Modeling Dynamic Systems

Bruce Hannon
Matthias Ruth

Modeling Dynamic Biological Systems

Second Edition

Springer

Modeling Dynamic Systems

Series Editors

Matthias Ruth
Bruce Hannon

For further volumes:
http://www.springer.com/series/3427

Bruce Hannon • Matthias Ruth

Modeling Dynamic Biological Systems

Second Edition

Springer

Bruce Hannon
University of Illinois
Urbana, IL, USA

Matthias Ruth
Northeastern University
Boston, MA, USA

A save–disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems

ISBN 978-3-319-05614-2 ISBN 978-3-319-05615-9 (eBook)
DOI 10.1007/978-3-319-05615-9
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014941224

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Foreword to the First Edition

A mathematical model is a caricature, a deliberate oversimplification of reality. As such, its weaknesses may be transparent, its limitations obvious. This is not so, perhaps, with the invisible model a decision-maker or researcher must inevitably use. But some form of model must be implicit in informed decision-making, lest those decisions become random acts of whimsy; and some form of model similarly guides any experimental design. For all of the limitations of any model, the likelihood is that the explicit consideration of assumptions is a step in the direction of better understanding and better decision-making. One goal of this book is to develop that thesis and to provide the reader the tools to become a better decision-maker or better scientist. A more general goal is simply to make the power of dynamical simulation models available to the widest possible audience of researchers, as aids to exploring the dynamics of ecosystems.

The use of dynamic mathematical models in ecology is not new; indeed, it has a rich and glorious history. In the early part of the twentieth century, the brilliant mathematician, Vito Volterra, challenged by his son-in-law Umberto D'Ancona to explain the oscillations of the Adriatic fisheries, formulated a simple but now-classic pair of differential equations to show how the interaction between predator and prey could drive sustained oscillations. More sophisticated models, incorporating historical effects via Volterra's own specialty, integral equations, were explored; but it was the simplicity of the ordinary differential equation models that captured the attention of later generations. Indeed, the power of simple models as tools for understanding is also a central theme of this book.

Volterra was not alone in laying out the foundations of today's mathematical ecology. Alfred Lotka, an actuary and part-time genius, developed similar equations, as did the Russian V.I. Kostitzin. Generations of mathematicians explored the complexities of these apparently simple equations; and even today, new and esoteric discoveries about bifurcations and chaos are being made. But this research has largely been the domain of mathematicians, and at times has made little contact with biological fact or application. It is time, in the words of the authors, for a "democratization of modeling," driven by their conviction that modeling is too important and too much fun to be left to mathematicians. It is also the case that

modeling in the hands of the uninformed can be a dangerous thing. Hence, as modeling becomes more widely used, it is essential for those who will use the output as well as those who will do the modeling to learn the complexities and limitations to the greatest extent possible; the touchstone of democracy is an informed electorate.

This book could not have been imagined 20 years ago, or consummated even 10 years ago. The advent of computers indeed introduced a new dimension, the ability to go beyond analytical treatments and to use simulation as an experimental tool for much more complex models. High-speed computation has led to increases in what can be done at a dizzying rate. But the initial result of this revolution was to replace one kind of theoretical elite—the mathematician—with another—the computer nerd. The power of the modeling was no more in the hands of the proletariat than it was when the keys to the kingdom were in the intricacies of dynamical systems theory. Hannon and Ruth have set out to change that. Using STELLA as a platform, they have produced the modeling cookbook for those who thought they hated modeling. Their goal, admirably achieved, was to show that some forms of modeling are no harder to operate than a VCR, and that there are intellectual and practical rewards waiting for those willing to venture forth.

The student of this text will be well rewarded, learning about modeling in the only way one really can. . .by putting one's modeling muddy boots on and slogging through exercises. Along the way, the hardy traveler will enjoy a wonderful tour of a wide range of applications in ecology, while learning both techniques and pitfalls. Modeling is just one tool available to the researcher, to be used in concert with observation, experimentation, and hypothesis construction. Simulation such as that taught in this book is just one part of the process, to be combined with analysis and thought. But it is a powerful tool for exploring the consequences of our assumptions, and Hannon and Ruth have provided an introduction that opens this world to a far greater part of the research community than could have been included a decade ago. Word processing seemed forbidding when it was first made available, but now is a skill performed more easily by most Americans than arithmetic or spelling. Modeling too can be made easily available, at least in its simplest forms. Word processing is no substitute for thoughtful composition, and STELLA similarly has limits. For the advanced researcher, or one who wants to become one, more sophisticated approaches beckon. Nonetheless, we are nearing the day when every researcher will be able to use simple tools such as STELLA to construct representations of their hypotheses, and use these to explore broad classes of scenarios unavailable through experimentation. Read on and join the revolution.

Princeton, NJ
August 21, 1995

Simon A. Levin

Preface

The problems of understanding complex system behavior and the challenge of developing easy-to-use models are apparent in the fields of biology and ecology. In real-world ecosystems, many parameters need to be assessed. This requires tools that enhance the collection and organization of data, interdisciplinary model development, transparency of models, and visualization of the results. Neither purely mathematical nor purely experimental approaches will suffice to help us better understand the world we live in and shape so intensively.

Until recently, we needed significant preparation in mathematics and computer programming to develop such models. Because of this hurdle, many have failed to give serious consideration to preparing and manipulating computer models of dynamic events in the world around them. This book, and the methods on which it is built, will empower us to model and analyze the dynamics characteristic of human–environment interactions.

Without computer models we are often left to choose between two strategies. First, we may resort to theoretical, mathematical models that describe the world around us. Mathematics offers powerful tools for such descriptions, adhering to logic and providing a common language by sharing similar symbols and tools for analysis. Mathematical models are appealing in social and natural science where cause and effect relationships are confusing. These models, however, run the risk of becoming detached from reality, sacrificing realism for analytical tractability. As a result, these models are only accessible to the trained scientist, leaving others to "believe or not believe" the model results.

Second, we may manipulate real systems in order to understand cause and effect. One could modify the system experimentally, such as introduce a pesticide or some CO_2, or remove a population or introduce a new one, and then observe the effects. If no significant effects are noted, one is free to assume the action has no effect and increase the level of the system change. This is an exceedingly common approach. It is an elaboration of the way an auto mechanic repairs an engine, by trial and error. But social and ecological systems are not auto engines. Errors in tampering with these systems can have substantial costs, both in the short and long term. Despite growing evidence, the trial-and-error approach remains the meter of the day.

We trust that, just like the auto mechanic, we will be clever enough to clear up the problems created by the introduced change. We let our tendency toward optimism mask the new problems.

However, the level of intervention in social and ecological systems has become so great that the adverse effects cannot be ignored. As our optimism about repair begins to crumble, we take on the attitude of patience toward the inevitable—unassignable cancer risk, global warming, fossil fuel depletion—the list is long. We are pessimistic about our ability to identify and influence cause and effect relationships. We need to understand the interactions of the components of dynamic systems in order to guide our actions. We need to add synthetic thinking to the reductionist approach. Otherwise we will continue to be overwhelmed by details, failing to see the forest for the trees.

There is something useful that we can do to turn from this path. We can experiment using computer models. Models give us predictions of the short- and long-term outcomes of proposed actions. To do this we can effectively combine mathematical models with experimentation. By building on the strengths of each we will gain insight that exceeds the knowledge derived from choosing one method over the other. Experimenting with computer models will open a new world in our understanding of dynamic system. The consequences of discovering adverse effects in a computer model are no more than ruffled pride.

Computer modeling has been with us for over 50 years. Why then are we so enthusiastic about its use now? The answer comes from innovations in software and powerful, affordable hardware available to most individuals. Almost anyone can now begin to simulate real-world phenomena on their own, in terms that are easily explainable to others. Computer models are no longer confined to the computer laboratory. They have moved into every classroom, and we believe they can and should move into the personal repertoire of every educated citizen. Even more important, we believe that the modern biologist and ecologist should, before beginning any lab or field experiments, formulate their hypothesis and construct a model to address it. This struggle for understanding will not only clarify the biological dynamics but also point to the parameters that need the appropriate levels of determination through the ensuing lab and field experiments. Model first, before the lab or field experiment. It is less time and resource consuming and produces more meaningful experiments.

The ecologist Garrett Hardin and the physicist Heinz Pagels have noted that an understanding of system function, as a specific skill, needs to be and can become an integral part of general education. It requires the recognition (easily demonstrable with exceedingly simple computer models) that the human mind is not capable of handling very complex dynamic models by itself. Just as we need help in seeing bacteria and distant stars, we need help modeling dynamic systems. We *do* solve the crucial dynamic modeling problem of ducking stones thrown at us or of safely crossing busy streets. We learned to solve these problems by being shown the logical outcome of mistakes or through survivable accidents of judgment. We experiment with the real world as children and get hit by hurled stones, or we let adults play out their mental model of the consequences for us and we believe them.

These actions are the result of experimental and predictive models and they begin to occur at an early age. In the complex social, economic, and ecological world, however, we cannot rely on the completely mental model for individual or especially for group action, and often we cannot afford to experiment with the system in which we live. We must learn to simulate, to experiment, and to predict with complex models.

In this book, we have selected the modeling software STELLA. Programming languages such as STELLA are changing the way in which we think. They enable each of us to focus and clarify the mental model we have of a particular phenomenon, to augment it, elaborate it and then to do something we cannot otherwise do: to run it, to let it yield the inevitable dynamic consequences hidden in our assumptions and the structure of the model. STELLA and easy-to-use personal computers are not the ultimate tools in this process of mind extension. However, they make the path to freer and more powerful intellectual inquiry accessible to every student.

These are the arguments for this book on *Modeling Dynamic Biological Systems*. We consider such modeling as the most important task before us. To help students learn to extend the reach of their minds in this unfamiliar yet very powerful way is the most important thing we can do.

