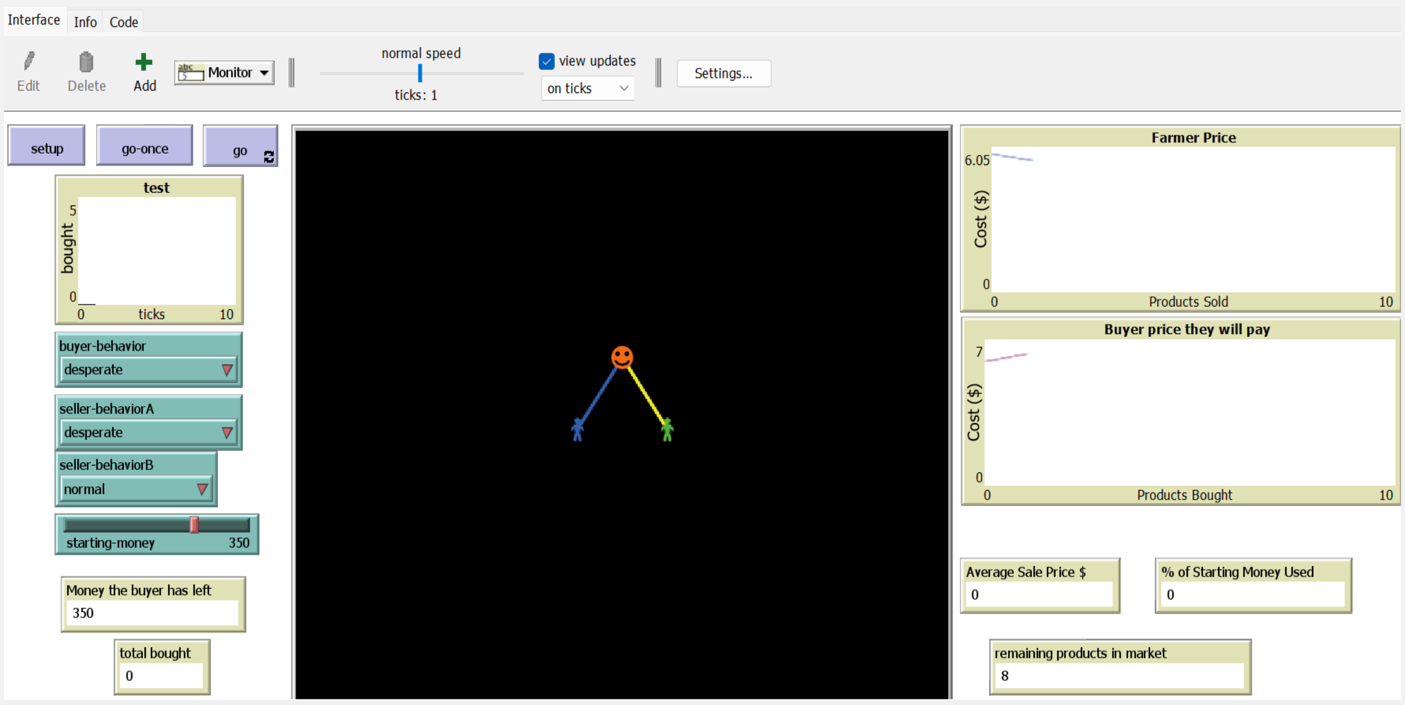

The coding tab, which houses the programming environment
The green plus sign allows you to add buttons and output windows to the visualization environment
* Farmer Economy - Prototype 1 - NetLogo {C:\ABM Work}
File Edit Tools Zoom Tabs Help
Interface Info Code
Edit Delete Add
Button
normal speed
ticks: 1
view updates
on ticks
Settings...
setup
go-once
go
The interface tab, which houses the visualization environment (currently selected)
The info tab, which houses the documentation environment
Farming Selling Price
Buyer Price Willing to Pay
Network of patches, where the agents interact as the model runs
Output windows, which track variables as the model runs

Figure 2: Screenshot of the model in development at its first level of complexity, showcasing the interactions between one buyer and one seller. Labels indicate several of the critical interfaces within the NetLogo environment.

opportunity for a buyer to interact with a seller and potentially make a purchase. In this scenario, a tick can be thought of as each time a buyer goes to the market to buy a product. So, the first tick can represent the first time a buyer goes to market, the second tick is the second time they go to the market, etc. Also included now are windows showing how much money the buyer has left, how many items have been bought, the average sale price, and number of products remaining.

All of NetLogo's buttons, sliders, monitors, and output windows, as seen in Figure 3, need to be manually added by clicking on the green "add" button in the top left corner of the program. Once the desired window is added, it needs to be specified and referenced in the code in order to know which variable the window is synchronized with. Each window can be edited by right-clicking and changing the settings. This is where the variable(s) will be selected and where other settings can be changed, such as the range of values for a sliders button, or the colors of a graph, or the graph's axis parameters.

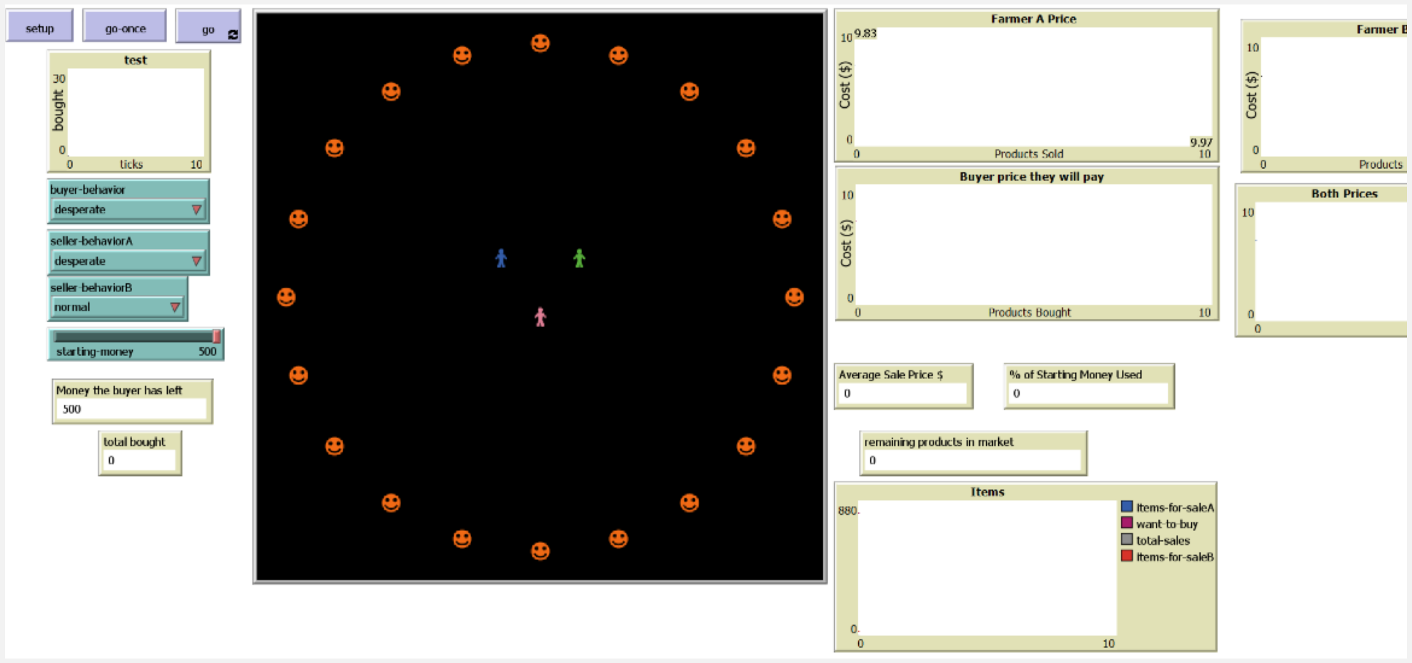
It is relatively easy to increase the number of buyers or sellers in NetLogo, as the user simply has to increase the population (such as in the code, changing from population of 1 to population of 10). The difficulty lies with assigning each new agent its own unique characteristics. If the user were to simply change the buyer population from 1 to 5, they would all be clones who have the same characteristics, such as having the same willingness to pay value, same starting budget, same attribute preferences, etc. Figure 4 shows the same model now with 3 sellers and 20 buyers. However, if all 20 buyers are clones, there is little value in running the model with 1 buyer or 20 buyers. The assumptions remain true at this level of complexity in the model, where buyers will only purchase from the cheapest seller, if the seller's products are below the buyer's price range.



buyer-behavior: desperate
seller-behaviorA: desperate
seller-behaviorB: normal
starting-money: 350
Money the buyer has left: 350
total bought: 0
Average Sale Price $: 0
% of Starting Money Used: 0
remaining products in market: 8

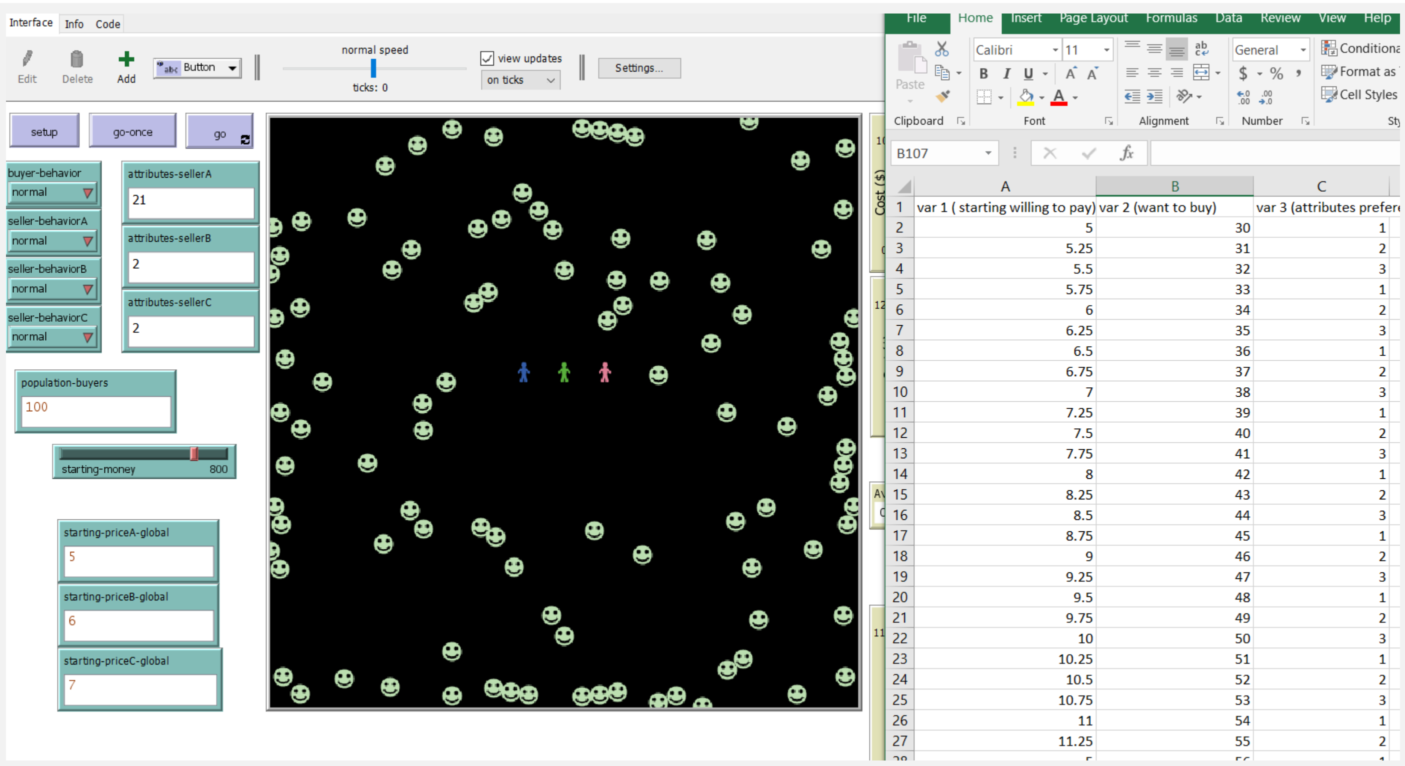
Figure 3: Screenshot of the model in development at its second level of complexity, where one buyer is now interacting with multiple sellers.

