

(Preview Draft)

Appendix C. Software Choices and Individual-Based Models

This appendix describes a variety of software programs for system dynamics modeling. It begins with spreadsheet programs since these are probably the most widely recognized software. The appendix demonstrates that spreadsheets can be constructed to simulate behavior over time. But this is not their best use; spreadsheets are better used to support dynamic modeling in programs like Stella or Vensim.

Most current applications of system dynamics are implemented in Stella, Vensim or Powersim. Many useful applications from earlier decades were implemented in Dynamo, the first software to implement Forrester's ideas. This appendix shows the visual similarity of Stella, Vensim, Powersim and Dynamo. It then describes related software such as Simile, Simulink and GoldSim. These programs also provide icon-based methods for numerical simulation of system dynamics models.

The appendix then changes the topic from system dynamics modeling to individual-based modeling. This new type of model represents each and every individual in the population. If we know the rules to describe individual behavior, the models could lead to insights on the emergent behavior of the entire population. These results can then be used to support the aggregate relationships that appear in system dynamics models.

Spreadsheets: The Most Familiar Software

Spreadsheet programs are in wide use. If you are looking for a software that is widely known, spreadsheets might be the answer. So it is natural to ask if we can build system dynamics models in a spreadsheet? And if we can, should we do so?

Let's use the sales model from chapter 7 for illustration. It may be implemented in a spreadsheet if we make the effort to deal with time. Time is the most important variable in dynamic models. Stella and Vensim make it easy to define and control the passage of time, but we have to deal with time ourselves in a spreadsheet. Table C.1 illustrates with the sales model. It shows that time occupies an entire row and is advanced by one DT as we move from column to column. DT is set to 0.25 years, so we need four columns for the first year. (The other 19 years of the simulation are outside our view.)

The spreadsheet begins with the constants; these may be located at the top because their values will not change over time. We assign a row to time, and a new row for each of the stocks. We have only one stock in the sales model. We set the initial value to 50 in the first column. The subsequent values will be found by the action of the flows located at the bottom of the spreadsheet. The next step is to assign a separate row to each of the converters. Their values are found by formulas based on the discussion in chapter 7. For example, the initial widget sales is 36,500. This makes sense when 50 people are selling 2 widgets per day for 365 days per year.

Sales Model from Chapter 7.xls						
Start with the constants; then define time to fill a row. The stocks are next.						
Then define the converters (called auxiliaries in Vensim); The flows are last.						
There are five constants						
average annual salary	\$25,000					
exit rate (fraction/year)	0.20					
fraction of revenues to sales	0.50					
hiring fraction (fraction/year)	1.00					
widget price	\$100					
Time (in years with DT = 0.25 years)	0.00	0.25	0.50	0.75	1.00	...
Size of Sales Force (persons)	50.00	53.25	56.71	60.40	64.32	...
There are five converters						
effectiveness (widgets per person per day)*	2.00	2.00	2.00	2.00	2.00	...
widget sales (widgets/year)	36,500	38,873	41,399	44,090	46,956	...
annual revenues (\$/yr)	\$3,650,000	\$3,887,250	\$4,139,921	\$4,409,016	\$4,695,602	...
sales department budget (\$/yr)	\$1,825,000	\$1,943,625	\$2,069,961	\$2,204,508	\$2,347,801	...
budgeted number of sales persons	73.0	77.7	82.8	88.2	93.9	...
There are two flows						
new hires (persons/yr)	23.0	24.5	26.1	27.8	29.6	...
departures (persons/yr)	10.0	10.7	11.3	12.1	12.9	...
The flows are used to update the size of the sales force in the next column						
*The effectiveness is found in a vertical lookup function (located off screen)						

Table C.1. Introductory columns in a spreadsheet model of the sales force.

Nonlinear relationships are somewhat more complicated in spreadsheets. A nonlinear relationship is needed to find the effectiveness of the average sales person. Table C.1 shows the effectiveness at 2.00 in the first five columns. These values are found in a vertical lookup function (BWeb). The flows appear at the bottom of the sheet. The simulation starts with 23 new hires per year and 10 departures per year. The effect of these flows over the 0.25 year interval is a net addition of 3.25 persons, so there are 53.25 persons in the next column. (The numbers are a continuous approximation to the actual number of workers.) The remaining columns are a repetition of previous columns. The hard work is to set up the initial values and formulas. If you build this sheet, you will find that it gives the results shown in chapter 7.