Urbana, IL, USA — Bruce Hannon
Boston, MA, USA — Matthias Ruth

Contents

Part V Single Population Models

Part VI Multiple Population Models

Part I
Introduction

Chapter 1
Modeling Dynamic Biological Systems

Data can become information if we know the processes involved. Information can become knowledge if we see the system that is operating. But knowledge only becomes wisdom when we can see how any system must change, and can deal with that reality.

(Peter Allen, Coherence, Chaos and Evolution in the Social Context, *Futures*, 1994, Vol. 26, p. 597)

1.1 Process and Art of Model Building

Biologists, from those who study the mechanisms of the nerve cell to those who study ecosystems are in one way or another inescapably involved in dynamic modeling. This book is dedicated to those people, with an understanding of at least some of the problems they face.

The book is about the process and art of modeling. We define the process of model building as an unending one—one with rewards typically proportional to the effort extended. The art of modeling is implied throughout this book, first by virtue of our continuing reference to style of the modeling approach and second by the plethora of modeling examples from nearly every field in biology, and finally by regular reference to the use of analogy.

Modeling style is important. Throughout our life we have learned to develop models in our mind of the processes that we face every day. We do solve an amazing class of dynamic problems, such as hitting baseballs and driving cars, by acquiring through trial and error the skills that are necessary to put the various components of a dynamic system together in our mind, draw the necessary conclusions and react accordingly. However, the more complex the system, the less are we

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_1,

able to sufficiently grasp in our mind its workings, anticipate its behavior, and prepare our actions. We simply cannot hold the many aspects of a dynamic process at once in mind.

The history of modern science, including the life sciences, is one of reduction—an effort to concentrate on ever-smaller parts of the overall system in hopes of advancing knowledge about its constituent parts. Being able to assemble the knowledge about the pieces so something new and meaningful can be said about overall system behavior, however, requires a different kind of knowledge—systems knowledge—and new concepts and tools. Or, as Freeman Dyson (2008) observed:

> I'm 84, so I'm definitely over the hill. If I were starting today as a scientist, I'd certainly study biology. I'd probably be much better at doing biology today than I used to be, because it is now much more of a theoretical subject. Now you can do biology pretty well with computers.
>
> . . ..
>
> The future of biology is exciting and unknown. The main thing is that the era of molecules is over and the era of organisms is here. The reductionist model was the basis of biology in the 20th century, and it was enormously successful – we reduced everything to molecules. We found out wonderful things. The problem in the next century is putting it together. We know pretty much what the building blocks are. The question is, how do they actually function? How does the system work as a system? [1]

We need to be able to capture our knowledge—and possibly that of others—in a consistent and transparent way so that we can better understand, and act in, a changing world. There are by now general rules and computer programs available that have been found useful in letting the modelers quickly get down to the business of capturing their experience inside a computer. Such knowledge capturing is essential to both learning and understanding. But just as we needed microscopes and telescopes to extend the reach of the eye, we need dynamic simulation to extend the reach of our mind. In this process, the computer becomes a facilitator, but it does not substitute for our ability to develop and understand complex dynamic systems, rather, it requires the process and skill of modeling that we called for above. The computer is a means by which we can enlarge our reach into as of yet unexplored territory, and we need to accustom ourselves to the possibilities it opens for us. In this sense, the computer is not unlike other great technologies that required from us that we familiarized ourselves with them and got to understand their powers and limitations.

> We have long been accustomed to machinery which easily out-performs us in *physical* ways. *That* causes us no distress. On the contrary, we are only too pleased to have devices which regularly propel us at great speed across the ground - a good five times as fast as the swiftest human athlete - or that can dig holes or demolish unwanted structures at rates which would put teams of dozens of men to shame. We are even more delighted to have machines that can enable us physically to do things we have never been able to do before: they can lift us into the sky and deposit us at the other side of an ocean in a matter of hours. These achievements do not worry our pride. But to be able to *think* - that has been a very human prerogative [2].

Dynamic modeling is a process of extending our knowledge, and the computer is the only means toward this end. The history of dynamic modeling is traced back to World War II and the immense technical effort mounted by the scientists involved

in the development of automated computation. Dynamic modeling was done before then, but to escape the use by the skilled mathematicians, we had to wait for the advent of machine computation, the development by Jay Forrester of the basic computer logic, and the eventual emergence of this development to the personal computer in the form of STELLA and other easy-to-use programming languages.

STELLA is a graphical programming process that evokes the most easily accessible form of symbolic understanding by humans, the use of icons. As you will find throughout the text, the classification of variables is quite simple and the resulting icons associated with them are quite appropriate for capturing all the parts that influence a system's behavior. An experienced STELLA modeler literally sees and understands the dynamic process through the arrangement of these icons. This is part of what we mean by the modeling art. Now the art can be practiced by anyone with knowledge of basic mathematics and the ability to use a personal computer.

The only way to achieve the facile use of the icons of dynamic modeling is to use them repeatedly in many different applications. Thus our books are a carefully arranged set of models of biological processes that build on each other to various degrees and make use of the same modeling tools in various contexts. We have tried to arrange the models from simple to more complex and by scale, from small to large—from simple growth models of a cell to a rather involved set of interacting spatial ecosystems. As you work through these models you will not only become fluent in the use of the modeling language STELLA but you will develop a new way of thinking about dynamic systems. Practice is the foundation of the modeling art.

Throughout the examples, we try to show how the principal idea of one model can be used again in a different application. The basic growth models are elements of large models. The law of mass action in chemistry is used in epidemic and ecosystem modeling. The play of analogy is dangerous in that it can be misleading but the loss is never more than blush of embarrassment and a little electricity. Much new science seems to ride on analogy: its skillful use is the final piece in the construct of the modeler's art.

The goals of dynamic modeling are to explain and, with enough effort and luck, predict. The dynamic events occurring in the real world are multifaceted, interrelated and difficult, perhaps impossible, to understand. To reduce our worry and to state our curiosity about such events, we pose and then try to answer specific questions about the dynamic processes that seem to comprise these events. Of necessity, we abstract from most of the details and attempt to concentrate on some portion of the larger picture—a particular set of features of the real world. The resulting models are true abstractions of reality. They force us to face the results of the structural and dynamic assumptions we have used in our abstractions.

The modeling process is necessarily complicated and it is unending. Well-posed questions lead us to develop a model—that model leads us to more questions. If we are good at modeling, we improve our understanding of reality. If we are expert at modeling, we approach reality asymptotically. To become expert, we make the modeling process an iterative one: We observe what we call real events from the world, we create an abstract version of these events, highlighting our view of the important elements, then we build and run a dynamic model, we draw

conclusions from the results of the model, we compare our explanation of the events with the reality of such events, and finally we advance our understanding, refine our questions, and improve our models. If we are really good, we make and test predictions with the model. According to John L. Casti, good models are the simplest that explain most of the data from an operating system, and yet do not explain it all, leaving some room for the model, or theory, to grow [3]. Good models must have elements that directly correspond to objects in the real world. All models are necessarily simple constructs of reality; some are just too simple, some just too complex.

The biological modeling process is certainly a knowledge capturing one and yet it is exceedingly difficult to do well. Nature seems ineffable. The ultimate difficulty stems from the complexity of the living system, both its structure and its dynamics, and the impossibility of removing ourselves fully to observational status. Even an expert in biochemistry, in medicine or in ecosystem analysis cannot make much headway against such complexity alone. A team of such people, expert in the various aspects of a problem may be the answer. As a team, researchers can more easily see anthropomorphism in each other. We suspect that such team modeling will become much more common in the near future. The elements that make such a team successful are of course the possession of real and pertinent expertise, compatible personalities, a common concept of the questions to be answered, and a common modeling language. Since the level of modeling expertise is likely to vary among even the best set of experts, a simple modeling language is needed—one that can be understood by each of the team in a very short time. The programming language STELLA meets that requirement. Our book contains the guidelines for the most likely successful process of team or individual modeling and it is based on that language. Many generations of students both at universities, in government, and in corporations have contributed to the process described herein.

1.2 Static, Comparative Static, and Dynamic Models

Three general types of models can be defined. The first type of models is those that represent a particular phenomenon at a point of time, a static model. For example, a map of the world might show the location and size of a city or the location of infection with a particular disease, each in a given year. The second type comprises comparative static models that compare some phenomena at different points in time. This is like using a series of snapshots to make inferences about the system's path from one point in time to another without modeling that process itself.

Other models describe and analyze the very processes by which a particular phenomenon is created. We may develop a mathematical model describing the change in the rate of migration to or from a city, or the change in the rate of the spreading of a disease. Similarly, we may develop a model that represents the change of these rates *over time*. This latter type of models is dynamic in the sense that they attempt to capture the change in real or simulated time.

With the advent of easy-to-use computers and software we can all build on the mathematical descriptions of a system and carry them further. The world is not a static or comparative static process, and so the models treating it in that way will become obsolete, and are perhaps even misleading. We can now investigate in great detail and with great precision the system's behavior over time, its movement toward, or away from, equilibrium positions, rather than restrict the analysis to an equilibrium point itself.

An understanding of the dynamics and changing interrelationships of systems, such as social, biological, and physical systems, are of particular importance in a world in which we face increasing complexity. In a variety of disciplines scientists ask questions that involve complex and changing interrelationships among systems. How do mutation and natural selection affect the distribution of genetic information in a population? How does a vaccination program affect the spread of a disease? All good modeling processes begin (and end) with a good set of questions. These questions keep the modeler focused and away from the miasma of random exploration. Starting with a clear question in mind also helps the modeler decide when the model is done—namely when it yields a satisfactory answer. Conversely, modelers who start with the intention to "model the behavior of a system," without ever having articulated what the question is that they wish to address, tend to keep on going, in part because there always seems to be an aspect of the real world that has not yet found its way into the model, and as a result, the model is perceived as incomplete. And, of course, it will always be incomplete. But should the model ever include all the parts of reality, then it will be as complex as reality, and the modelers will have altogether missed the point of modeling.