The solution to having buyers that are clones of each other is to make use of a CSV data file extension created in Microsoft Excel (Figure 5), which allows for the user to directly specify the traits of each buyer agent (willing to pay, amount they want to buy, preference for attributes, etc.). Each column



buyer-behavior: desperate
seller-behaviorA: desperate
seller behaviorB: normal
starting-money: 500
Money the buyer has left: 500
total bought: 0
Average Sale Price $: 0
% of Starting Money Used: 0
remaining products in market: 0

Figure 4: Screenshot of the model in development at its third level of complexity, where multiple buyers are interacting with multiple sellers. In this screenshot, all of the buyer agents are exactly the same.



buyer-behavior: normal
seller-behaviorA: normal
seller-behaviorB: normal
seller-behaviorC: normal
attributes-sellerA: 21
attributes-sellerB: 2
attributes-sellerC: 2
population-buyers: 100
starting-money: 800
starting-priceA-global: 5
starting-priceB-global: 6
starting-priceC-global: 7

	A	B	C
1	var 1 (starting willing to pay)	var 2 (want to buy)	var 3 (attributes prefer
2	5	30	1
3	5.25	31	2
4	5.5	32	3
5	5.75	33	1
6	6	34	2
7	6.25	35	3
8	6.5	36	1
9	6.75	37	2
10	7	38	3
11	7.25	39	1
12	7.5	40	2
13	7.75	41	3
14	8	42	1
15	8.25	43	2
16	8.5	44	3
17	8.75	45	1
18	9	46	2
19	9.25	47	3
20	9.5	48	1
21	9.75	49	2
22	10	50	3
23	10.25	51	1
24	10.5	52	2
25	10.75	53	3
26	11	54	1
27	11.25	55	2

Figure 5: Screenshot of the model in development at its third level of complexity, where multiple buyers are interacting with multiple sellers; however, unlike in Figure 4, all of the buyers are unique based on the attributes shown in the CSV data spreadsheet shown. Variables for each buyer include their willing to pay value, number of items they want to buy, and product attribute preference.

in the Excel table represents a trait. Each row represents one individual. Figure 5 shows the environment with 3 sellers and 100 buyer agents and the spreadsheet that holds their characteristics. The data is currently made up, but later on in the process, the data will be corrected to reflect real-world buyer characteristics.

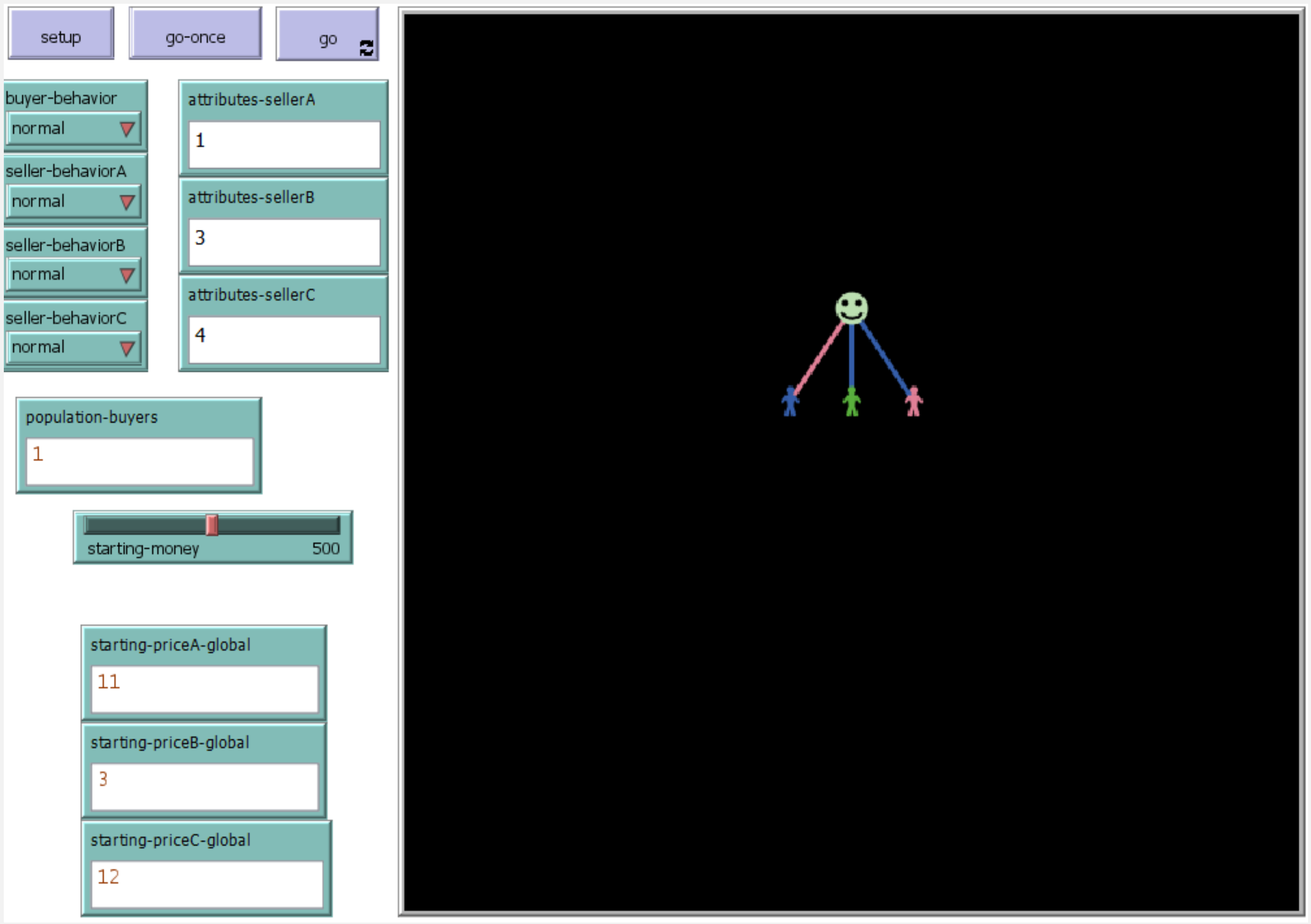
While a large percent of the population solely makes purchasing decisions based on price, several make their decisions on other factors such as targeting desirable attributes in the product they are seeking. Implementing levels to the buyer decision-making process increases realism. While not introduced in this example, other decision rules can be assigned to agents. One example is specifying agent behavior based on probabilities. Giving individual probabilities to agents can further mirror how irrational agents interact with their space. Within the context of this buyer-seller marketplace simulation, a simple example for probability-based behavior could be that 50 percent of the time a buyer goes to market, they prefer products based on their attribute, and 50 percent of the time they prefer products solely based on price. Bonabeau (2002) further discusses individual probabilities for use in simulating human systems.

4.4 Order of Operations

The order in which buyers engage with sellers is important to note when interpreting the results of the model. Consider when there is only one buyer, such as introduced in the initial steps of model development. Each tick represents one time the buyer goes to market, and each time they go to market, the variables are the same as when they had left the tick prior. However, when multiple sellers are engaging with multiple buyers, the order of operations becomes necessary to understand. For each tick, each buyer interacts with each seller to determine if a purchase will be made. Using the example in Figure 4, the first engagement starts with Buyer 1 deciding what to buy from Sellers A, B, or C. After this engagement, Buyer 2 repeats this process, followed by Buyer 3, etc., until all buyers have gone to market to engage with the sellers. Once this happens, the next tick commences and the process repeats. This means that the opportunities presented to each buyer are not completely equal. We will return to this point when we increase the complexity of the model to discuss what this order of operations conceptualization means for interpreting results.