Spreadsheet Discussion

This example demonstrates that we can build stock-and-flow models in a spreadsheet. But should we do so? Spreadsheets allow us to see each and every number, and we can confirm the algebraic manipulations to arrive at each number. If you want to display the numbers in a clear and orderly fashion, spreadsheets are hard to beat. Spreadsheets are also popular because of their widespread use. If your colleagues think in terms of spreadsheets, you may find that the best chance to contribute within your organization is through spreadsheets. Also, the widespread popularity of spreadsheets

has spawned a market for supplemental software to aid in the analysis of uncertainty (i.e., @Risk) or to search for an optimum solution (i.e., *What's Best*). Other specialized software support a wide variety of applications, especially in management science and operations research (LeBlanc and Grossman 2008). Many of the specialized add-ins have been available for over two-decades, providing support for decision analysis, risk analysis, project management, expert systems and forecasting (Bodily 1986).

Clearly, spreadsheets are an intriguing option for modeling. Indeed, Nicolson (2002, 382) suggests that spreadsheets could be a good choice for interdisciplinary modeling, especially compared to “traditional programming languages such as FORTRAN, BASIC or C++.” I believe spreadsheets could be our best choice if we are asked to perform a complex calculation for a single point in time. But this book focuses on behavior which changes over time. Dynamic behavior is more easily and clearly simulated with programs like Stella and Vensim.

The better role for spreadsheets is in support of Stella or Vensim. For example, spreadsheets can provide a convenient way to store, display and check model inputs which are imported to Stella or Vensim. Spreadsheets also provide a convenient way to display or analyze results from dynamic models. Examples are explained later in the book:

- Appendix D describes Vensim results exported to a special spreadsheet to search for the most important inputs to a model.
- Appendix F shows Vensim results exported to a spreadsheet for display of the hour-by-hour operations of the electric power system in the western USA.
- Appendix G shows Vensim inputs to a spreadsheet for shaded display of elevations in a catchment.

Stella, Vensim and Powersim

Stella, Vensim and Powersim are the most widely used programs to implement the system dynamics ideas explained in this book. The programs start with the stocks and flows, the building blocks of system dynamics modeling. The models may be viewed as a collection of first-order differential equations, with a separate equation for each stock in the model. The equations in realistic models are almost always nonlinear, so it makes sense to solve the equations through numerical simulation. The three programs are icon-based, so they promote the development of models with visual clarity. The programs are visually similar, as illustrated with the sales company models shown below. Figs C-1 and C-2 show the Stella and Vensim models. By now, the icon conventions are familiar to you. You can see the similarity at a glance. And if you are inclined to strive for even more similarity, you can change the size and shape of the icons (BWeb). The Powersim version of the sales model is shown in Fig. C-3. The software is grounded in the system dynamics philosophy first published by Forrester. And like Stella and Vensim, Powersim enforces the philosophy by the connections that are permitted in the model.

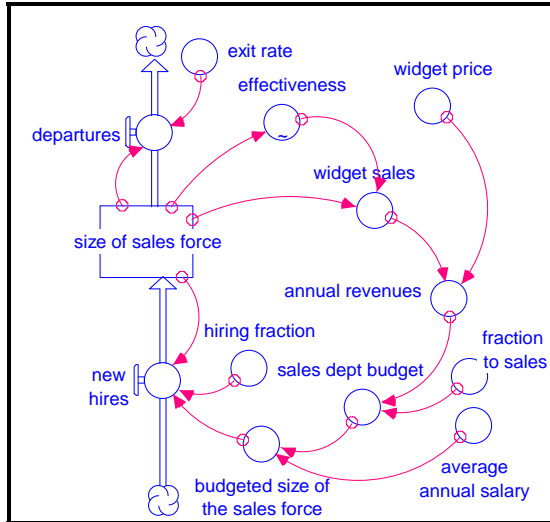


Fig. C.1. Sales model in Stella.