Models help us understand the dynamics of real-world processes by mimicking with the computer the actual but simplified forces that are assumed to result in a system's behavior. For example, it may be assumed that the number of people migrating from one country to another is directly proportional to the population living in each country, and migration decreases the further these countries are apart. In a simple version of this migration model, we may abstract away from a variety of factors that impede or stimulate migration, besides those directly related to the different population sizes and distance. Such an abstraction may leave us with a sufficiently good predictor of the known migration rates, or it may not. If our answers do not compare sufficiently well with reality, we re-examine the abstractions, reduce the simplifying assumptions, and re-test the model for its new predictions. The results will not only be a better model of the system under investigation but most importantly a better understanding of our conception of that system, showing us whether we were indeed able to identify and properly represent the essential features of that system.

We cannot overstress the fact that one should keep the model simple, even simpler than one knows the cause and effect relationship to be, and only grudgingly add additional features to the model when it does not reproduce the real effects. After all, it is not the goal to develop models that capture all facets of real life systems. Such models would be useless because they would be as complicated as the systems we wanted to understand in the first place. The real quest of dynamic

modeling is to "discover" the hopefully few rather simple underlying principles that together bring the observed complexity. This is our meaning of simplicity.

Each element of the model is specified by initial conditions and the computer works out the system's responses according to the specified relations among the model elements. The initial conditions may derive from actual measurement, such as the number of people living on an island, or estimates, such as estimates of the number of voles living in a specific garden. The estimates, in turn, could be derived from empirical information or even reasonable guesses by a modeling team. Models built on such uncertain parameters may still be of great value, providing a picture of a particular processes, rather than exact information. Documentation of the parameters and assumptions, always necessary at each step in the modeling process, is important when the modeler's judgment is used.

In the end, models can be no better than the modelers. Hence the elegant statement by Botkin [4] is very appropriate,

> by operating the model the computer faithfully and faultlessly demonstrates the implications of our assumptions and information. It forces us to see the implications, true or false, wise or foolish, of the assumptions we have made. It is not so much that we want to believe everything that the computer tells us, but that we want a tool to confront us with the implications of what we think we know.

1.3 Analogies, Anomalies, and Reality

For many years, physicists have known of the analogous relations between the principle variables in the basic equations of hydraulics, electricity, and mechanical systems. Force, springs, dampers, inertia, velocity, and displacement have their counterparts in voltage, current, resistance, inductance and capacitance and again pressure, mass flow, frictional loss, and vorticity. Coulomb apparently was convinced that the attractive force between charged particles was of the same form as the gravitational attraction between planetary bodies, put forth by Newton centuries before. These analogies are more than curiosities. They show a common worldview of such important phenomena. As scientists developed each of these disciplines in their turn, they recognized the heritage of hard-won successes in describing parts of the real world.

Not only were these analogies useful in physics and engineering, but it is well known that the great economist Walras produced his equations of the economy from the principles of hydraulics. Before him, the medical doctor and physiocratic economist Quesnay divined his input–output tables of the French economy from an analogy with the circulation of blood in the body. Analogy, carried to the right level of detail, is the cornerstone if not the foundation of the creative enterprise.

Analogies abound between economics, biology, and chemistry. For example, the most common production functional form used in economics is:

$$\text{Rate of Production} = Q = AL^{\alpha}K^{\beta} \tag{1.1}$$

where A, α, and β are constants and L is the input of labor and K the input of capital services, in a process that produces say, widgets at rate Q. But to a chemist, this is the law of mass action at work. Q is the production rate of some product made from the reaction of L and K, two chemicals that combine to produce the product. The constants α and β reflect temperature or perhaps pressure effects on the reaction rate. Or if you were an epidemiologist, you would say that Q is the rate at which people are getting sick, that L is the size of the healthy population, K is the size of the sick population, and A is the contact coefficient. If you are a metapopulation theorist in ecology, L would be the number of patches occupied by an inferior species, K the number of superior species, A is the colonization rate of the superior species, and Q is the rate of conversion of the inferior patches to superior ones.

Some ecologists use a variant analogy to the economic production function above. They say that Q, an insect birth rate for example, equals a maximum growth rate for the insect (A) times a series of factors each of whose value varies from 0 to 1, where 1 is the factor value associated with the maximum possible growth rate. These are usually graphically based, experimentally derived factors that naturally show diminishing returns. Examples of such factors are temperature and humidity. Under the exact condition of optimal temperature for the growth rate for this insect, the temperature factor would be 1.0.

In its various uses of the equation described above, the factors are assumed to be completely independent. Capital and labor can be substituted for one another without concern about the availability of the other. But actually, it clearly takes labor to make more capital to substitute for the displaced labor. So the factors are not actually independent although they are commonly assumed to be. Neither are temperature and humidity independent, despite the assumptions in the insect model for example. For a single firm, the independent factor assumption is not a terribly bad one, but to make this assumption for the economy as a whole is absurd. Such assumptions are usually a matter of expediency of model building. Be careful how you use them.

Analogies can also help you spot anomalies. Lightman and Gingerich [5] point to the "retrorecognition" phenomenon, where anomalies in one theory are only recognized when they are explained later by a superseding theory. For a variety of reasons, we scientists are essentially blind to those facts not explained by the dominant theory. By the use of appropriate analogy, *de rigor* for the nineteenth century likes of Lord Kelvin and J. C. Maxwell, we should be able to turn up anomalies in our current explanations of the way things work. A rule of thumb for anomaly-finding is to push your model to its reasonable limits.

Our discussion on the use of analogy is intended to raise your optimism about the idea of dynamic modeling. But not everyone is optimistic. We might be able to accurately simulate some very complex biological system but can we actually learn more about these systems from such models? Can we learn to make wiser decisions from our modeling exercises? We think the answer is "yes" but our view has its dissenters. (For an excellent summary of this argument, see Denning [6].) The counter argument is based on the idea the human problem solving is very context-dependent, while most computer models are not. This sounds to us more like a

complaint than a fatal criticism, that is, we should be able to model in a context-dependent way when that is needed. We should be able to model in such a way that the guiding rules in the model shift as the context shifts. We further argue that the terrific complexity of biological, ecological, and sociological systems can mask the possibility of simple underlying rules. These rules when used together in a model might cause the system behavior to appear exceedingly complex. The quest is to find the underlying rules. Such a quest pushes us well beyond simulation. It is what we mean by the term "dynamic modeling."

1.4 Model Components

The most important elements of a system are the state variables. State variables are indicators of the current status of the system. They are the variables on which all the other calculations in the model are based. State variables come in two flavors: conserved and non-conserved. A conserved state variable represents an accumulation or stock of something—water, people, materials, or information. These stocks are created and destroyed by the results of the control variables in the system. But non-conserved state variables, such as price and temperature, are also possible. Clearly, the temperature of a hot body sitting in a cool room will determine the rate that the body cools. The changing price of a natural resource will signal the changes in its rate of optimal depletion. To maintain simplicity in the model, strive to minimize the number of its state variables.

System elements that represent the action or change in a state variable are called flows or control variables. As a model unfolds in time, control variables update the state variables at the end of each time step. Examples for control variables are the rate of flood water inflowing to a reservoir, the rate of water release from that reservoir, and the rate of water evaporation from its surface—all acting to change the water contained or "conserved" in the state variable, the reservoir.

The remaining set of variables in any model might be classed as converters or transforming variables. They take in parameters or perhaps the results of calculations elsewhere in the model and transform these inputs still further. These results are relayed on to other such transforming variables or to that special class of transforming variables, the control variables. These interactions in a model are often classed in terms of feedback—the flow of information from a state variable through a chain of transforming variables, and ultimately back to the control variables, change that state variable, and so continue in an ever changing loop, perhaps finally reaching a steady state or maybe race off to infinity, to zero or to chaos. The nature of such circulations of information is negative or positive. Negative feedback tends to force state variables toward goals set up either implicitly or explicitly in the model. Negative feedback leads to balance. Positive feedback tends to do the opposite: it allows variables to reinforce differences rather than minimize these differences. Positive feedback, as we shall see in the text has some surprising results. Negative feedback is the basic idea of the controlled dynamic

system, and ultimately it is the confederate of the causally-oriented theorist. Real-world systems typically contain both reinforcing and balancing processes, often with their strengths varying over time.

Variation in feedback processes can be brought about by nonlinear relationships. Such nonlinear relationships are present if a control variable does not depend on other variables in a linear fashion but changes, for example, with the square root of some other variable. As a result of nonlinear feedback processes, systems can exhibit complex dynamic behavior. Similarly, adjustments in system components to influences on them may not occur instantly but be lagged in time. Consequently, system modelers must pay special attention to nonlinearities and lagged effects that describe the relationships among models components.

Throughout this book we encounter a variety of nonlinear feedback processes that give rise to complex system behavior. Let us develop a simple model to illustrate the concepts of state variables, flows, and feedback processes. We will then return to discuss some "principles of modeling" that will help you to structure the model building process in a set of steps.

1.5 Modeling in STELLA

To explore modeling with STELLA, we will develop a basic model of the dynamics of a fish population. Assume you are the sole owner of a pond that is stocked with 200 fish that all reproduce at a fixed rate of 5 % per year. For simplicity, assume also that none of the fish die. How many fish will you own after 20 years?

A run-time version of STELLA can be downloaded for free at the following internet site:

www.iseesystems.com/modelingdynamicbiologicalsystems.

There you will also find the models developed and described in this book. The models are also available at:

www.iseesystems.com/modelingdynamicbiologicalsystems.