4.5 How to Modify Preferences for the Agents

Figure 6 displays the difference in how buyer preferences can affect the market economy. If we have one buyer who prefers a specific attribute, say attribute "1" (e.g., organic, pasture-raised, etc.), and a farmer is selling a product with that attribute, the buyer will buy that product and disregard any of the other sellers' product, even if other sellers are selling products at prices that the buyer is willing to pay. It is only if there are no products with matching attributes that the buyer seeks will the buyer then only buy products that are cheaper than they are willing to pay. Just as in real life, not every buyer will have the same preferences for products, or even have any preference at all. In this case, they are likely to choose



buyer-behavior: normal
seller-behaviorA: normal
seller-behaviorB: normal
seller-behaviorC: normal
attributes-sellerA: 1
attributes-sellerB: 3
attributes-sellerC: 4
population-buyers: 1
starting-money: 500
starting-priceA-global: 11
starting-priceB-global: 3
starting-priceC-global: 12

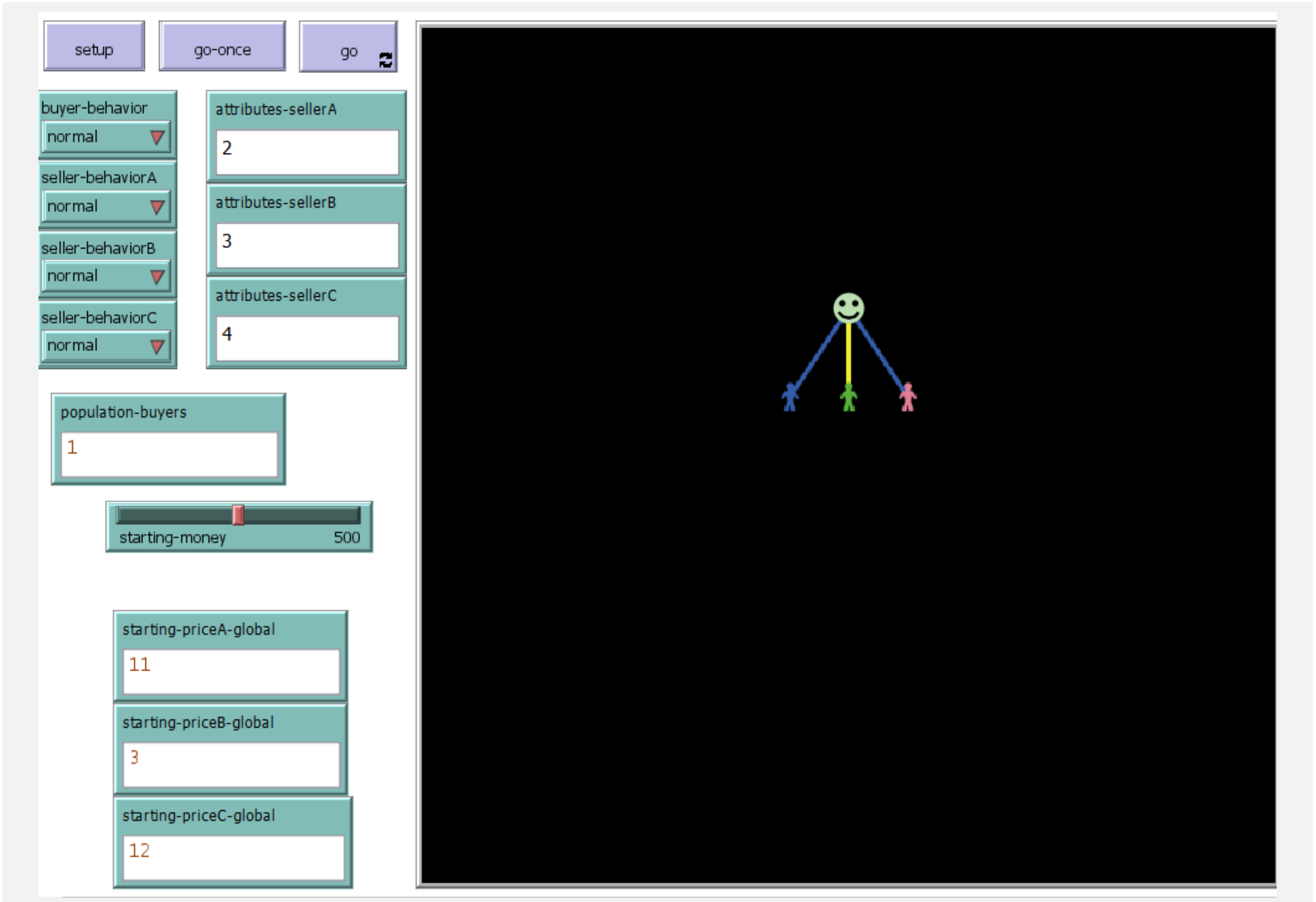
Figure 6: Screenshot of the model in development at its fourth level of complexity, where buyers need to choose between buying products with attributes they desire or buying the cheapest products. In this example, because Seller A has attributes the buyer desires, the buyer buys from Seller A despite cheaper options on the market.

the cheapest option.

In Figure 6, a pink line indicates a purchase made on behalf of matching attributes between buyer and seller (see how the green input element "attributes-Seller A" is "1," which matches the buyer's preference for attributes coded as "1"). So even though the middle seller is selling their product cheaply, the buyer still only buys from the seller on the left. In Figure 7, the attributes of the left seller have changed (it is now "2" instead of "1"), so the buyer does not have any choices on the market for attributes they desire. In this case, they will only purchase from the middle seller because they have the best price point, which the buyer is willing to pay. The yellow indicates this purchase. Now that buyers must consider attributes, an extra level of complexity is added that requires the modeler to make new assumptions. In this case, buyers will first decide who to buy from based on attribute preferences. If no attributes available in the marketplace match what the buyer prefers, the buyer will default to purchasing from the cheapest seller, as was the assumption in earlier stages of the model.

Step 4: Slowly Add Complexity and Realism to the Model by Connecting Variables and Assigning Agent Behavior Based on Empirical Studies

Adding other variables to this relationship can add realism, such as by assigning a number of items that the seller owns. For example, each time there is a successful transaction, the seller will own one less



buyer-behavior: normal
seller-behaviorA: normal
seller-behaviorB: normal
seller-behaviorC: normal
attributes-sellerA: 2
attributes-sellerB: 3
attributes-sellerC: 4
population-buyers: 1
starting-money: 500
starting-priceA-global: 11
starting-priceB-global: 3
starting-priceC-global: 12

Figure 7: Screenshot of the model in development at its fourth level of complexity, where buyers need to choose between buying products with attributes they desire or buying the cheapest products. In this example, because no seller has attributes the buyer desires, the buyer buys from Seller B because they have the cheapest product available.

item. If the seller has no more items to sell, no more transactions can be made. Consider how the order of operations of how buyers and sellers interact may impact this in a scenario with multiple buyers. If there is one seller that has only three items of one product and both buyers want to buy it during each trip to the marketplace (tick), Buyer 1 will buy the first item, and Buyer 2 will buy the second during the first tick, and Buyer 1 will buy the third item during the second tick. This leaves Buyer 2 without the option of buying that product because it is sold out. So, the modeler needs to be careful when interpreting these results, as in this case it would be a mistake to assume that Buyer 1 had greater ambition than Buyer 2 to buy the products because at the end of the model run, they had purchased more of that product.

Once more than one buyer and seller agent are introduced, the user must start to make certain assumptions about how the two sets of agents interact. For example, because several studies on consumer behavior find that cost is the first determinant for consumers deciding what product to buy, one assumption is if there is one buyer and multiple sellers that sell the same item, the buyer will purchase from the seller who is selling their item at the cheapest price. However, more assumptions must be made if there are several factors at play for when a consumer is deciding. For example, if a seller is selling products with a specific attribute, such as if it is certified organic, how does that compare to another seller’s product that is cheaper, but not certified organic? These are the questions a modeler must think about throughout the duration of model development.

Several studies exist that demonstrate behavior of consumers and producers that can be referenced when assigning behaviors to the model’s agents. Consumer surveys demonstrate that consumers are often willing to pay a premium for poultry products with desirable traits, such as if it is labeled organic or has other ethical production claims (antibiotic free, free range, pasture-raised, etc.). Several variables determine these consumer behaviors, such as gender and age, or income levels (Fatha and Ayoubi 2023; Mohammadi, Saghaian, and Boccia 2023). Education level and awareness of ethical food production also contributes to a consumer’s willingness to pay premiums for poultry with enhanced labeling (Kamphuis, Bekker-Grob, and van Lenthe 2015; Karavolias et al. 2018). Van Loo et al. (2011) determined that consumers seeking general organic poultry breast were willing to pay a 35 percent premium above the conventional breast price, and those seeking USDA-certified organic poultry breast were willing to pay a 103 percent premium. Lai and Yue (2020) compiled a list of more than twenty similar scoping studies that measured consumers’ willingness to pay a premium for products labeled organic, covering foods such as fruits, dairy, and salty snacks.

One final example implemented in the model to add complexity is assigning generalized behaviors to both buyers and sellers. Buyers can be assigned to have a buyer behavior of normal, desperate, random, or a mix. Normal behavior means that the buyer does not change their willingness to pay after each transaction opportunity. Desperate behavior means that the buyer’s willingness to pay increases after each tick in the model, with the willingness to pay increasing even more when there is not a successful purchase. So, in this model’s iteration, if a buyer is willing to pay $5 but does not buy anything, the next round they will be willing to pay 5 percent more than the $5 they were willing to pay previously, indicating a sense of “desperation” in their behavior. Random behavior means that after each round, the buyer’s willingness to pay will fluctuate from being willing to pay 10 percent more to 10 percent less than their previous offer. This behavior mirrors the sporadic nature of a consumer as time passes. A mix behavior means that the buyer’s behavior may be either normal, desperate, or random. The seller’s behavior options operate in the same way, where a seller set to desperate will lower their asking price each time they do not successfully sell one of their items.

The ability to toggle consumer behavior in this manner can offer more possibilities to explore dynamic consumer behavior. For example, much literature put forward about consumer behavior during the COVID-19 pandemic demonstrated the reactionary and occasionally unpredictable nature of consumer buying habits, forcing producers to adapt as well. For example, several studies noted that a perceived scarcity in product availability increased consumer demand, influencing both consumers’ willing to pay and producers' willing to sell levels (Laato et al. 2020; Pantano et al. 2020). Four key factors are always shaping consumer habits. The factors are the current dominant social context, emerging technology, new rules and regulations surrounding shopping, and unpredictable events such as the COVID-19 pandemic (Eger et al. 2021). Assigning dynamic behaviors to buyer and seller agents allows for more flexibility in running scenarios that can mirror real-world marketplaces, such as consumer stockpiling phenomena, which occurred during the COVID-19 pandemic. It is possible to create different populations of buyers each with their own assigned unique behaviors, but that is outside of the scope of this introductory demonstration.

4.6 Notes on Model Theory and If the Model Is Deterministic vs. Stochastic

Rules assigned to agents that drive agents' behavior mirror that of a hierarchical decision tree model. Agents are given sets of scenarios, and then the rules given to them decide how they act. For example, the first stage of interaction between buyer and seller in Scenario 2 (further elaborated below) kickstarts a series of true/false tests to determine the next course of action in the model. The flow of these tests can be generally summarized as follows.

1. Does Buyer A's preference for a specific product attribute match what Seller A is selling? If true, a purchase is made. If false, the test repeats between Buyer A and Seller B. If Buyer A finds no matches with any of the sellers based on product attribute, the next test proceeds based on price.
2. Is Buyer A's willingness to pay value greater than Seller A's willingness to sell value? If true, a purchase is made. If false, the test repeats between Buyer A and Seller B. If Buyer A makes no purchases with any sellers, the tick ends for that buyer. The agent will repeat the process when the next tick starts after all other buyers have gone through the decision tree.
3. If a purchase takes place, the following variables are updated; buyer money, seller money, buyer products desired, seller inventory, and the buyer's and seller's assigned behavior (if they are set to a behavior other than "normal"). If no purchase takes place, none of these variables are updated.