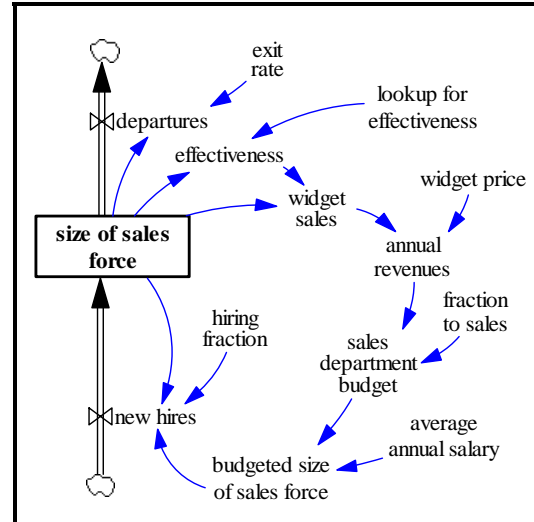


Fig. C.3. Sales model in Vensim.

Fig. C.3. illustrates the visual similarity of Powersim with a flow diagram for the sales model. The stock is the size of the sales force; the flows are new hires and departures. The flows may link one stock to another, or they may link the stocks to clouds, as shown in this diagram. The diamonds are used for the exit rate, the widget price and other constants in the model. The circles are used for widget sales, annual revenues and other variables which change over time. An extra symbol is inserted inside the circle for effectiveness. This reminds us that the effectiveness of the average sales person is determined as a graph function based on the size of the sales force.

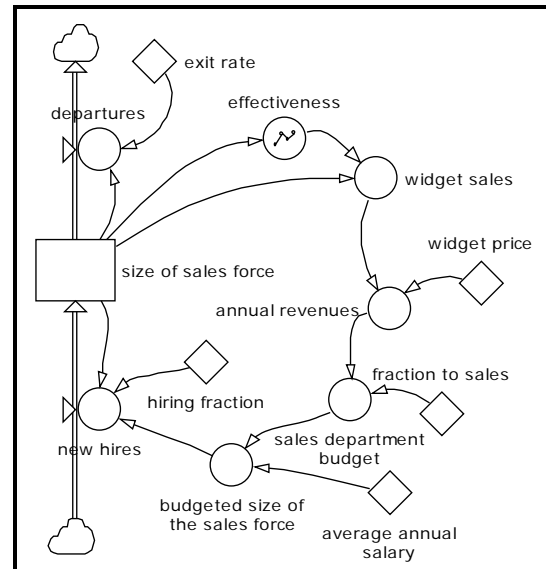


Fig. C.3. Sales model in Powersim.

I have not used Powersim sufficiently in my own work to include a detailed description in this book. But my colleagues who use Powersim can testify to it's usefulness in dynamic modeling. Powersim is valued for its core features to facilitate system dynamics modeling. It is also valued for the capability to simulate in multiple dimensions, to support hierarchical modeling and for ease of interface design (BWeb).

Dynamo

Dynamo is the original program developed for implementing system dynamics models. It is described by Forrester (1961) and by Richardson and Pugh (1981). A wide range of applications were implemented during 1960s – 1980s, including several important application to environmental and resource problems. The most widely known application is undoubtedly the World3 model used in *The Limits to Growth* (Meadows

1972). The work from the 1960s – 1980s provides good examples of modeling practice. A knowledge of Dynamo will help us appreciate and learn from this previous work.

Fig. C.4 shows a diagram drawn in preparation for writing the Dynamo equations for the sales model. These diagrams were drawn to help one see the structure of the model and to get organized to write the equations. Dynamo allowed up to seven characters in a variable name, so a short name appears along side of the long name. In this example, SSF represents the size of the sales force. SSF would be called a stock in this book, but the custom with Dynamo was to call it a *level*. The terminology of the time was that levels grow or decline based on the *rates*, so NH and D are the rates. The short lines represent variables like ER (exit rate) or WP (widget price). These remain constant over time, so they were called constants in the Dynamo equations. The circled variables were called auxiliaries. They may change during the simulation; their role is to help explain the rates (i.e., they were viewed as auxiliary to the rates.) An example is WS, the widget sales, one of the auxiliaries that helps us understand NH, the new hires rate. The diagram shows lines above and below the auxiliary E (effectiveness). These lines alert us to the nonlinear relationship for the effectiveness.