In building the model, we will utilize all four of the graphical "tools" for programming in STELLA. On opening STELLA, you will be faced with a user interface that shows, on the left side, a series of tabs. Click on the Model tab and you should see a window as in Fig. 1.1. The left-hand side can be used to develop the model, get an overview over its structure (that is the mapping layer), build a user interface for it, or display the equations, initial conditions and such, on which the

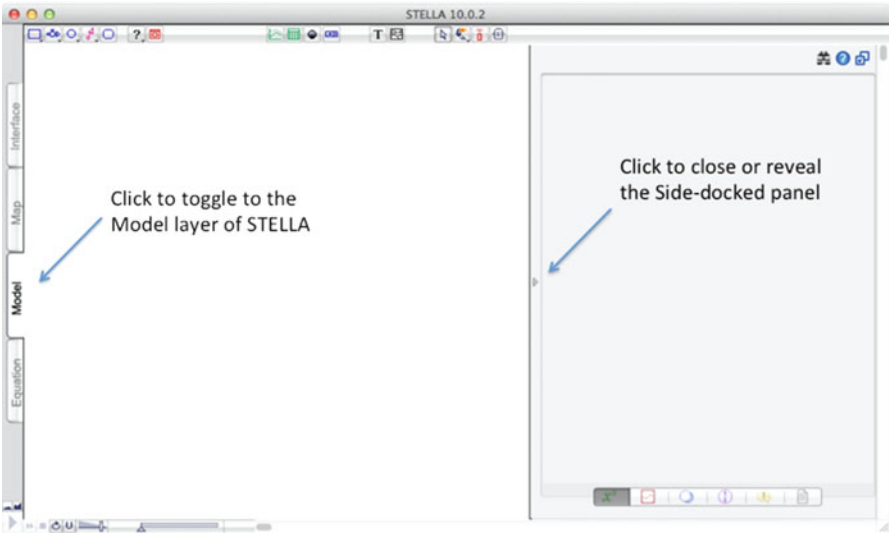

Fig. 1.1

Fig. 1.2

Fig. 1.3

model is based. The right-hand side captures details such as the equations used to calculate a particular parameter, and can be displayed or collapsed by clicking the little triangle. More about all this later.

The model layer displays the following symbols, "building blocks" (from left to right): stocks, flows, converters, action connectors (information arrows), and modules (Fig. 1.2).

We begin with the first tool, a stock (rectangle). In our example model, the stock will represent the number of fish in your pond. Click on the rectangle with your mouse, drag it to the center of the screen, and click again. Type in FISH. This is what you get (Fig. 1.3).

This is the first state variable in our model. Here we indicate and document a state or condition of the system. In STELLA, this stock is known as a reservoir.

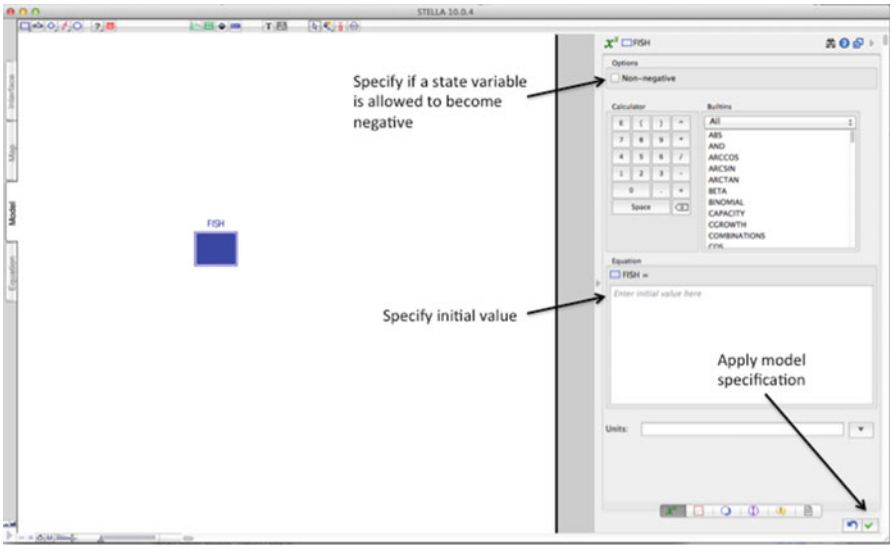

Fig. 1.4

In our model, this stock represents the number of fish of the species we are studying that populate the pond. If we assume that the pond is one square kilometer large, the value of the state variable FISH is also its density, which will be updated and stored in the computer's memory at every step of time (DT) throughout the duration of the model. The fish population is a stock, something that can be contained and conserved in the reservoir; density is not a stock, it is not conserved. Nonetheless, both of these variables are state variables. So, because we are studying a species of fish in a specific area (one square kilometer), the population size and density are represented by the same rectangle.

Inside the rectangle is a question mark. This is to remind us that we need an initial or starting value for all state variables. Click on the rectangle and the side-docked panel will be activated. Its bottom portion is asking for an initial value: "Enter initial value here." Add the initial value you choose, such as 200, using the keyboard or the mouse and the dialogue keypad in the upper part of the side-docked panel. In the upper part, you can also check whether the stock can or cannot become negative. In our case, click the "Non-negative" check-box, since the stock of fish will never be allowed to drop below zero. When you have finished, click the check mark in the lower right-hand side of the side-docked panel to apply your specifications. The question mark in the stock FISH will have disappeared (Fig. 1.4).

We must decide next what factors control (i.e., add to or subtract from) the number of fish in the population. Since we assumed that the fish in your pond never die, we have one control variable: REPRODUCTION. We use the *flow* tool (the right-pointing arrow, second from the left in Fig. 1.2) to represent the control variable, so named because they control the states (variables). Click on the flow symbol, then click on a point about 2 in. to the left of the rectangle (stock) and drag

Fig. 1.5

the arrow to POPULATION, until the stock becomes dashed, and release. Label the circle REPRODUCTION. Figure 1.5 shows what you will have.

Here, the arrow points only into the stock, which indicates an inflow. But, you can get the arrow to point both ways if you want it to. You do this by clicking on the circle in the flow symbol and choosing "Biflow" in the upper portion of the side-docked panel. A biflow enables you to add to the stock if the flow generates a positive number and subtract from the stock if the flow is negative. In our model, of course, the flow REPRODUCTION is always positive and newly born fish go only *into* the population. Our control variable REPRODUCTION is a uniflow: "new fish per annum."

Next we need to know how the fish in our species reproduce. Not the biological details, just how to accurately estimate the number of new fish per annum. One way to do this is to look up the birth rate for the fish species in our pond. Say we find that the birth rate = 5 new fish per 100 adults each year which can be represented as a transforming variable. A transforming variable is expressed as a *converter*, the circle that is third from the right in the STELLA toolbox (Fig. 1.2). So far REPRODUCTION RATE is a constant, later we will allow the reproduction rate to vary. The same clicking and dragging technique that got the stock on the screen will bring up the circle. Click on the converter and then enter in the side-docked panel the number of 0.05 (5/100). In the upper right-hand side of the side-docked panel is an impressive list of "built-in" functions that we can use for more sophisticated model specifications. We'll use some of those later.

At the right of the STELLA toolbox (Fig. 1.2) is the *connector* (information arrow). We use the connector to pass on information (about the state, control, or transforming variable) to a circle, to control or transforming variable. In this case, we want to pass on information about the REPRODUCTION RATE to REPRODUCTION. Once you draw the information arrow from the transforming variable REPRODUCTION RATE to the control and from the stock FISH to the control, open the control by double clicking on it. Recognize that REPRODUCTION RATE and FISH are listed in the side-docked panel as two required inputs for the specification of REPRODUCTION. Note also that STELLA asks you to specify the control: REPRODUCTION = . . . "Enter equation here." Click on REPRODUCTION, then on the multiplication sign in the lower portion of the side-docked panel and then on FISH to generate the equation:

$$\text{REPRODUCTION} = \text{REPRODUCTION RATE} * \text{FISH} \tag{1.2}$$

Click the check mark in the lower right-hand side of the side-docked panel, and the question mark in the control REPRODUCTION disappeared. Your STELLA diagram should now look like the one in Fig. 1.6.

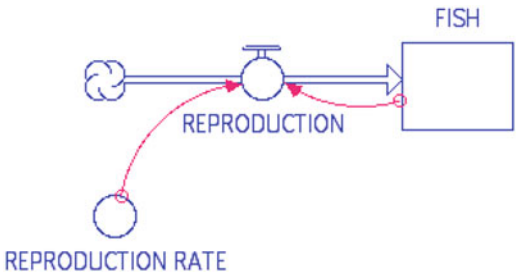

Fig. 1.6

Fig. 1.7

Next, we set the temporal (time) parameters of the model. These are DT (the time step over which the stock variables are updated) and the total time length of a model run. Go to the RUN pull-down menu on the menu bar and select Time Specs. . . A dialogue box will appear in which you can specify, among other things, the length of the simulation, the DT, and the units of time. We arbitrarily choose DT = 1, length of time = 20, and units of time = years.

To display the results of our model, click on the graph icon (the left symbol in Fig. 1.7) and drag it to the diagram. If we wanted to, we could display these results in a table by choosing the table icon instead (the right symbol in Fig. 1.7).

When you create a new graph pad it will open automatically. To open a pad that had been created previously, just double-click on it to display the list of stocks, flows, and parameters for our model. Each one can be plotted. Select FISH to be plotted and, with the ≫ arrow, add it to the list of selected items. Then set the scale from 0 to 600 and check OK. You can set the scale by clicking once on the variable whose scale you wish to set and then on the arrow next to it (Fig. 1.8). Now you can select the minimum on the graph and the maximum value defines the highest point on the graph. Rerunning the model under alternative parameter settings will lead to graphs that are plotted over different ranges. Sometimes these are a bit difficult to compare with previous runs, because the scaling has changed, unless, of course, you have fixed the scale, as we suggest here.

Would you like to see the results of our model so far? We can run the model by selecting RUN from the pull-down menu. The results are shown in Fig. 1.9.

We see a graph of exponential growth of the fish population in your pond. This is what we should have expected. It is important to state beforehand what results you expect from running a model. Such speculation builds your insight into system behavior and helps you anticipate (and correct) programming errors. When the results do not meet your expectations, something is wrong and you must fix it. The error may be either in your STELLA program or your understanding of the system that you wish to model, or both.