Several small caveats exist within the decision tree schema. For example, all conditions may be met for a purchase to take place (product attribute match or willing to pay value outweighs willing to sell value), but if the seller has no products left because they have all been sold, a purchase will not take place. This also occurs when the buyer has bought their desired number of items or if the buyer does not have sufficient money to make the purchase. The decision tree schema can be further explored in the code, which is made available in this manuscript in the appendix as well as through GitHub. The code is available as an open-source resource.

Each of the two scenarios presented below offer a different perspective on modeling and help illustrate the possibility of needing to run the model multiple times. The model presented in Scenario 1 is a deterministic model, where if the parameters are not changed, the model will produce the same results every time. This is because there is no randomness associated within the model's operations. NetLogo uses Java's "strict math" library, which will produce identical results no matter how many times the model is run or if the model is run on different platforms. However, this changes for Scenario 2. The assigned behaviors of the buyers and sellers in this scenario are switched to an option aside from the default setting of "normal." The coding for these alternative behaviors ("random," "desperate," and "mix") introduces random numbers to determine how much to alter the buyers' and sellers' willing to pay and sell values. This puts the model in a stochastic state, where running the model multiple times will produce different results. This requires the need to fully understand the model's parameters and how exactly randomness is integrated into the model's operations to best understand any stochastic model's results. To note, while not used in this project, the *random-seed* command may be used instead of the *random* command in order to get the same sequence of random numbers each time the model is run, which will create scientifically reproducible results.

5. Run the Model Under Different Parameters to Explore Different Scenarios

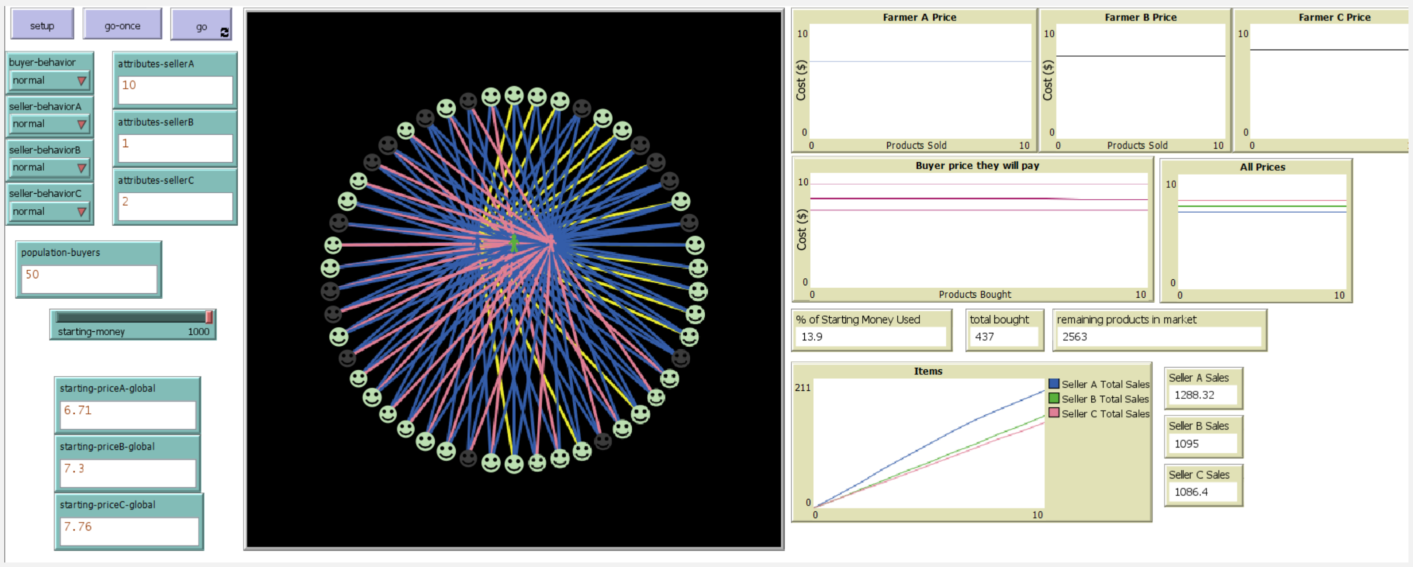
We have set up two different scenarios to demonstrate some potential uses of the modeling framework. The first example demonstrates how buyers have a choice between buying products that are the lowest cost or instead buying products that have desirable attributes, where they are willing to pay a premium for those attributes. We run the model to see how these dynamics reflect in terms of market sales.

Scenario 1: Comparing the Value of Cheapness Versus Selling Products with Desirable Attributes
The parameters for the model are as follows: there are three sellers, each selling a different product. Seller A, representing a large-scale poultry firm, is selling conventionally raised chicken meat at $6.71 per lb., which is the break-even sales cost for North Carolina farmers selling this product in 2021 according to the North Carolina Farm School (2022) and North Carolina Cooperative Extension (n.d.). Seller B, representing a mid-size regional poultry farm, is selling non-GMO pasture-raised chicken meat at $7.30 per lb., and Seller C, representing a small-scale local poultry farm, is selling organic chicken meat at $7.76 per lb., both of which are also the break-even costs for those respective strategies.

Of 50 buyers total, 40 percent will pay the cheapest price and will not factor in product attributes at all, and 40 percent would be willing to pay 10 percent more to receive a more desirable product, in this case non-GMO pasture-raised meat. The remaining 20 percent are willing to pay a 35 percent premium to get organic products. These trait assignments are generalized from the U.S.-based consumer surveys of Vander Mey (2004) and Van Loo et al. (2011). Therefore, the first category of buyers is willing to spend $6.71 per lb., the minimum to buy conventional chicken meat in this scenario, the second category of buyers is willing to spend $7.48 per lb. for non-GMO meat as long as there is a seller with that product selling beneath that price, and the third category is willing to spend $9.06 per lb. as long as there is an organic seller selling cheaper. Buyers are assigned a random number between 5 and 20 for the number of products they wish to buy, which is designed to account for not every buyer buying products at equal intervals across time. When a buyer has bought all of the products they wish to buy, they will no longer participate in the model, indicated by their icon in the model fading to a dark grey. After every successful or non-successful purchase, both buyers and sellers will maintain a constant willing to pay and willing to sell price, so a constant purchasing behavior over ticks is employed.

Upon running the model through ten ticks, the results are presented by Figure 8. Seller A sold 192 products and had total sales of $1,288, Seller B sold 150 products and had total sales of $1,095, and Seller C sold 140 products and had total sales of $1,087. In Figure 8, these results are displayed by the output monitor buttons in the lower right-hand corner. Because in this example the generalized buyer and seller behavior is set to “normal,” the willing to pay and willing to sell values are kept static during each tick (see how the graphs in the right corner indicating these values are flat). In this case, running the model again without changing any parameters will result in the exact same results each time. We will see how this is different in the next example, which introduces dynamic agent behaviors.

Scenario 2: Analyzing Market Dynamics When Buyer and Seller Behavior Is Modified
Poultry market dynamics are constantly changing due to behavioral sensitivity of both buyers and sellers. Over time, internal and external forces may shift the willingness to pay of buyers and willingness to sell of sellers. Additionally, not all buyers will behave in the same manor when deciding on appropriate value they are willing to pay. An agent-based model can capture these adjustments. This



buyer-behavior: normal
seller-behaviorA: normal
seller-behaviorB: normal
seller-behaviorC: normal
attributes-sellerA: 10
attributes-sellerB: 1
attributes-sellerC: 2
population-buyers: 50
starting-money: 1000
starting-priceA-global: 6.71
starting-priceB-global: 7.3
starting-priceC-global: 7.76
% of Starting Money Used: 13.9
total bought: 437
remaining products in market: 2563
Seller A Sales: 1288.32
Seller B Sales: 1095
Seller C Sales: 1086.4

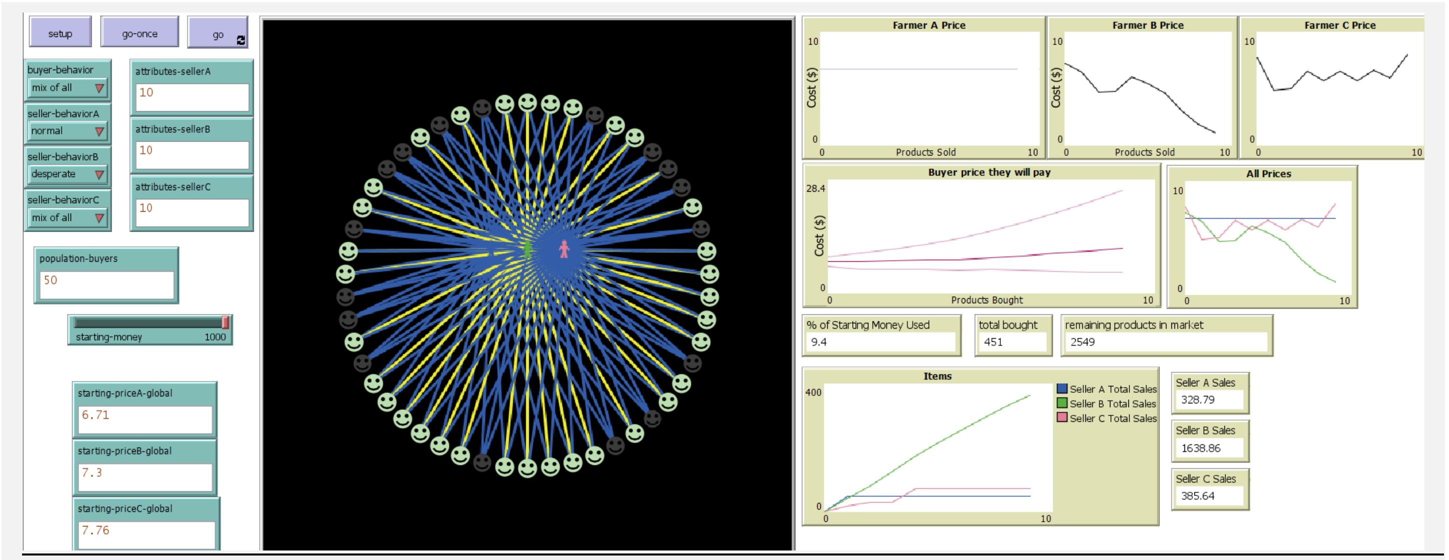
Figure 8: Screenshot of the model interface after running the first scenario.

next scenario introduces four different behavioral options, which were first discussed in Step 5 of the model development section. In this model scenario, the buyers are given a random behavior, where after each tick, their willingness to pay changes a random amount between 5 percent and -5 percent. Seller A's behavior will remain as normal, where their selling price remains constant across ticks. Seller B's behavior is set to "desperate" where with each successful purchase, their asking price will increase 1 percent, while for each unsuccessful purchase their asking price will decrease by 2 percent. This models a seller's potential behavior to further capture the market by decreasing their asking price to find the optimum price buyers will consistently pay. Seller C's behavior is set to "mix of all," which randomly assigns either a "normal," "desperate," or "random" behavior. In this model run, attribute preferences are not accounted for.