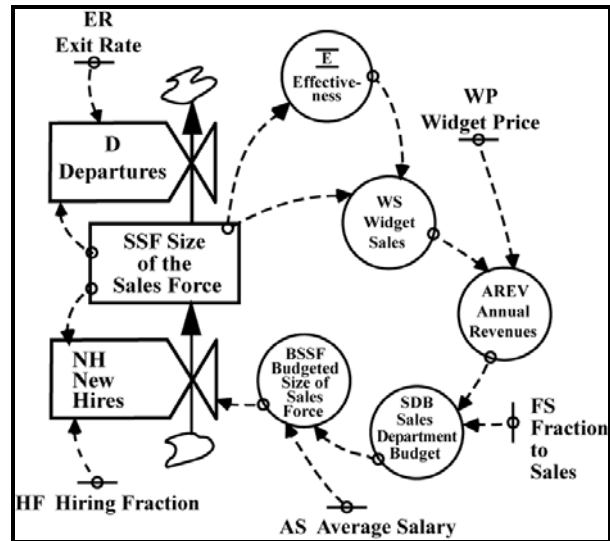


Fig. C.4. Dynamo diagram.

This brief explanation is sufficient for you to see the similarity between Dynamo and the icon-based programs used in this book. The BWeb provides more information, including the Dynamo equations for the sales model. It also provides exercises to learn from some of the models published in *Toward Global Equilibrium* (Meadows 1973). They deal with a wide range of environmental and resource problems. The cases include exploration for natural gas, eutrophication of lakes and cycles in the human and pig populations in the New Guinea Highlands.

Simile

A variety of icon-based programs can be put to good use in dynamic modeling. The programs were developed for general-purpose modeling so they can be adapted to build and test models in the fashion explained in this book. One example is Simile, the software developed by Simulistics, Ltd. (BWeb). It is described as “simulation software with a system dynamics heart.” Simile grew out of simulation applications to earth, environmental and life sciences. It provides similar capabilities to Stella and Vensim, including icons with a close resemblance to the stocks and flows shown in this book. Fig. C.5 illustrates with the familiar model of the sales company. The visual similarity is clear, but the terminology is slightly different. (Simile uses the term “compartment” in the same way as Stella uses the term “stock.”)

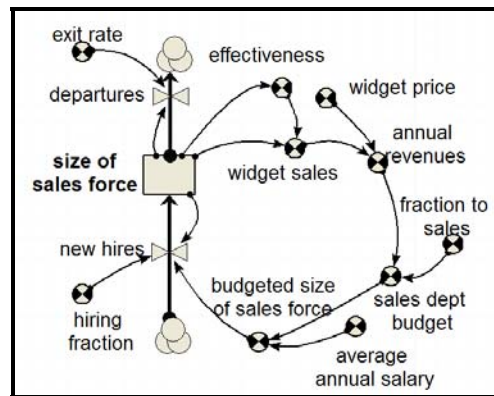


Fig. C.5. Sales model in Simile.

Fig. C.6 further illustrates the close similarity between Simile and Stella. This diagram is taken from the library of models at the Simulistics, Ltd. website (BWeb). It corresponds to the second model of Mono Lake, shown previously in Fig. 5.10.

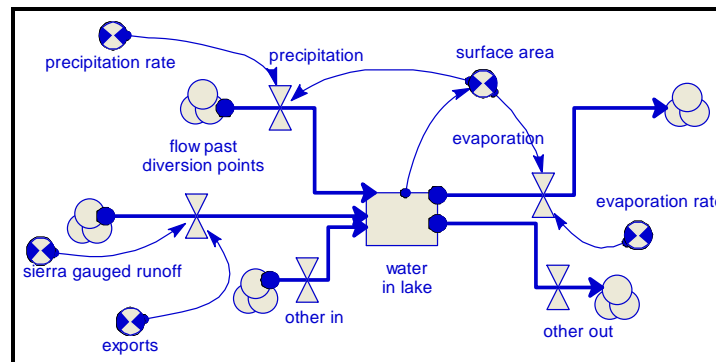


Fig. C.6. The second model of Mono Lake in Simile.

I have only used Simile sufficiently to appreciate the developer's slogan that the software was developed with a "system dynamics heart." My initial tests were also sufficient to see that the developers are striving to serve multiple purposes, many of which go beyond the goals of the system dynamics approach first explained by Forrester. The greater generality brings greater challenges in learning, but it provides opportunities for more complex simulations. My environmental modeling colleagues are particularly interested in exploring Simile's capability for combining spatial modeling with dynamic modeling. These explorations are in progress, so the current publications are limited (Boer 2004, Raper 2005). Perhaps more will be available later (BWeb). Meanwhile, spatially explicit modeling with Stella or Vensim is described in Appendix G.