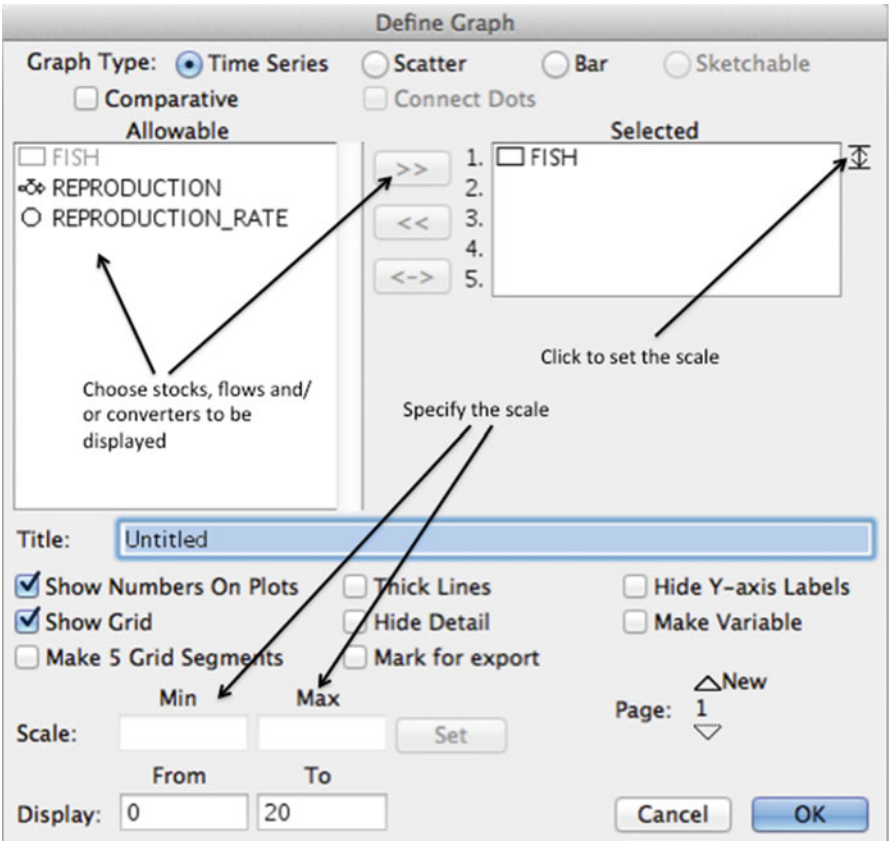
Define Graph
Graph Type:
Time Series
Scatter
Bar
Sketchable
Comparative
Connect Dots
Allowable
Selected
FISH
REPRODUCTION
REPRODUCTION_RATE
>>
<<
<->
1. FISH
2.
3.
4.
5.
Choose stocks, flows and/ or converters to be displayed
Click to set the scale
Specify the scale
Title:
Untitled
Show Numbers On Plots
Thick Lines
Hide Y-axis Labels
Show Grid
Hide Detail
Make Variable
Make 5 Grid Segments
Mark for export
Min
Max
Scale:
Set
New
Page:
1
From
To
Display:
0
20
Cancel
OK

Fig. 1.8

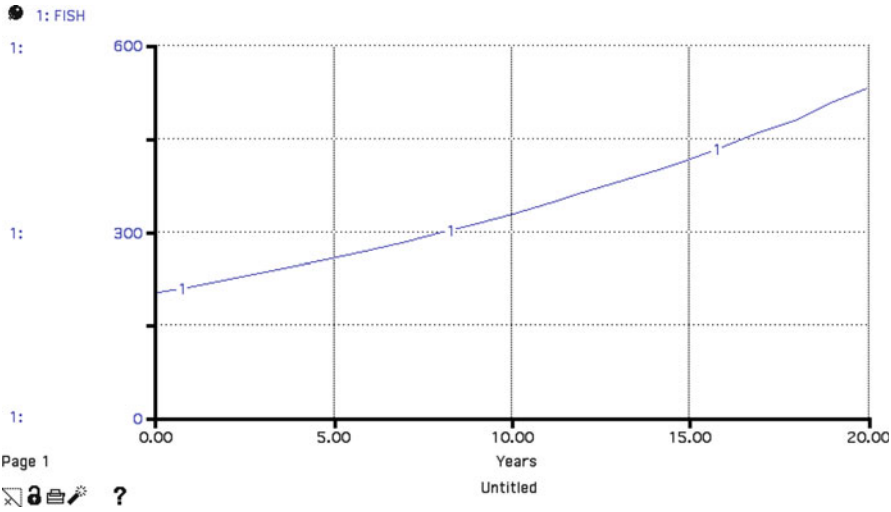
1: FISH
1:
600
1:
300
1:
0
0.00
5.00
10.00
15.00
20.00
Page 1
Years
Untitled
?

Fig. 1.9

The lower left-hand corner of the graph pad also shows several symbols—a triangle, which allows you to flip though the pages of a graph pad, assuming there is more than one graph specified in that pad; a lock, to "freeze" the output from a model run so it does not lost of the model is run again; a printer symbol to print the graph; and a dynamite stick to clear the graph. Also, below the graph is the name of the graph, which we have not specified, but you can do so by double clicking on "Untitled" and then typing in a name. At the top right-hand corner of the graph pad, you see a "pin" that can be used—by clicking on it—to affix the graph to the top layer of your diagram so that it does not go to the background when you, for example, open or close a stock or other symbol.

What do we really have here in our model? How does STELLA determine the time path of our state variable? Actually, it is not very difficult. At the beginning of each time period, starting with time = 0 years (the initial period), STELLA looks at all the components for the required calculations. The values of the state variables will probably form the basis for these calculations. Only the variable REPRODUCTION depends on the state variable FISH. The estimate the value of REPRODUCTION after the first time period, STELLA multiplies 0.05 by the value FISH (@ time = 0) or 200 (provided by the information arrows) to arrive at 10. From time = 1 to time = 2, the next DT, STELLA repeats the process and continues through the length of the model. When you plot you model results in a table you find that, for our simple fish model, STELLA calculates fractions of fish from time = 1 onward. This problem is easy to solve for example by having STELLA round the calculated number of fish—there is a built-in function that can do that—or just by re-interpreting the population size as "thousands of fish."

This process of calculating stocks form flows highlights the important role that is played by the state variable. The computer carries that information—and only that information—from one DT to the next, which is why it is defined as the variable that represents the *condition* of the system.

You can drill down in the STELLA model to see the parameters and equations that you have specified and how STELLA makes use of them. Click on the Equation tap on the far right of your STELLA diagram. The equations and parameters of your models are listed here. Note how the fish population in time period t is calculated from the population one small time step DT earlier and all the flows that occurred during a DT.

The model of the fish population dynamics is simple. So simple, in fact, we could have solved it with pencil and paper, using analytic or symbolic techniques. The model is also linear and unrealistic. So let us add a dimension of reality, and by doing so explore some of STELLA's flexibility. This may be justified by the observation that, as populations get large, mechanisms set in that influence the rate of reproduction.

To account for feedback between the size of the fish population and its rate of reproduction, an information arrow is needed to connect FISH with REPRODUCTION RATE. The connection will cause a question mark to appear in the symbol for REPRODUCTION RATE (Fig. 1.10). The previous specification is no longer correct; it now requires FISH as an input.

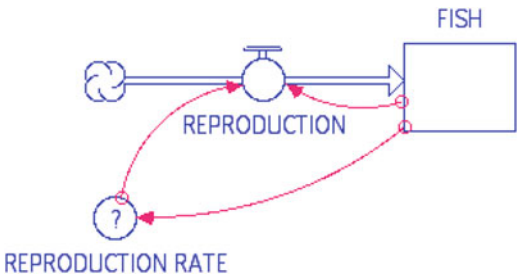

Fig. 1.10

Open REPRODUCTION RATE. STELLA alerts you that there is an unused, but required input FISH. The relationship between REPRODUCTION RATE and FISH must be specified in mathematical terms, or at least, we must make an educated guess about it. Our educated guess about the relationship between two variables can be expressed by plotting a graph that reflects the anticipated effect on variable (REPRODUCTION) will have on another (FISH). The feature we will use is called a *graphical function*.

To use a graph to delineate the extended relationship between REPRODUCTION RATE and FISH, we click on REPRODUCTION RATE to access the side-docked panel, then select FISH to make it part of the specification of the REPRODUCTION RATE, and then select the graphical function option in the lower part of the panel (see Fig. 1.11).

Set the limits on the FISH at 2 and 500; set the corresponding limits on the REPRODUCTION RATE at 0 and 0.20, to represent a change in the birth rate when the population is between 0 and 500. Here we are using arbitrary numbers for a made-up model. Finally, use the cursor to draw a curve from the maximum birth rate and population of 2 to the point of zero birth rate and population of 500.

Suppose a census of the fish population was taken at three points in time. The curve we just drew would then go through all three points. We can assume that, if a census had been taken at other times, it would show a gradual transition through all the points. Here, we use STELLA's default of 11 data points. This sketch is good enough for now. Click on OK.

Before we run the model again, let us speculate what our results will be. Think of the graph for FISH through time. Generally, it should rise, but not in a straight line. At first the rise should be steep: the initial population is only 200, so the initial birth rate should be very high. Later it will slow down. Then, the population should level off at 500, when the population density would be so great that new births tend to cease. Run the model. Figure 1.12 shows that we were right!

This problem has no analytic solution, only a numerical one. We can continue to study the sensitivity of the answer to changes in the graph and the size of DT. We are not limited to a DT of one. Generally speaking, a smaller DT leads to more accurate numerical calculation for updating state variables and, therefore, a more accurate answer. Choose Time Specs from the RUN menu to change DT.