Results of the model after ten ticks are displayed in Figure 9. Seller A sold 84 products for a total sales of $584, Seller B sold 314 products for a total sales of $1,615, and Seller C sold 144 products for a total sales of $653. While Seller B had the most total sales, the desperate behavior greatly lowered the asking price of each product so the total average sale was $5.14 per lb., indicating a return below the break-even value for their product production. In Figure 9, notice how the different behaviors influence the plots for each seller's asking price and for the buyer's price they will pay. Seller A's price remained a flat line, Seller B's price trended downward as they continually lowered their price to meet demand, and Seller C's price fluctuated. The buyer's average price they will pay remained relatively constant as indicated by the middle red line. The top red line indicates the maximum amount one of the fifty buyers was willing to spend, which increased up to $11 per lb. at one point. The bottom line indicates the minimum amount the buyers were willing to spend, which dipped down to $5 per lb. near the end of the model cycle. In contrast with the first static example, if we were to run this model again without changing any parameters, the results would be different each time due to the dynamic nature of the assigned behaviors.

6. Discussion and Implication

This paper provides an overview of the definition of ABM, their applications, and some unique features to separate ABM from other types of model frameworks. More interdisciplinary and transdisciplinary studies are choosing ABM to incorporate complex interactions, decisions, outcomes, and consequences



buyer-behavior: mix of all
seller-behaviorA: normal
seller-behaviorB: desperate
seller-behaviorC: mix of all
attributes-sellerA: 10
attributes-sellerB: 10
attributes-sellerC: 10
population-buyers: 50
starting-money: 1000
starting-priceA-global: 6.71
starting-priceB-global: 7.3
starting-priceC-global: 7.76
% of Starting Money Used: 9.4
total bought: 451
remaining products in market: 2549
Seller A Sales: 328.79
Seller B Sales: 1638.86
Seller C Sales: 385.64

Figure 9: Screenshot of the model interface after running the second scenario.

for system analyses. The increasing popularity of ABM has gained attention across the social sciences when dealing with agri-food systems. We offered an example using NetLogo as one toolkit to construct ABM, given its user-friendly model nature and well-documented resources that are easily accessible online. However, this is only one tool, and it poses some challenges.

A first challenge upon initially starting the project is properly conceptualizing how agents interact with each other and how to make those interactions reflect in the model by manipulating the code. Essentially, it can be difficult knowing exactly how to get started. One excellent solution is to explore the model's library built into NetLogo. The model's library has pre-built models that range in application. In looking for ideas in economics, the simple bidding market model (Baker and Wilensky 2017) provides a good overview of how to write code for interactions between a buyer and seller, and also provides a starting point for understanding how to write code for changing buyer and seller behavior. Additionally, noticing the sections of code used to model transactions, namely using the links feature to demonstrate which relationships occur between agents, was a major asset for development of the poultry model.

A second challenge is scaling up from one to multiple agents. Upon having successfully created a model showing the interaction between one buyer and one seller, creating a second seller that the buyer can buy from proved complex given how much of the code needed to be edited to account for the second seller. A solution was to assign variables as global variables rather than turtle-only variables. Using global variables means those variables can be applied to several different agents at once, as opposed to just being applied to one agent. A general tip is to make small changes to your code at a time and work incrementally. Each time the code slowly improves and the model runs with positive changes, note what changes were made via comments or writing in the documentation tab.

The value in setting variables as global variables also applies for setting up dynamic plots or histograms in the interface tab to track variables as the model runs, as tracking variables can only be assigned as global variables. Making sure there is an intuitive way to see and track variables as they change as the model runs is an excellent method for interpreting the model as it is running. Along these lines, adding input buttons to the interface tab will make the model much more intuitive and user-friendly, both for the modeler as well as for any collaborators that do not have coding experience. Using input buttons and sliders allows for changing of model parameters in a more intuitive way rather than going into the programming environment. For example, if you wanted to edit how much total money exists in the simulation, a slider that allows for selecting values between 1.00 and 10,000 is easier than going into the code, navigating to the appropriate variable, and manually entering the desired value.

Last, rather than creating hundreds of new buyer agents explicitly in the code, each with their own unique properties, figuring out how to connect a CSV data file to the model and creating a loop that assigns traits to buyers from the CSV data file is invaluable in saving time and efficiency. The CSV data file extension was introduced after reaching out to the NetLogo community. After watching a brief YouTube instructional video on how to properly connect a CSV data file and learning how to connect variables in the programming environment to variables in the CSV data file, we found it possible to build as many buyer agents as desired. For those who are interested in more examples and coding strategies using NetLogo, a new book is now available: *Modeling Social Behavior: Mathematical and Agent-Based Models of Social Dynamics and Cultural Evolution* by Paul E. Smaldino (2023). A recent review from the *Journal of Economic Literature* states "This book provides advanced undergraduate or graduate students with a thorough introduction to agent-based models as a tool kit for social studies. To this end, the book relies on a widely adopted software package for agent-based models; NetLogo codes for all the models studied in the book are available and referenced in detail when necessary. This makes it possible for the reader to advance in the study of agent-based models without too much coding skill and experience. While the book reads fundamentally as a textbook, it covers enough material in enough depth to represent an interesting introduction to the literature on social dynamics and cultural evolution, so that it could be profitably read by a social scientist looking for a port of entry into these topics" (Bisin 2024).

About the Authors: Bryan Collins is a Visiting Assistant Professor at State University of New York at Oneonta (Corresponding author email: blcollins1@ncat.edu). Chyi-Lyi (Kathleen) Liang is the W.K. Kellogg Distinguished Professor of Sustainable Agriculture at North Carolina Agricultural and Technical State University.

Acknowledgments: This work is supported by USDA Sustainable Agriculture Systems grant no. 2020-69012-31823.