Simulink and GoldSim

A variety of icon-based programs provide capability for dynamic simulations. Simulink and Goldsim are discussed here. These programs make use of icons to represent stocks and flows, but the visual correspondence is not as strong as with Simile. Readers who are already familiar with these programs will see the correspondence and be able to

develop models similar to those shown in this book. Readers who are not familiar with these programs will make more progress using Stella or Vensim.

Simulink is the dynamic modeling component of MatLab, a multi-purpose modeling software developed by The MathWorks (BWeb). MatLab has become the defacto standard for engineering calculations in academic circles. The Simulink component uses a combination of electrical circuit icons and mathematical symbols to represent the structure of the model. The mathematically sophisticated engineer will see the correspondence between a Simulink model and the models shown in this book. However, my experiments with Simulink lead me to conclude that dynamic modeling is easier and more productive with Stella or Vensim. In my view, the key feature of Simulink is it's position within Matlab. Matlab is valued in engineering circles for its ease of use, its versatility and the very large library of functions. Matlab code can also be compiled into stand alone DLL (dynamic link library). This feature permits a productive link with Vensim's DLL, as explained in Appendix E.

GoldSim is a dynamic modeling software developed by the GoldSim Technology Group (BWeb). It was designed from the ground-up as a general-purpose, probabilistic simulation framework. The initial applications were in civil and environmental engineering with special emphasis on stochastic simulation. Subsequent applications span a range of systems where statistical analysis of stochastic simulations was needed. Readers familiar with GoldSim will know how to implement some of the models shown in this book. Fig. C.6 illustrates with a GoldSim diagram of the second model of Mono Lake. (This is the model shown previously in Stella and in Simile.) The stock variable is the water volume, the amount of water stored in Mono Lake. It changes over time as the software integrates the effect of the total inflows and the total outflows. The diagram illustrates the mix of mathematical and visual symbols used in GoldSim models.

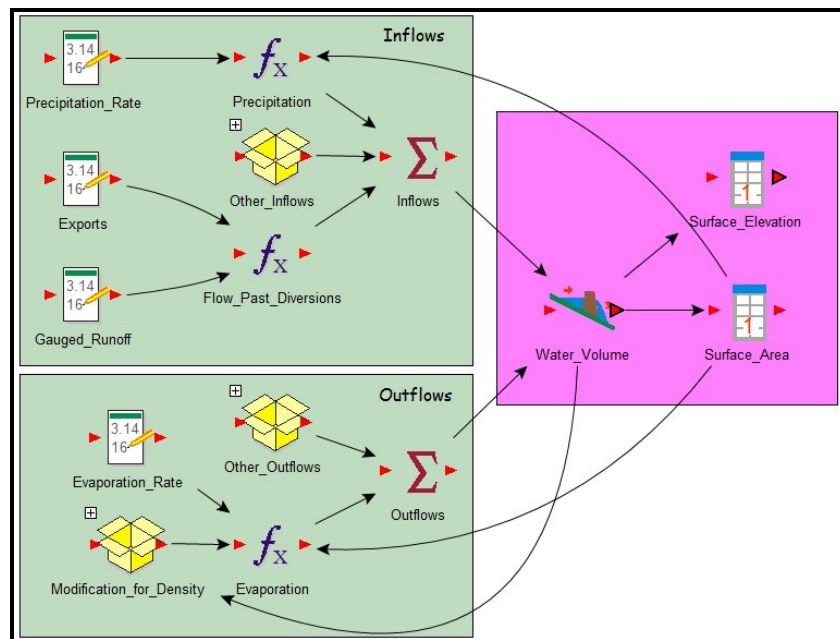


Fig. C.6. Goldsim version of the second model of Mono Lake.

GoldSim, Simulink, Simile, Powersim, Vensim and Stella are among the programs to provide icon-based support for dynamic modeling. These programs are most frequently used to represent the average behavior of the individuals in the system. For example, the sales model simulates the average behavior of the members of the sales force. The average effectiveness is multiplied by the size of the sales force to give total sales, total revenues and the ability of the company to grow. The model delivers insight on the company's dynamic behavior without representing each individual in the sales force.