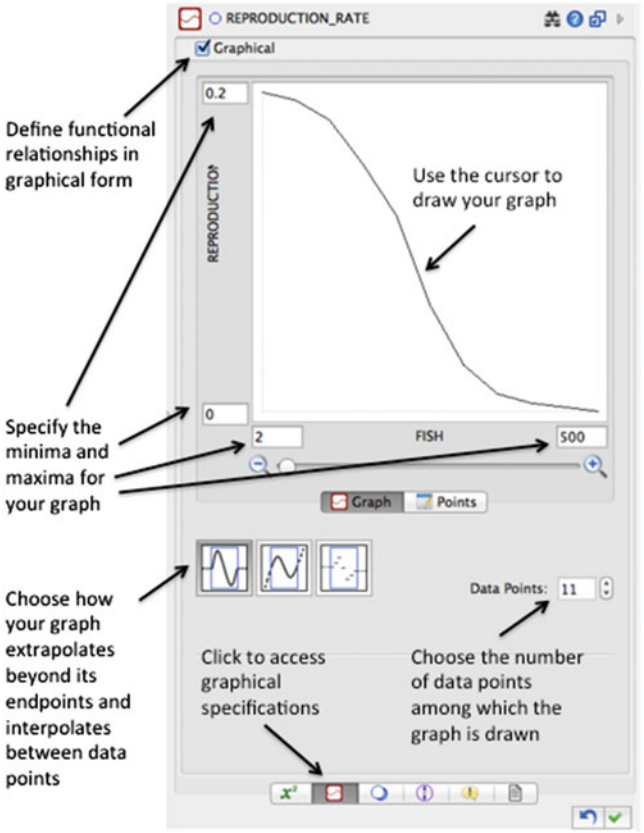

Fig. 1.11

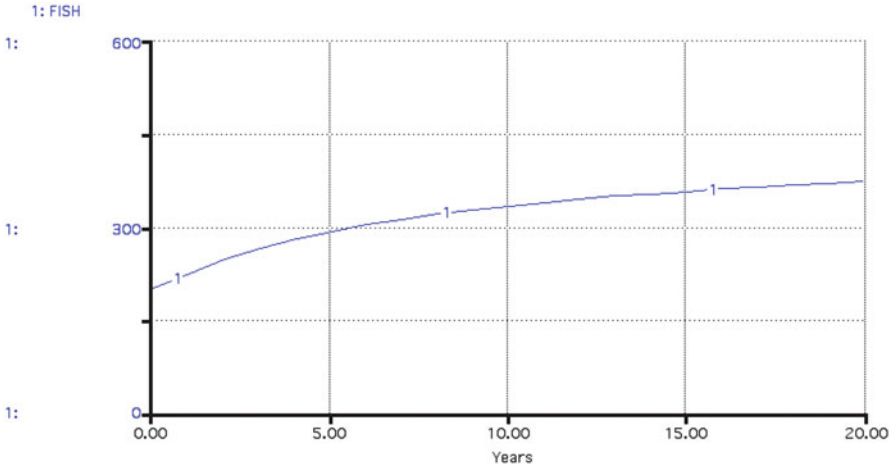

Fig. 1.12

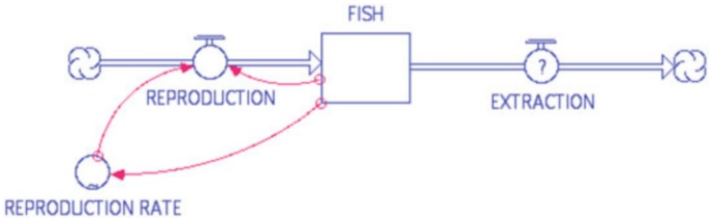

Fig. 1.13

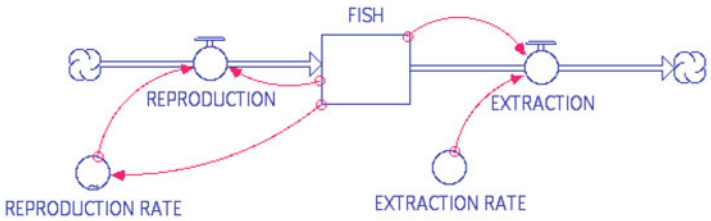

Fig. 1.14

Change DT to reflect ever-smaller periods until the change in the critical variable is within measuring tolerances. Start with a DT = 1 and reduce it to 0.5, 0.25, 0.125... for subsequent runs—each time cutting it into half of its previous value. We may also change the numerical technique used to solve the model equations. Euler's method is chosen as a default. Two other methods, Runge–Kutta-2 and Runge–Kutta-4, are available that update state variables in different ways. We will discuss these methods later.

Start with a simple model and keep it simple, especially at first. Whenever possible, compare you results against measured values. Complicate your model only when your results do not predict the available experimental data with sufficient accuracy or when your model does not yet include all the features of the real system that you wish to capture. For example, as the owner of a pond, you may want to extract fish for sale. What are the fish population dynamics if you wish to extract fish at a constant rate of 3 % per year? To find the answer to this question, define an click on the stock FISH. Click on the converter, then click onto the stock to have the converter connected to the stock, and then drag the flow from the stock to the right. Now fish disappear from the stock into a "cloud." We are not explicitly modeling where they go. Figure 1.13 shows what you should have developed thus far as your STELLA model.

Next, define a new converter called EXTRACTION RATE and set is to 0.03. Specify the outflow as:

$$\text{EXTRACTION} = \text{EXTRACTION RATE} * \text{FISH} \tag{1.3}$$

after making the appropriate connections with information arrows. Your model should now look like as in Fig. 1.14.

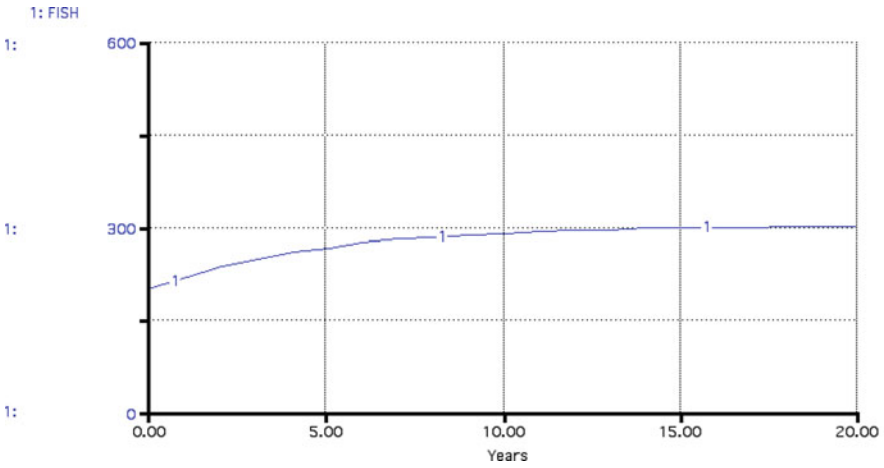

Fig. 1.15

Fig. 1.16

Run your model again. The new time-profile of the fish population in you pond is shown in Fig. 1.15.

You can easily expand this model, for example, to make the decision on EXTRACTION endogenous to your model, or introduce unforeseen outbreaks of diseases in your pond or other problems that may occur in a real world setting. When your model becomes increasingly complicated, try to keep your STELLA diagram as organized as possible, so it clearly shows the interrelationships among the model parts. A strong point of STELLA is its ability to demonstrate complicated models visually. Use the "arrow" symbol (Fig. 1.16) to move model parts around the diagram; use the "paintbrush" to change the color of icons. The "dynamite" will blast off any unnecessary parts of the model.

Be careful when you blast away information arrows. Move the dynamite to the place at which the information arrow is attached and click on that spot. If you click, for example, on the converter itself, it will disappear, together with the information arrow.

The tools we mentioned here are likely to prove very useful when you develop more complicated models and when you want to share your models and their results with others. STELLA contains many more helpful tools, which we hope you will explore and use extensively. Space limitations preclude us from describing all of STELLA's features. You will probably want to try out on your own such features, or consult the web help option accessible through the question mark in the upper right-hand corner of the side-docked panel.

Make thorough use of your model, running it over again and always checking your expectations against its results. Change the initial conditions and try running the model to its extremes. At some point you will want to perform a formal sensitivity analysis. Later, we will discuss STELLA's excellent sensitivity analysis procedures and other features to complement your modeling skills.

1.6 Principles of Modeling

Though our title of this section may seem somewhat ostentatious, we surely have learned something general about the modeling process after many years of trying. So here is our set of ten steps for the modeling process. We expect you to come back to this list once in a while as you proceed in your modeling efforts, and to challenge and refine these principles. A good set of principles should be useful to the novice and aid in speeding the process of learning to become an effective modeler.

1. Define the problem and the goals of the model. Set the questions you want the model to answer. The power of a good set of specific questions is hard to overstate. Good questions focus the mind on some aspect of the entire system in which your subsystem of interest is embedded. Appropriate generalization will come with time. Spend a lot of time defining the question(s) to be answered by your model.
2. Select the state variables, those variables that are to be the indicators of the status of the system through time. Designate the condition for non-negativity of the state variables, as appropriate. Some state variables are conserved, some are not. Identify those in your model. Keep the number of state variables as small as possible. Purposely avoid complexity in the beginning. Record the units of the state variables either in the "Units" editor available in the side-docked panel or by adding them in a comment within braces—{. . ..}—in your specification of the state variable.
3. Designate the control variables, those flow controls that will change the state variables. Note which state variables are donors and which are recipients with regard to each of the control variables. Note whether lagged effects should be included either in the controls or in the variables that compose the controls. Be sure to set these flows as biflows if appropriate. Also, note the units of the control variables as you have done for state variables, and ensure that they match with their uses in other parts of the model. Are there any useful analogies to apply here? Keep it simple at the start. Try to capture only the essential features. Put in one type of control as a representative of a class of similar controls. Add the others as needed, in step 10.
4. Select the parameters for converters. Note the units of these converters and check their consistency with other parts of the model. Ask yourself: Of what are these converters a function? Do you expect some of these variables to be *lagged or delayed* functions of some of the other variables? Only begrudgingly expand your model.