Appendix: NetLogo Code for the Poultry Economy Model

```
extensions [csv]
globals [

  min-price
  population-sellers
  sales-per-tick
  starting-asking-price
  amount-high
  amount-low
  chicks
  fuel-electricity
  feed
  medicine
  transportation
  water
  remaining-costs


  inputdata ;referring to csv files

  ; these variables track data as the model runs
  avg-per-buyer
  avg-per-sellerA
  avg-per-sellerB
  avg-per-sellerC
  total-sales
  remaining-supply
  starting-money-actual
  total-bought

]

breed [ sellersA sellerA ]
breed [ sellersB sellerB ]
breed [ sellersC sellerC ]
breed [ buyers buyer ]
breed [ pops pop ]

turtles-own [
  money        ; keeps track of the amount of money the turtle has
  next-xcor    ; the x-coordinate of the next position
  next-ycor    ; the y-coordinate of the next position
  percent
]
```

```
sellersA-own [
  items-for-saleA ; the quantity that the seller has to sell
  asking-priceA
  asking-price
  starting-supplyA
  behavior-after-sale ; the behavior of seller after a sale
  behavior-no-sale ; the behavior of the seller after a no sale
  sold ; the quantity that the seller has sold

  want-to-buy ; added as a test
  attributes-ownedA
]

sellersB-own [
  items-for-saleB ; the quantity that the seller has to sell
  items-for-saleA
  asking-price
  asking-priceB
  starting-supplyB
  behavior-after-sale ; the behavior of seller after a sale
  behavior-after-saleB
  behavior-no-saleB ; the behavior of the seller after a no sale
  behavior-no-sale
  sold ; the quantity that the seller has sold
  want-to-buy ; added as a test
  attributes-ownedB
]

sellersC-own [
  items-for-saleC ; the quantity that the seller has to sell
  items-for-saleA
  items-for-saleB
  asking-price
  asking-priceB
  asking-priceC
  starting-supplyC
  behavior-after-sale ; the behavior of seller after a sale
  behavior-after-saleC
  behavior-no-saleB ; the behavior of the seller after a no sale
  behavior-no-sale
  sold ; the quantity that the seller has sold
  want-to-buy ; added as a test
  attributes-ownedC
]
```

```
buyers-own [
  want-to-buy ; the quantity the buyer wants to buy
  willing-to-pay
  starting-demand
  behavior-after-purchase
  behavior-no-purchase ; the behavior of the buyer after not buying
  boughtA ; the quantity that the buyer has bought from A
  boughtB ; total bought from seller B
  boughtC
  medicine-costs
  items-for-saleA ;added as test
  items-for-saleB
  items-for-saleC
  starting-priceA
  starting-willing-to-pay

  variable-list var1 var2 var3 ;for get-data command
]

to setup
  clear-all

  directories-and-files

  ; set the global variables
  set min-price 0.01
  ;set population-buyers 3
  set population-sellers 1
  set total-sales 0
  set fuel-electricity 1
  set feed .21
  set water 1
  set medicine .5
  set transportation 1
  set chicks 1.08
  set remaining-costs (6.79 - (chicks + feed))
  set starting-asking-price (chicks + feed + remaining-costs) * 2
  ;set starting-willing-to-pay random 6                                    ;starting willing to pay for buyer!
  set amount-high 10
  set amount-low 20
  ;set starting-priceA-global 5
  ;set starting-priceB-global 5
 ; set changing-priceA-global asking-priceA
```

```
create-ordered-sellersA population-sellers [
 forward 6
 set color blue
 set shape "person"
 setxy -2 2
 set money 0

 set items-for-saleA 1000
 set starting-supplyA items-for-saleA
 set asking-priceA starting-priceA-global
 ;starting asking price for A!!
 set attributes-ownedA attributes-sellerA

 ;set attributes-sellerA "1"
 let mix-behavior ifelse-value seller-behaviorA = "mix of all" [random 3] [-1]
 ifelse seller-behaviorA = "normal" or mix-behavior = 0 [
  set behavior-after-sale [        -> change-priceA 0 ] ;prices changed from 2 and -2 respectively
  set behavior-no-sale    [ hide? -> if (not hide?) [ change-priceA 0] ]
 ] [
  ifelse seller-behaviorA = "desperate" or mix-behavior = 1 [
   set behavior-after-sale [        -> change-priceA 0.7 ]
   set behavior-no-sale    [ hide? -> if (not hide?) [ change-priceA -5.0 ] ]
  ] [
   ; "random" or mix-behavior = 2
   set behavior-after-sale [     -> change-priceA (random 11 - 5)]
   set behavior-no-sale    [     -> change-priceA (random 11 - 5)]
 ] ]
]

create-ordered-sellersB population-sellers [
 forward 0
 setxy 0 2
 set color green
 set shape "person"
 set money 0

 set items-for-saleA 1000
 set items-for-saleB 1000
 set starting-supplyB items-for-saleB
 set asking-priceB starting-priceB-global
 set attributes-ownedB attributes-sellerB

 let mix-behavior ifelse-value seller-behaviorB = "mix of all" [random 3] [-1]
 ifelse seller-behaviorB = "normal" or mix-behavior = 0 [
  set behavior-after-sale [        -> change-priceB 0 ] ;prices changed from 2 and -2 respectively
  set behavior-no-sale    [ hide? -> if (not hide?) [ change-priceB 0 ] ]
 ] [
  ifelse seller-behaviorB = "desperate" or mix-behavior = 1 [
   set behavior-after-sale [        -> change-priceB 0.7 ]
```

```
      set behavior-no-sale    [ hide? -> if (not hide?) [ change-priceB -5.0 ] ]
    ] [
      ; "random" or mix-behavior = 2
      set behavior-after-sale [     -> change-priceB (random 11 - 5)]
      set behavior-no-sale    [     -> change-priceB (random 11 - 5)]
  ] ]
]

create-ordered-sellersC population-sellers [
  forward 0
  setxy 2 2
  set color pink
  set shape "person"
  set money 0

  set items-for-saleA 1000
  set items-for-saleB 1000
  set items-for-saleC 1000
  set starting-supplyC items-for-saleC
  set asking-priceC starting-priceC-global
  set attributes-ownedC attributes-sellerC

  let mix-behavior ifelse-value seller-behaviorC = "mix of all" [random 3] [-1]
  ifelse seller-behaviorC = "normal" or mix-behavior = 0 [
    set behavior-after-sale [         -> change-priceC 0 ] ;prices changed from 2 and -2 respectively
    set behavior-no-sale    [ hide? -> if (not hide?) [ change-priceC 0 ] ]
  ] [
    ifelse seller-behaviorC = "desperate" or mix-behavior = 1 [
      set behavior-after-sale [         -> change-priceC 0.7 ]
      set behavior-no-sale    [ hide? -> if (not hide?) [ change-priceC -5.0 ] ]
    ] [
      ; "random" or mix-behavior = 2
      set behavior-after-sale [     -> change-priceC (random 11 - 5)]
      set behavior-no-sale    [     -> change-priceC (random 11 - 5)]
  ] ]
]

create-ordered-buyers population-buyers [
  forward 10
  facexy 0 0
  set color 58
  ;let new-color [color] of buyer 6 green
  set shape "face happy"
  set items-for-saleA 6;test
  set want-to-buy var2
  set starting-demand want-to-buy
  set money get-starting-value starting-money
  ;set starting-willing-to-pay 5 + random 6
  set starting-willing-to-pay var1
```

```
    set willing-to-pay get-starting-value starting-willing-to-pay * 2

    ask buyers [
      get-data]

    file-close-all

    let mix-behavior ifelse-value buyer-behavior = "mix of all" [random 3] [-1]
    ifelse buyer-behavior = "normal" or mix-behavior = 0 [
      set behavior-after-purchase [-> change-payment 0 ]
      set behavior-no-purchase    [-> change-payment  0]
    ] [
      ifelse buyer-behavior = "desperate" or mix-behavior = 1 [
        set behavior-after-purchase [-> change-payment -1 ]
        set behavior-no-purchase    [-> change-payment  7 ]
      ] [
          ; "random"  or mix-behavior = 2
          set behavior-after-purchase [-> change-payment (random 11 - 5)]
          set behavior-no-purchase    [-> change-payment (random 11 - 5)]
      ]
    ]
  ]

  ; update our tracking variables
  set avg-per-buyer (sum [starting-demand] of buyers) / (count buyers)
  set avg-per-sellerA (sum [starting-supplyA] of sellersA) / (count sellersA)
  set avg-per-sellerB (sum [starting-supplyB] of sellersB) / (count sellersB)
  set avg-per-sellerC (sum [starting-supplyC] of sellersC) / (count sellersC)

  set starting-money-actual sum [money] of buyers

  reset-ticks
end

to-report get-starting-value [ starting-value ]
  report precision (starting-value / 2) 2
end

to go
  if (sum [items-for-saleA] of sellersA = 0 or (0 = count buyers with [money > 0 and want-to-buy > 0])) [
stop ]
  if (sum [items-for-saleB] of sellersB = 0 or (0 = count buyers with [money > 0 and want-to-buy > 0])) [
stop ]
  if (sum [items-for-saleC] of sellersC = 0 or (0 = count buyers with [money > 0 and want-to-buy > 0])) [
stop ]

  clear-drawing
  set sales-per-tick 0
  set remaining-supply (sum [items-for-saleA] of sellersA + sum [items-for-saleB] of sellersB + sum
[items-for-saleC] of sellersC)
  set total-bought (sum [boughtA] of buyers + sum [boughtB] of buyers + sum [boughtC] of buyers)

let sellersA-commerce sellersA
  ask buyers [ do-commerce-withA sellersA-commerce ]
  ask buyers [update-buyer-display]
  ask sellersA [update-seller-displayA]
  set total-sales (total-sales + sales-per-tick)

 let sellersB-commerce sellersB
  ask buyers [ do-commerce-withB sellersB-commerce ]
  ask buyers [update-buyer-display]
  ask sellersB [update-seller-displayB]
  set total-sales (total-sales + sales-per-tick)

 let sellersC-commerce sellersC
  ask buyers [ do-commerce-withC sellersC-commerce ]
  ask buyers [update-buyer-display]
  ask sellersC [update-seller-displayC]
  set total-sales (total-sales + sales-per-tick)


  ; sanity check
  if (any? buyers with [want-to-buy > 0 and willing-to-pay > money]) [ error "Cannot have buyers that
want to pay more than they have cash available!" ]

  tick
end

to update-buyer-display
  if want-to-buy = 0 [
    set color 2
  ]
  set size 1 + (boughtA + boughtB + boughtC / avg-per-buyer) * .01
end

to update-seller-displayA
  if items-for-saleA = 0 [ set color 2 ]
end

to update-seller-displayB
  if items-for-saleB = 0 [ set color 2 ]
end

to update-seller-displayC
  if items-for-saleB = 0 [ set color 2 ]
end
```

```
to do-commerce-withA [sellersA-commerce]
 let asking [asking-priceA] of sellerA 0
 let attributes-desiredA [attributes-ownedA] of sellerA 0
 let attributes-desiredB [attributes-ownedB] of sellerB 1
 let attributes-desiredC [attributes-ownedC] of sellerC 2


 (ifelse
  ;attributes-desiredA = "1" [                    ;changed "1" to var3
  attributes-sellerA = var3 [
  create-link sellersA self pink

  set sales-per-tick (sales-per-tick + 1)
  set want-to-buy (want-to-buy - 1)
  let price asking
  set money precision (money - price) 2
  set money ifelse-value money < min-price [0] [money]
  set boughtA (boughtA + 1)
  ask sellersA [
   set items-for-saleA (items-for-saleA - 1)
   set money precision (money + price) 2
   set sold (sold + 1)
   run behavior-after-sale
   ]
  run behavior-after-purchase
   ]


  attributes-sellerB = var3 or attributes-sellerC = var3 [                                    ;changed "1" to var3
  create-link sellersA self blue
  let hide? (sellers-ignore-full-buyers? and (want-to-buy = 0))
  ask sellersA [ (run behavior-no-sale hide?) ]
  run behavior-no-purchase
 ]
  items-for-saleA > 0 and want-to-buy > 0 and asking <= money and asking <= willing-to-pay and
attributes-sellerA != var3 and asking < [asking-priceB] of sellerB 1 and asking < [asking-priceC] of
sellerC 2[             ;changed "1" to var3
 create-link sellersA self yellow                                        ;if causing issues, remove the "and asking <
[asking-priceB] of sellerB 1 and asking < [asking-priceC] of sellerC 2

   set sales-per-tick (sales-per-tick + 1)
  set want-to-buy (want-to-buy - 1)
  let price asking
  set money precision (money - price) 2
  set money ifelse-value money < min-price [0] [money]
  set boughtA (boughtA + 1)
  ask sellersA [
   set items-for-saleA (items-for-saleA - 1)
   set money precision (money + price) 2
```

```
      set sold (sold + 1)
      run behavior-after-sale
    ]
   run behavior-after-purchase
    ]
        [
    ; else no purchase was made
    create-link sellersA self blue
    let hide? (sellers-ignore-full-buyers? and (want-to-buy = 0))
    ask sellersA [ (run behavior-no-sale hide?) ]
    run behavior-no-purchase
  ])
end

to do-commerce-withB [sellersB-commerce]
  let asking [asking-priceB] of sellerB 1
 let attributes-desiredB [attributes-ownedB] of sellerB 1
  let attributes-desiredA [attributes-ownedA] of sellerA 0
  let attributes-desiredC [attributes-ownedC] of sellerC 2


(ifelse
    ;attributes-desiredB = "1" [
      attributes-sellerB = var3 [
    create-link sellersB self pink

    set sales-per-tick (sales-per-tick + 1)
    set want-to-buy (want-to-buy - 1)
    let price asking
    set money precision (money - price) 2
    set money ifelse-value money < min-price [0] [money]
    set boughtB (boughtB + 1)
    ask sellersB [
      set items-for-saleB (items-for-saleB - 1)
      set money precision (money + price) 2
      set sold (sold + 1)
      run behavior-after-sale
      ]
    run behavior-after-purchase
  ]
   attributes-sellerA = var3 or attributes-sellerC = var3 [
    create-link sellersB self blue
    let hide? (sellers-ignore-full-buyers? and (want-to-buy = 0))
    ask sellersB [ (run behavior-no-sale hide?) ]
    run behavior-no-purchase
  ]
 items-for-saleB > -1 and want-to-buy > 0 and asking <= money and asking <= willing-to-pay and
attributes-sellerB != var3 and asking < [asking-priceA] of sellerA 0 and asking < [asking-priceC] of
sellerC 2[
  create-link sellersB self yellow

      set sales-per-tick (sales-per-tick + 1)
    set want-to-buy (want-to-buy - 1)
    let price asking
    set money precision (money - price) 2
    set money ifelse-value money < min-price [0] [money]
    set boughtB (boughtB + 1)
    ask sellersB [
      set items-for-saleB (items-for-saleB - 1)
      set money precision (money + price) 2
      set sold (sold + 1)
      run behavior-after-sale
    ]
   run behavior-after-purchase
    ]
        [
    ; else no purchase was made
    create-link sellersB self blue
    let hide? (sellers-ignore-full-buyers? and (want-to-buy = 0))
    ask sellersB [ (run behavior-no-sale hide?) ]
    run behavior-no-purchase
  ])
end

to do-commerce-withC [sellersC-commerce]
  let asking [asking-priceC] of sellerC 2
  let attributes-desiredA [attributes-ownedA] of sellerA 0
  let attributes-desiredB [attributes-ownedB] of sellerB 1
  let attributes-desiredC [attributes-ownedC] of sellerC 2


  (ifelse
    ;attributes-desiredC = "1" [
      attributes-sellerC = var3 [
    create-link sellersC self pink

    set sales-per-tick (sales-per-tick + 1)
    set want-to-buy (want-to-buy - 1)
    let price asking
    set money precision (money - price) 2
    set money ifelse-value money < min-price [0] [money]
    set boughtC (boughtC + 1)
    ask sellersC [
      set items-for-saleC (items-for-saleC - 1)
      set money precision (money + price) 2
      set sold (sold + 1)
      run behavior-after-sale
      ]
    run behavior-after-purchase
  ]
   attributes-sellerA = "1" or attributes-sellerB = var3 [
    create-link sellersC self blue
    let hide? (sellers-ignore-full-buyers? and (want-to-buy = 0))
    ask sellersC [ (run behavior-no-sale hide?) ]
    run behavior-no-purchase
  ]

  items-for-saleC > -1 and want-to-buy > 0 and asking <= money and asking <= willing-to-pay and
attributes-sellerC != var3 and asking < [asking-priceB] of sellerB 1 and asking < [asking-priceA] of
sellerA 0 [
  create-link sellersC self yellow

      set sales-per-tick (sales-per-tick + 1)
    set want-to-buy (want-to-buy - 1)
    let price asking
    set money precision (money - price) 2
    set money ifelse-value money < min-price [0] [money]
    set boughtC (boughtC + 1)
    ask sellersC [
      set items-for-saleC (items-for-saleC - 1)
      set money precision (money + price) 2
      set sold (sold + 1)
      run behavior-after-sale
    ]
   run behavior-after-purchase
    ]
        [
    ; else no purchase was made
    create-link sellersC self blue
    let hide? (sellers-ignore-full-buyers? and (want-to-buy = 0))
    ask sellersC [ (run behavior-no-sale hide?) ]
    run behavior-no-purchase
  ])
end

to create-link [ some-seller some-buyer some-color ]
  ask some-seller [
    let oc color
    let x xcor
    let y ycor
    set color some-color
    set pen-size 3
    pen-down
    move-to some-buyer
```

```
    pen-up
    setxy x y
    set color oc
  ]
end

to change-priceA [ change ]
  let before asking-priceA
  set percent 1 + (change / 100)
  set asking-priceA check-for-min-price (precision (percent * asking-priceA) 2)
  if before = asking-priceA [
    if change < 0 and before != min-price [
      set asking-price precision (asking-price - min-price) 2
    ]
    if change > 0 [
      set asking-price precision (asking-price + min-price) 2
    ]
  ]
end

to change-priceB [ change ]
  let before asking-priceB
  set percent 1 + (change / 100)
  set asking-priceB check-for-min-price (precision (percent * asking-priceB) 2)
  if before = asking-priceB [
    if change < 0 and before != min-price [
      set asking-price precision (asking-price - min-price) 2
    ]
    if change > 0 [
      set asking-price precision (asking-price + min-price) 2
    ]
  ]
end

to change-priceC [ change ]
  let before asking-priceC
  set percent 1 + (change / 100)
  set asking-priceC check-for-min-price (precision (percent * asking-priceC) 2)
  if before = asking-priceC [
    if change < 0 and before != min-price [
      set asking-price precision (asking-price - min-price) 2
    ]
    if change > 0 [
      set asking-price precision (asking-price + min-price) 2
    ]
  ]
end

to change-payment [ change ]
```

```
  let before willing-to-pay
  set percent 1 + (change / 100)
  set willing-to-pay check-for-min-price (precision (percent * willing-to-pay) 2)
  if before = willing-to-pay [
    if change < 0 and before != min-price [
      set willing-to-pay precision (willing-to-pay - min-price) 2
    ]
    if change > 0 [
      set willing-to-pay precision (willing-to-pay + min-price) 2
    ]
  ]
  if willing-to-pay > money [ set willing-to-pay money ]
end

to-report seller-cash
  report sum [money] of sellersA + sum [money] of sellersB + sum [money] of sellersC
end

to-report total-bought-per-buyer
  report total-bought
end

to-report average-price
  report (ifelse-value total-sales = 0 [ 0.00 ] [ precision (seller-cash / total-sales) 2 * 3])
end

to-report percent-money-taken
  report 100 * sum [money] of sellersA / starting-money-actual + 100 * sum [money] of sellersB /
starting-money-actual + 100 * sum [money] of sellersC / starting-money-actual
end

to-report remaining-products-to-be-sold
  report remaining-supply
end

to-report remaining-starting-money
  report starting-money - sum [money] of sellersA - sum [money] of sellersB
end

to-report percent-items-sold
  report 100 * sum [sold] of sellersA / sum [items-for-saleA + sold] of sellersA
end

to-report percent-demand-satisfied
  report 100 * sum [boughtA] of buyers / sum [want-to-buy + boughtA] of buyers
end

to-report check-for-min-price [ value ]
  report precision ifelse-value value < min-price [min-price] [value] 2
```

```
end

to directories-and-files
  set inputdata csv:from-file "C:/ABM Work/test.csv"
end

to get-data
  set variable-list []

  set variable-list item (who + 1) inputdata

  set var1 item 0 variable-list
  set var2 item 1 variable-list
  set var3 item 2 variable-list

end

;Copyright Bryan Collins 2023
;Coding ideas pulled from NetLogo Bidding Market Model
;Baker, J. and Wilensky, U. (2017). NetLogo Bidding Market model.
http://ccl.northwestern.edu/netlogo/models/BiddingMarket. Center for Connected Learning and
Computer-Based Modeling, Northwestern University, Evanston, IL.
```