Individual-Based Modeling

Now, imagine the challenge of building a model of the behavior of each and every person in a large company. There could be hundreds or thousands of employees. Such a model would seem to impose overwhelming computational requirements. But advances in computer power have spawned the development of models which represent each individual in the system. Such models have been used extensively in environmental systems (Grimm and Railsback 2005). A careful review of their use in ecology is provided by Grimm (1999). Individual based models have been used in business systems as well. The individuals may be individual customers or individual firms. These participants are viewed as the agents of behavior, and the models are called agent-based models. Grimm (2005) provides a recent review of the effectiveness of such models in ecosystems, financial systems and urban systems. North and Macal (2007) provide a detailed text on the development of agent-based modeling in business and governmental systems.

The simple example in Fig. C.7 is on the book's website (BWeb). It will illustrate the type of insight that may emerge from modeling individual behavior. There are four individuals hiking up a hill. The first hiker is assumed to walk at a natural pace (yards/minute). His pace might change as the slope changes during the hike, but it is not affected by the other three hikers. The other hikers are assumed to adjust their pace to achieve a suitable separation with the hiker in front of them. (If the gap grows too large, they accelerate the pace to close the gap.) This simple model may be implemented in Stella or Vensim with a separate stock for the distance walked by each individual. If you build and simulate this model (with the slowest hiker in the lead position), a clear pattern will emerge within the first few minutes of the simulation -- the four individuals will be hiking at the same pace as the first hiker, and they will maintain the desired separations. Their progress over the entire hike can then be calculated if we know the pace of the first hiker. And if we want to speed their total progress, we must find a way to improve the pace of the lead hiker (Goldratt 1986).

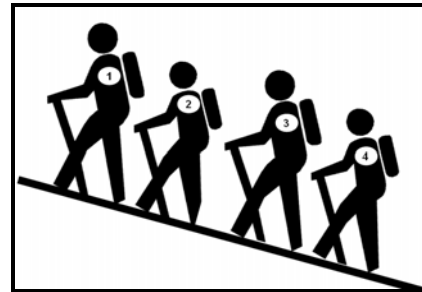


Fig. C.7. Hikers walking up hill.

This simple exercise is easily implemented in Stella because of the small number of individuals and clear rules for how they adjust their behavior. The model has a simple stock and flow structure with stocks representing distance and flows representing pace.

Three of the hikers exhibit goal-oriented behavior, so the model includes three negative feedback loops to achieve the desired separations. The example is easily implemented in Stella because of the relative simplicity of the spatial relationships between the individuals. Each hiker is assumed to follow the same trail up the hill, so their progress can be represented by the distance covered on the trail. In other words, we are simulating the individuals' behavior along a single dimension (the distance covered along the trail.)

Spatial Complexity

The hikers example illustrates an important point about the spatial complexity of the landscape. The hikers may be traveling across a complicated landscape, but that complexity does not necessarily mean that the model of the hikers has to be spatially complex. The key assumption is that the four hikers stay on the trail. This eliminates the need for the model to represent the individuals as they make their way through a three-dimensional landscape.

Let's turn now to situations with more spatial complexity. Think of the V pattern in the sky when geese are migrating. Ask yourself how the geese maintain the V pattern? And how do they select the lead individual? What is their average speed in migration? These are the sort of questions that could be addressed with an individual-based model. If we are willing to adopt some simple assumptions on how each individual controls its relative position in three-dimensional space, the model could help us anticipate the time and effort required to complete the migration.

A model of the geese migration is much more complex than the hiker's model because the geese are moving in three dimensions. A still more complicated situation is presented by the interaction of deer and cougars, the predator and prey populations described in chapter 20. The deer travel across the complicated landscape looking for water and forage while hoping to evade detection by the cougars. The cougars travel across the same landscape looking for opportunities for predation. The spatial interactions are complex indeed! Such interactions can be represented by individual-based models if we are willing to adopt assumptions on the individuals' behavior (BWeb).

The complexity of individual-based modeling has led to the development of software to support spatially-explicit simulation of hundreds or thousands of independent agents. One example is NetLogo, a multi-agent programming language developed at Northwestern University (BWeb). NetLogo is free software with an extensive library of models from a variety of domains (i.e., biology, chemistry, economics, physics and psychology). NetLogo also includes a "system dynamics modeler environment" to enable stock and flow models with relatively close visual similarity to the stock and flow models in this book.