5. Check your model for compliance with any appropriate physical, economic, or other laws; for examples, the conservation of mass, energy, value; any continuity requirements. Also, check for consistency of units. Look for the possibilities of division by zero, negative volumes, or prices, etc. Use conditional statements if necessary to avoid these violations. Fully document your parameters, initial values, the units of all variables, assumptions, and equations before going on.
6. Choose the time and space horizon over which you intend to examine the dynamic behavior of the model. Choose the length of each time interval for which state variables are being updated by reference to the space over which the dynamics occur, and mainly by reference to the fastest rate of change you expect in your model. Then choose the numerical computation procedure by which flows are calculated. Set up a graph showing the most important variables and guess their variation before running the model.
7. Run the model. Are your results reasonable? Are your questions answered? Choose alternative lengths of each time interval for which state variables are updated. Choose alternative integration techniques. Explain any differences.
8. Do a sensitivity analysis of the parameters and initial values in the model. Try out these small changes singly and collectively within their reasonable extremes and see if the results in the graph still make sense. Revise the model to repair errors and anomalies.
9. Compare the results to experimental data. This may mean shutting off parts of your model to mimic a lab experiment, for example.
10. Revise the parameters, perhaps even the model structure to reflect greater complexity and to meet exceptions to the experimental results, repeating steps 1–10. Do the results of this model suggest a new set of questions? They should.

1.7 Why Model?

Now that we introduced you to modeling, the software, and general principles of modeling, it is time to step back and ask ourselves again an important question: Why, and for what purposes, do we develop models? Dynamic modeling has four possible general uses:

- First, you can experiment with models. A good model of a system enables you to compare your result to those available from the real system and to change the model components in order to see how these changes affect the rest of the system. You can experiment, form and run scenarios, and bypass the inherent risk aversion to making changes in a real system.
- Second, a good model enables prediction of the future course of a dynamic system. Some modelers want only to explain what is going on, others aspire to a higher and more difficult (and dangerous) calling: forecasting. A good model will highlight gaps in what we know about the system we are studying.

A good model will indicate the normal fluctuations in a complex system. Such variations are sometimes the cause of great alarm and much unneeded change. Conversely, observation of variation that is unexpected from real world experience could signal the need for action. A good model should be able to indicate the results of these corrective actions.

- Third, a good model can serve as a thought-organizing device. Sometimes, most of the value in modeling comes from a deeper understanding of the variables involved in a system that people are routinely struggling to control. Modeling requires that you assemble the group of experts on the various parts of the system to be understood. Each group member gains a better understanding of the system and of the skills and knowledge of their colleagues. Good modeling stimulates further questions about the system behavior and in the long run, the applicability to other systems of any newly discovered principles.
- Fourth, a good model is a growing storage device for data and ideas that the human enterprise has struggled long and hard to find and learn. Most often such data and insights are left to gather dust in printed form or reside quietly in some distant computer. An ever-developing model should capture the knowledge that we have gained, and document those lessons learned through the years about how the system actually works.

1.8 Model Confirmation

When do you know that you developed a "good" model? Giving an answer to this question often is rather difficult. By definition, all models abstract away from some aspects of reality that the modeler perceives less relevant than others. As a result, the model is a product of the modeler's perceptions. Consequently, one model is likely not the same as the models developed for the same system by other modelers who have their own, individual perceptions. Plurality of, and competition among, models is therefore required to improve our collective understanding of real-world processes. The more open and flexible the modeling approach, and the more people are engaged in the specific modeling activity, the greater the chance of important discovery.

By enclosing a selected number of system components in the model, and determining the model-system's behavior over time solely in response to the forces inside the model, the model becomes closed. Real systems, in contrast, are not closed but open, allowing for new, even unprecedented development in response to highly infrequent but dramatic changes in their environment. It is therefore not possible to completely "verify" a model by comparing model results to the behavior of the real system. There may have been extenuating circumstances that led the real system to behave differently from the model. The model itself may not have been incorrect, but just *incomplete* with regard to those circumstances. Such circumstances will always be present, precisely due to the fact that it is the modeler's goal to capture only the "essentials" of the real system and abstract away from other factors.

Consequently, complete verification of a model can only be done with regard to the consistency, or logical accuracy, of its internal structure.

Generally, there are two ways to test a model. First, one can withhold some of the basic data that was used to set up the model to determine the model parameters, data that represent the real world to the extent that it can be measured. Then the model can be used to predict the used data. For example, one might develop a model to predict the future population of the U.S. based on actual data from, say, 1900 to 1960 and then use the resulting model to predict the (known) population data from 1970 to 1990. If the "prediction" is a success, the later data can be incorporated into the model, and prediction for the next 20 or 30 years can be made with some reasonable degree of confidence. Another way to test the "goodness" of a model is to predict the condition in some heretofore unmeasured arena and then proceed to measure these variables in the field. For example, suppose that a model is being devised to predict the location of an endangered species. The model is built and calibrated on the known habitats and then applied to the rest of the likely geography to qualify these places as likely locations.

If it is not possible, by definition, to verify a model by comparing its results to the performance of the real system, how can we know that we really captured the essentials of that system in the model? We know that our model is not unique—there are always other ways to construct a model. The guide for selection: always first try to choose the simplest form. We may compare the model results to reality, not to verify but to *confirm*, and if we are unable to reproduce at least the trends observed in the real system, we know something is missing or wrong and we are forced to revise our model or check the accuracy of the data that went into specifying the model parameters and initial conditions.

Ironically, things can also become more problematic if the model results coincide well with our observations of the real system. The problem, for example, lies in the possibility that errors in the model cancel each other. Such misspecifications are difficult to detect and this is why we will start in many chapters of this book with a theory of model behavior rather than with observations of real systems. Combining theory, observations and, indeed intuition, in a disciplined way in dynamic models is especially important when we make use of easy-to-learn and easy-to-use software packages. These devices allow us to develop models that can get ahead of ourselves. At each point in the model construction process is it important to be able to justify the assumptions that are made.

Once the model is built on a theoretical base and observations, or "reasonable" initial conditions and parameter values, we let it yield the consequences of the forces built into the model. The choice of observations versus "reasonable" values is basically a choice of providing a predictive or descriptive model. For a description of the role of feedback mechanisms for system development it is frequently sufficient to concentrate on those forces and the appropriate parameter range rather than precise numerical values.

To confirm our model results we may compare them to appropriate data. The greater the number of instances in which model results and reality coincide under a variety of different scenarios, the more probable it is that the model captures the

essential features of the real system it attempts to portray. We can increase our confidence in the model further by comparing its results to other models for the same or similar systems. The latter approach is of particular interest if the model was descriptive rather than predictive. Finally, we hope that we develop, through practice in modeling, experience and new understanding of system dynamics that enable us to more easily detect problems in model specifications. This is a learning process and we hope our book makes a significant contribution.

1.9 Extending the Modeling Approach

The models developed in this book are all built with the graphical programming language STELLA. In contrast to the majority of computer languages available today, STELLA enables you to spend the majority of your time and effort on understanding and investigating the features of a dynamic system, rather than writing a program that must follow some complicated, unintuitive syntax. With its easy-to-learn and easy-to-use approach to modeling, STELLA provides us with a number of features that enhance the development of modeling skills and collaboration of modelers. First there is the knowledge-capturing aspect of STELLA. The way we employ STELLA leads not only to a program that captures essential features of a system, but also more importantly results in a process that involves the assembly of experts who contribute their specialized knowledge to a model of a system. Experts can see the way in which their knowledge is incorporated into the model because they can pick up the fundamental aspects of STELLA very quickly. They can see how their particular part of the whole is performing and judge what changes may be needed. They can see how their part of the model interacts with others and how other specialists formulate their own contribution. As a result, these experts are more likely confident that the whole model is able and accurate than if they had entrusted a programmer with their insight into the system's workings. Once engaged in this modeling process, experts will sing its praises to other scientists. This is the knowledge capturing aspect of the modeling process.

These same experts are likely to take STELLA into the depths of their own discipline. But most important, they will return to their original models and repair them to meet broader challenges than first intended. Thus models grow along with the expertise and understanding of the experts. This is the expert-capturing part of our process. It is not only based on informed consensus but it has the possibility of continuing growth of the central model.

So we avoid the cult of the central modeler on the mainframe computer, who claims to have understood and captured the meaning of the experts. We avoid producing a group of scientists who, unsure how well their knowledge has been captured, and so, in their conservative default position, deny any utility or even connection to the model.

The process is not risk free and it requires delicate organization. The scientists who do participate in the modeling endeavor will reveal an approach to their field.

There may be, no doubt will be errors and omissions in that approach, and thus, they may worry about criticism. Therefore, the rules of interaction must recognize gently their courage. The process does promise to make young scientists wiser and older scientists younger.

A general strategy for modeling with experts is to build a STELLA model of the phenomena that all expect will be needed to answer the questions posed by them at the outset. That model should cover a space and have a time step that is commensurate with the detail needed for the questions. The whole space and time needed for the model may exceed reasonable use of the desktop computer. Eventually, any modeling enterprise may become so large that the program STELLA is too cumbersome to use. However, translators exist for the final STELLA model, converting its equations into other computer code, such as C^+ or FORTRAN. Translated models can be duplicated and put into parallel modes, adjoining cells on a landscape, for example and run simultaneously on large computers. For example, in spatial ecological modeling we use STELLA to capture the expertise of a variety of life science professionals. We then electronically translate that generic model into C^+ or FORTRAN and apply it to a series of connected cells, for example as many as 120,000 in a model of the Sage Grouse [7]. The next step is to electronically initialize these now cellularized models with a specific Geographic Information System of parameters and initial conditions maps. We then run the cellularized combine on a large parallel-processing computer or a large network of paralleled workstations. In this way, the knowledge-capturing features of STELLA can be seamlessly connected to the world's most powerful computers.