References

Abar, S., G.K. Theodoropoulos, P. Lemarinier, and G.M. O'Hare. 2017. "Agent Based Modelling and Simulation Tools: A Review of the State-of-Art Software." *Computer Science Review* 24:13–33.

Adamski, M., J. Kuzniacka, and N. Milczewska. 2017. "Preferences of Consumers for Choosing Poultry Meat." *Polish Journal of Natural Science* 32(2):261–271.

Axtell, R.L., and J.D. Farmer. 2022. "Agent-Based Modeling in Economics and Finance: Past, Present, and Future." INET Oxford Working Paper No. 2022-10 (Working paper).

Baker, J., and U. Wilensky. 2017. "NetLogo Bidding Market Model." Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston IL. http://ccl.northwestern.edu/netlogo/models/BiddingMarket.

Balmann, A. 1997. "Farm-Based Modelling of Regional Structural Change: A Cellular Automata Approach." *European Review of Agricultural Economics* 24(1):85–108.

Bankes, S.C. 2002. "Agent-Based Modeling: A Revolution?" *Proceedings of the National Academy of Sciences* 99(suppl_3):7199–7200.

Berger, T. 2001. "Agent-Based Spatial Models Applied to Agriculture: A Simulation Tool for Technology Diffusion, Resource Use Changes and Policy Analysis." *Agricultural Economics* 25(2–3):245–260.

Bisin, A. 2024. "Book Review of *Modeling Social Behavior: Mathematical and Agent-Based Models of Social Dynamics and Cultural Evolution* by Paul E. Smaldino." *Journal of Economic Literature*.

Bonabeau, E. 2002. "Agent-Based Modeling: Methods and Techniques for Simulating Human Systems." *Proceedings of the National Academy of Sciences* 99(suppl_3):7280–7287.

Brady, M., K. Kellermann, C. Sahrbacher, and L. Jelinek. 2009. "Impacts of Decoupled Agricultural Support on Farm Structure, Biodiversity and Landscape Mosaic: Some EU Results." *Journal of Agricultural Economics* 60(3):563–585.

Carsey, T.M., and J.J. Harden. 2015. "Can You Repeat That Please?: Using Monte Carlo Simulation in Graduate Quantitative Research Methods Classes." *Journal of Political Science Education* 11(1):94–107.

Chai, S.J., D. Cole, A. Nisler, and B.E. Mahon. 2017. "Poultry: The Most Common Food in Outbreaks with Known Pathogens, United States, 1998–2012." *Epidemiology & Infection* 145(2):316–325.

Chiacchio, F., M. Pennisi, G. Russo, S. Motta, and F. Pappalardo. 2014. "Agent-Based Modeling of the Immune System: NetLogo, a Promising Framework." *BioMed Research International* 2014:907171.

Choe, J. 2023. "Outlook for Livestock and Poultry in 2022," February 23. Washington DC: U.S. Department of Agriculture, 99th Annual Agricultural Outlook Forum. https://www.usda.gov/sites/default/files/documents/2023AOF-livestock-poultry-outlook.pdf.

Cuevas, E. 2020. "An Agent-Based Model to Evaluate the COVID-19 Transmission Risks in Facilities." *Computers in Biology and Medicine* 121:103827.

Dalle Nogare, D., and A.B. Chitnis. 2020. "NetLogo Agent-Based Models as Tools for Understanding the Self-Organization of Cell Fate, Morphogenesis and Collective Migration of the Zebrafish Posterior Lateral Line Primordium." *Seminars in Cell & Developmental Biology* 100:186–198.

De Marchi, S., and S.E. Page. 2014. "Agent-Based Models." *Annual Review of Political Science* 17:1–20.

Delcea, C., L.A. Cotfas, and R. Paun. 2018. "Agent-Based Evaluation of the Airplane Boarding Strategies' Efficiency and Sustainability." *Sustainability* 10(6):1879.

Devarajan, S., and S. Robinson. 2002. "The Impact of AGE Models on Policy." Paper presented at the Conference on Frontiers in Applied General Equilibrium Modeling, Yale University, New Haven CT, April 5–6.

Eger, L., L. Komárková, D. Egerová, and M. Mičík. 2021. “The Effect of COVID-19 on Consumer Shopping Behaviour: Generational Cohort Perspective.” *Journal of Retailing and Consumer Services* 61:102542.

Elkamel, M., A. Valencia, W. Zhang, Q.P. Zheng, and N.B. Chang. 2023. “Multi-Agent Modeling for Linking a Green Transportation System with an Urban Agriculture Network in a Food-Energy-Water Nexus.” *Sustainable Cities and Society* 89:104354.

Epstein, J.M. 2012. *Generative Social Science: Studies in Agent-Based Computational Modeling*. Princeton NJ: Princeton University Press.

Fatha, L., and R. Ayoubi. 2023. “A Revisit to the Role of Gender, Age, Subjective and Objective Knowledge in Consumers’ Attitudes Toward Organic Food.” *Journal of Strategic Marketing* 31(3):499–515.

Fezzi, C., and I.J. Bateman. 2011. “Structural Agricultural Land Use Modeling for Spatial Agro-Environmental Policy Analysis.” *American Journal of Agricultural Economics* 93(4):1168–1188.