Other programs for support of individual-based modeling are described by North and Macal (2007). They describe "desk top software" such as spreadsheets, Repast Py, NetLogo, StarLogo, Mathematica and MatLab. They view spreadsheets as the simplest

approach, one which can lead to insightful modeling. Other “desktop software” include Repast Py and NetLogo, programs which avoid the spreadsheet limitations on the diversity of agents and restrictions on their behavior. They also describe large-scale software such as Swarm and Repast. They view Repast as the leading free and open-source toolkit for support of large-scale, agent-based modeling.

Individual-Based Modeling and System Dynamics

Individual-based models are sometimes described as “bottom-up” models since the emergent behavior arises from the assumptions adopted for each individual in the population. System dynamics models, on the other hand, are sometimes characterized as “top-down” models since they represent the average behavior of large groups of individuals. Grimm uses the term “top-down” or “state variable modeling” in ecology. His state variable category is not necessarily synonymous with the system dynamics approach explained in this book. But it is sufficiently close for Grimm’s conclusions to apply to the benefits of using individual-based modeling as a complement to system dynamics. Grimm (1999, 139) believes the bottom-up and top-down approaches are not exclusive alternatives but rather complementary approaches which are mutually dependent.”

I believe Grimm is right to point out the complementary nature of the two approaches. Indeed, the findings from individual-based models may provide valuable support for the aggregate relationships in system dynamics models. A model of salmon survival in the Tuolumne River will illustrate. The model was developed by Jager (1997) to represent the day-by-day struggle for survival by thousands of juvenile salmon. The model simulates the competition in a river with space for around 40 thousand redds. The potential redd sites (and the potential feeding locations) are dependent on the flow in the Tuolumne. The river flow is subject to external disturbances and decisions on reservoir releases. The Tuolumne model was designed to help scientists understand the impact of reservoir operations on salmon survival. Such a model might be put to other uses as well. If the model were simulated with a wide range of values for egg deposition, for example, it could provide useful information to support a system dynamics model.

To illustrate, think of the salmon model in chapter 15. It simulates juvenile survival as part of the life-cycle of salmon population of the Tucannon River. Juvenile survival is represented by the nonlinear curve in Fig. 15.5. The general shape corresponds to the Beverton-Holt curve which has proven useful in estimating fish survival. The curve parameters were estimated by Bjornn (1987) based on observations in the Tucannon watershed. Support for these parameters could be provided by an individual-based model similar to Jager’s (1997) model for the Tuolumne River salmon. Her model represented the juveniles competing for suitable feeding locations. The individuals that survive from one day to another become larger and are then in an improved position to command better feeding locations for the next day of the simulation. After 365 days of simulated competition, the model estimates the number of juveniles that out-migrate as smolts. If one were to build a similar model for the Tucannon, it could be used to show the number of out-migrating smolts from simulations

with a wide range of values for the emergent fry. The results of these simulations could then inform the shape of the juvenile survival curve in Fig. 15.5.

Further Reading

Schmickl and Crailsheim (2006) describe an individual-based model of an artificial population of predators and prey. The predators are assumed to adopt a hunting behavior that focuses on an apparently weak individual in the prey population. The individuals in the prey population avoid predation by speed of escape (which may be treated as an evolved characteristic). The model was designed to contribute to the field of artificial life (Adami 1998).

Rahmandad and Sterman (2008) ask when is it better to use agent-based models and when should system dynamics models be used? (The system dynamics models are expressed in the form of differential equations.) The question is addressed with models of the spread of an epidemic. Agent-based models are valued for simulating heterogeneity among individuals. They can also be used to represent the complex network of interactions between groups of individuals. System dynamics models are valued for their ability to simulate behavioral feedbacks and the ease of conducting sensitivity analysis of the behavioral assumptions. The system dynamics approach makes more sense if the impact of feedback effects are expected to be larger than the impact of network structure and heterogeneity among individuals.

Osgood (2009) summarizes the tradeoffs between individual-based and aggregate models. The individual-based models are valued “when studying the impact of interventions in systems where populations exhibit high heterogeneity, small size or clustering, and complex and dynamic network structures.” But these models “frequently require significantly greater time to understand and analyze than do their aggregate cousins.” Osgood describes a technique for dimensional analysis and scale modeling to reduce the long simulation times for individual-based models of large populations.