Freeman, T., J. Nolan, and R. Schoney. 2009. “An Agent-Based Simulation Model of Structural Change in Canadian Prairie Agriculture, 1960–2000.” *Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie* 57(4):537–554.

Fresco, L.O., F. Geerling-Eiff, A.C. Hoes, L. van Wassenaer, K.J. Poppe, and J.G. van der Vorst. 2021. “Sustainable Food Systems: Do Agricultural Economists Have a Role?” *European Review of Agricultural Economics* 48(4):694–718.

Gebrehiwot, A.A., L. Hashemi-Beni, L.A. Kurkalova, C.L. Liang, and M.K. Jha. 2022. “Using ABM to Study the Potential of Land Use Change for Mitigation of Food Deserts.” *Sustainability* 14(15):9715. https://doi.org/10.3390/su14159715

Gilbert, N. 2019. *Agent-Based Models*. Thousand Oaks CA: Sage Publications.

Happe, K., H. Schnicke, C. Sahrbacher, and K. Kellermann. 2009. “Will They Stay or Will They Go? Simulating the Dynamics of Single-Holder Farms in a Dualistic Farm Structure in Slovakia.” *Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie* 57(4):497–511.

Hare, M., and P. Deadman. 2004. “Further Towards a Taxonomy of Agent-Based Simulation Models in Environmental Management.” *Mathematics and Computers in Simulation* 64(1):25–40.

Junior, C.J.C., and A.C. Garcia-Cintado. 2018. “Teaching DSGE Models to Undergraduates.” *EconomíA* 19(3):424–444.

Kamphuis, C.B., E.W. de Bekker-Grob, and F.J. van Lenthe. 2015. “Factors Affecting Food Choices of Older Adults from High and Low Socioeconomic Groups: A Discrete Choice Experiment.” *The American Journal of Clinical Nutrition* 101(4):768–774.

Karavolias, J., M.J. Salois, K.T. Baker, and K. Watkins. 2018. “Raised Without Antibiotics: Impact on Animal Welfare and Implications for Food Policy.” *Translational Animal Science* 2(4):337–348.

Kerr, C.C., R.M. Stuart, D. Mistry, R.G. Abeysuriya, K. Rosenfeld, G.R. Hart, ... D.J. Klein. 2021. “Covasim: An Agent-Based Model of COVID-19 Dynamics and Interventions.” *PLOS Computational Biology* 17(7):e1009149.

Klügl, F., and A.L. Bazzan. 2012. “Agent-Based Modeling and Simulation.” *AI Magazine* 33(3):29–29.

Kowalska-Pyzalska, A. 2017. “Willingess to Pay for Green Energy: An Agent-Based Model in NetLogo Platform.” Presentation at the 2017 14th International Conference on the European Energy Market (EEM), Dresden, Germany, June 6–9.

Kremmydas, D., I.N. Athanasiadis, and S. Rozakis. 2018. “A Review of Agent Based Modeling for Agricultural Policy Evaluation.” *Agricultural Systems* 164:95–106.

Laato, S., A.N. Islam, A. Farooq, and A. Dhir. 2020. “Unusual Purchasing Behavior During the Early Stages of the COVID-19 Pandemic: The Stimulus-Organism-Response Approach.” *Journal of Retailing and Consumer Services* 57:102224.

Lai, Y., and C. Yue. 2020. “Consumer Willingness to Pay for Organic and Animal Welfare Product Attributes: Do Experimental Results Alig with Market Data?” *Journal of Agricultural and Resource Economics*.

Liang, C., and Z. Plakias. 2022. “Chapter 86: Interdisciplinary System and Network Perspectives in Food and Agricultural Economics.” In C.B. Barrett and D.R. Just, eds. *Handbook of Agricultural Economics.* London: Elsevier, 4705–4779. https://doi.org/10.1016/bs.hesagr.2022.03.002.

Martínez Michel, L., P. H. Punter, and W.V. Wismer. 2011. “Perceptual Attributes of Poultry and Other Meat Products: A Repertory Grid Application.” *Meat Science* 87(4):349–355.

Mielke, R.R., M.W. Scerbo, K.T. Gaubatz, and G.S. Watson. 2009. “A Model for Multidisciplinary Graduate Education in Modelling and Simulation.” *International Journal of Simulation and Process Modelling* 5(1):3–13.

Miksch, F., B. Jahn, K.J. Espinosa, J., Chhatwal, U. Siebert, and N. Popper. 2019. “Why Should We Apply ABM for Decision Analysis for Infectious Diseases?—An Example for Dengue Interventions.” *PloS one* 14(8):e0221564.

Mohammadi, H., S. Saghaian, and F. Boccia. 2023. “Antibiotic-Free Poultry Meat Consumption and Its Determinants.” *Foods* 12(9):1776.

Monti, C., M. Pangallo, G. De Francisci Morales, and F. Bonchi. 2023. “On Learning Agent-Based Models from Data.” *Scientific Reports* 13:9268.

Muis, J. 2010. “Simulating Political Stability and Change in the Netherlands (1998–2002): An Agent-Based Model of Party Competition with Media Effects Empirically Tested.” *Journal of Artificial Societies and Social Simulation* 13(2):4.

Murphy, K. J., Ciuti, S., & Kane, A. (2020). An Introduction to Agent-based Models as an Accessible Surrogate to Field-based Research and Teaching. *Ecology and Evolution*, *10*(22), 12482-12498.

North Carolina Cooperative Extension. n.d. “Best Practices for Buying Meat Directly From Farmers.” https://www.ncat.edu/caes/cooperative-extension/covid-19/best-practices-for-buying-meat.php.

North Carolina Farm School. 2022. “NC Choices Meat Chicken Cost Breakout.” https://ncfarmschool.ces.ncsu.edu/wp-content/uploads/2016/08/NC-Choices-NC-Farm-School-Meat-Chicken-Info-graphic-Breakout.pdf?fwd=no.

Norton, G.W., J. Alwang, and W.A. Masters. 2021. *Economics of Agricultural Development: World Food Systems and Resource Use*. London: Routledge.

Oren, T.I., S.K. Numrich, A.M. Uhrmacher, L.F. Wilson, and E. Gelenbe. 2000. “Agent-Directed Simulation-Challenges to Meet Defense and Civilian Requirements.” *2000 Winter Simulation Conference Proceedings* 2:1757–1762.

Pantano, E., G. Pizzi, D. Scarpi, and C. Dennis. 2020. “Competing During a Pandemic? Retailers’ Ups and Downs During the COVID-19 Outbreak.” *Journal of Business Research* 116:209–213.

Platow, M.J. 2012. “PhD Experience and Subsequent Outcomes: A Look at Self-Perceptions of Acquired Graduate Attributes and Supervisor Support.” *Studies in Higher Education* 37(1):103–118.

Schipmann-Schwarze, C., and U. Hamm. 2020. “Exploring Drivers and Barriers for Organic Poultry Consumption.” *British Food Journal* 122(12):3679–3693.

Shamil, M.S., F. Farheen, N. Ibtehaz, I.M. Khan, and M.S. Rahman. 2021. “An Agent-Based Modeling of COVID-19: Validation, Analysis, and Recommendations.” *Cognitive Computation*:1–12.

Shiflet, A.B., and G.W. Shiflet. 2014. “An Introduction to Agent-Based Modeling for Undergraduates.” *Procedia Computer Science* 29:1392–1402.

Silva, P.C., P.V. Batista, H.S. Lima, M.A. Alves, F.G. Guimarães, and R.C. Silva. 2020. “COVID-ABS: An Agent-Based Model of COVID-19 Epidemic to Simulate Health and Economic Effects of Social Distancing Interventions.” *Chaos, Solitons & Fractals* 139:110088.

Singh, D., L. Padgham, and B. Logan. 2016. "Integrating BDI Agents with Agent-Based Simulation Platforms." *Autonomous Agents and Multi-Agent Systems* 30:1050–1071.

Smaldino, P. 2023. *Modeling Social Behavior: Mathematical and Agent-Based Models of Social Dynamics and Cultural Evolution*. Princeton NJ: Princeton University Press.

Sun, Z., I. Lorscheid, J.D. Millington, S. Lauf, N.R. Magliocca, J. Groeneveld, ... C.M. Buchmann. 2016. "Simple or Complicated Agent-Based Models? A Complicated Issue." *Environmental Modelling & Software* 86:56–67.

Tesfatsion, L., and K.L. Judd, eds. 2006. *Handbook of Computational Economics: Agent-Based Computational Economics*. London: Elsevier.

Tisue, S., and U. Wilensky. 2004. "Netlogo: A Simple Environment for Modeling Complexity." *International Conference on Complex Systems Proceedings* 21:16–21.

U.S. Department of Agriculture, Economic Research Service. 2022. "Poultry Sector at a Glance." https://www.ers.usda.gov/topics/animal-products/poultry-eggs/sector-at-a-glance/.

U.S. Department of Agriculture, National Agriculture Statistics Service. 2022. "Results from the 2021 Organic Survey." *NASS Highlights* (2022-9), December. https://www.nass.usda.gov/Publications/Highlights/2022/2022_Organic_Highlights.pdf.

U.S. Department of Agriculture, National Agriculture Statistics Service. 2023. *Poultry – Production and Value: 2022 Summary* (1949-1573). Washington DC, April. https://downloads.usda.library.cornell.edu/usda-esmis/files/m039k491c/wm119387d/5138kw352/plva0423.pdf.

Van Dyke Parunak, H., R. Savit, and R.L. Riolo. 1998. "Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide." *Proceedings of the First International Workshop on Multi-Agent Systems and Agent-Based Simulation:*10–25.

Van Loo, E.J., V. Caputo, R.M. Nayga Jr, J.F. Meullenet, and S.C. Ricke. 2011. "Consumers' Willingness to Pay for Organic Chicken Breast: Evidence from Choice Experiment." *Food Quality and Preference* 22(7):603–613.

Vander Mey, B.J. 2004. "The Globalization of Food and How Americans Feel About It: Results of Two Surveys." *Journal of Food Distribution Research* 35(856-2016-57069):6–17.

Velasco-Muñoz, J.F., J.M.F. Mendoza, J.A. Aznar-Sánchez, and A. Gallego-Schmid. 2021. "Circular Economy Implementation in the Agricultural Sector: Definition, Strategies and Indicators." *Resources, Conservation and Recycling* 170:105618.

Velasquez, C.G., K.S. Macklin, S. Kumar, M. Bailey, P.E. Ebner, H.F. Oliver, ... M. Singh. 2018. "Prevalence and Antimicrobial Resistance Patterns of Salmonella Isolated from Poultry Farms in Southeastern United States." *Poultry Science* 97(6):2144–2152.

Wilensky, U. 1999. "NetLogo." Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston IL. http://ccl.northwestern.edu/netlogo/.