The cellular or parallel approach to building dynamic spatial models with STELLA and running those models on ever more powerful computers is receiving increasing attention in landscape ecology and environmental management. An alternative, but closely related approach has been chosen by Ruth and Pieper [8] in their model of the spatial dynamics of sea level rise. The model consists of a relatively small set of interconnected cells, describing the physical processes of erosion and sediment transport. Each cell of the model is initialized with site-specific data. These cells are then moved across the landscape to create a mosaic of the entire area to be covered by the model. In its use of an iconographic programming language, its visual elements for data representation and its representation of system dynamics the model are closely related to pictorial simulation models [9] and cellular automata models [10]. The approach is flexible, computationally efficient, and typically does not require parallel-processing capabilities. Though slightly awkward, it is also possible to use STELLA to carry out object-oriented models.

It is the intention of this book to teach you how to model, not just how to use models. We have chosen STELLA toward this end because it is a very powerful tool for building dynamic models. The software also comes with an "Authoring" version that enables you to develop models for use by others who may be uninterested in the underlying structure of the model. However, since model development and understanding are the purpose of this book, we do focus here on the modeling process itself.

References

1. Redniss L (2008) The beautiful mind of Freeman Dyson. Discover, June 2008, pp 66–68
2. Penrose R (1989) The emperor's new mind: concerning computers, minds, and the laws of physics. Penguin Books/Oxford University Press, New York
3. Casti JL (1989) Alternate realities: mathematical models of nature and man. Wiley, New York
4. Botkin D (1977) Life and death in a forest: the computer as an aid to understanding. In: Hall C, Day J (eds) Ecosystem modeling in theory and practice: an introduction with case studies. Wiley, New York, p 217
5. Lightman A, Gingerich O (1992) When do anomalies begin? Science 255(5045):690–695
6. Denning PJ (1990) Modeling reality. Am Sci 78(6):495–498
7. Westervelt J, Hannon B (1993) A large-scale, dynamic spatial model of the sage grouse in a desert steppe ecosystem, Mimeo. Department of Geography, University of Illinois, Urbana
8. Ruth M, Pieper F (1994) Modeling spatial dynamics of sea level rise in a coastal area. Syst Dyn Rev 10(4):375–389
9. Câmara AS, Ferreira FC, Fialho JE, Nobre E (1991) Pictorial simulation applied to water quality modeling. Water Sci Technol 24(6):275–281
10. Toffoli T, Margolus N (1987) Cellular automata: a new environment for modeling. MIT, Cambridge

Chapter 2
Exploring Dynamic Biological Systems

Art may be said. . .to overcome, and advance nature, as in these Mechanicall disciplines.

(Wilkins, Mathematical Magick, 1648)

2.1 Simple Population Dynamics

In this chapter we will return to the concepts and ideas presented above, and explore in more detail the dynamics of seemingly simple dynamic population models. In the process of developing and exploring these models you will learn more about the features of the STELLA software. The findings of this exploration should sensitize your perception of dynamic processes and help you develop your dynamic modeling skills.

Let us begin with a simple model of a population N in a given ecosystem with carrying capacity K. The initial size of the population is $N(t=0)=10$, and the carrying capacity is $K=100=$ constant. For population sizes below the carrying capacity, N will increase. Above the carrying capacity, N will decrease. The maximum rate of increase of N is $R=0.1$, measured in individuals per individual in N per time period. A convenient specification for the change in the population size is the logistic function

$$\Delta N = R * N * \left(1 - \frac{N}{k}\right) \tag{2.1}$$

To set up the STELLA model for our investigation of the dynamics of this population, use the reservoir icon for the stock N, the flow symbol for ΔN, and

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_2,

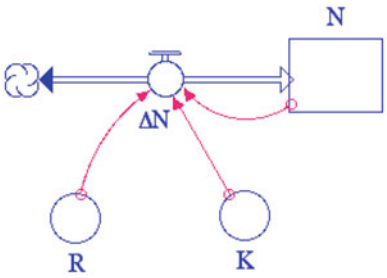

Fig. 2.1

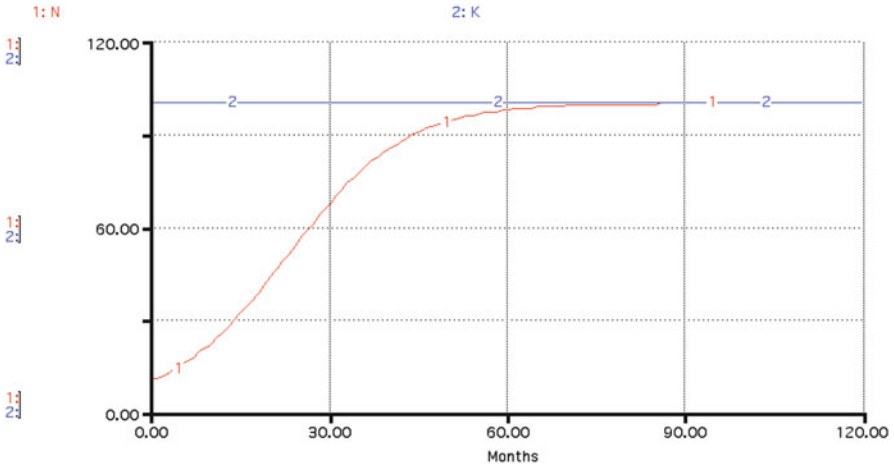

Fig. 2.2

converters for the transforming variables R and K. Specify the flow ΔN as a biflow by clicking on the biflow option in the side-docked panel. Your model should look like the one in Fig. 2.1.

Set up a graph to plot N and K over time. Specify in the Run Specs menu a DT of one and the length of the model run to extent from time period 0–120, and specify the units of time as months. Before you run the model, make an educated guess of the population size N over time. Will N reach the carrying capacity? Will the approach be exponential? Will N overshoot K? Here is what you should get when you run your model (Fig. 2.2).

The population size N asymptotically approaches K, and this approach is at first fairly rapid—as long as N is far below K—but the increase slows down as K is approached. The ratio of N/K approaches 1 as N increases, and thus $(1 - N/K)$ becomes ever smaller. As a result, ΔN approach zero, but it never quite gets there. Experiment with alternative values for R, K and initial population sizes, and observe the resulting population dynamics. Always make an educated guess about the results before you run the model.

2.2 Simple Population Model Equations

N(t) = N(t − dt) + (ΔN) ∗ dt
INIT N = 10
INFLOWS:
ΔN = R∗N∗(1−N/K)

K = 100
R = .1

2.3 Simple Population Dynamics with Varying Carrying Capacity

Let us now explore the dynamics of this system by making small changes to the parameter values. We have already modeled in the previous chapter the case in which the rate of natural increase changes as a function of the population size. Another parameter that may not be constant over time is the carrying capacity K. For example, there may be seasonal fluctuations in the physical environment that affect the resource base on which our population feeds. For simplicity, we assume that these seasonal fluctuations occur along a sinewave around the carrying capacity of 100. Click on the converter for K, then type "100+" and scroll down in the list of built-in functions to find SINWAVE to add SINWAVE to the value 100. The built-in function SINWAVE requires an amplitude and period for its specification. We set those arbitrarily to 10 and 12, respectively. You should now have

$$K = 100 + \text{SINWAVE}(10, 12) \tag{2.2}$$

which yields a carrying capacity that fluctuates between 90 and 110 over the course of a twelve-month period. The STELLA diagram (Fig. 2.3) should look as before, but the results (Fig. 2.4) are different because of the change in the specification of ΔN.

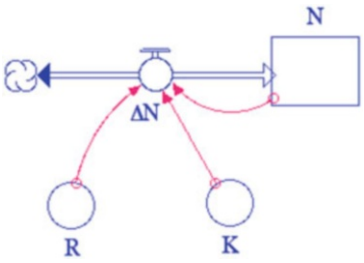

Fig. 2.3

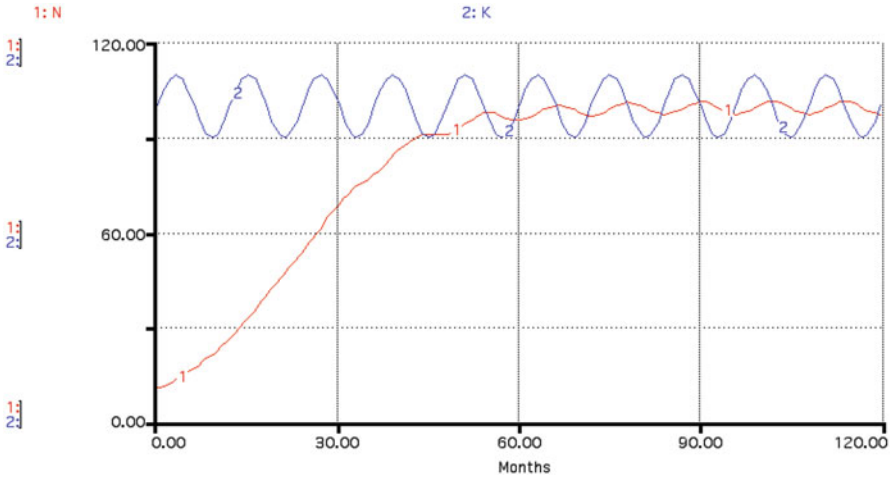

Fig. 2.4

How will this change in the carrying capacity over time affect the population size N. Since the carrying capacity has only little influence on N as long as N is small, we would expect the change in K not to alter the early sigmoidal growth phase of N. However, as N gets larger, K has an increasing influence on the subsequent changes in N. When you run the model, you should find that this is indeed the case. Look closely at the graph and recognize, however, that the changes in N and K are not exactly in sync with each other. Rather, an increase in K is not instantly matched by an increase in N. Can you explain why?

Again, explore the dynamics of the system by successively running the model for alternative specifications of R, K and initial population sizes. For example, enlarge the range over which K fluctuates over the course of a year. Alternatively, abandon the assumption that K fluctuates along a sinewave and make it a random variable. You can do so with the built-in function RANDOM which requires that you specify upper and lower bounds. For example, specify

$$\mathrm{K} = 100 + \mathrm{RANDOM}(-10, 10) \tag{2.3}$$

and K will fluctuate randomly between the values 90 and 110. Figure 2.5 shows the results of one model run with K specified as in Eq. (2.3). The time paths of this system are different from run to run because of the random number.

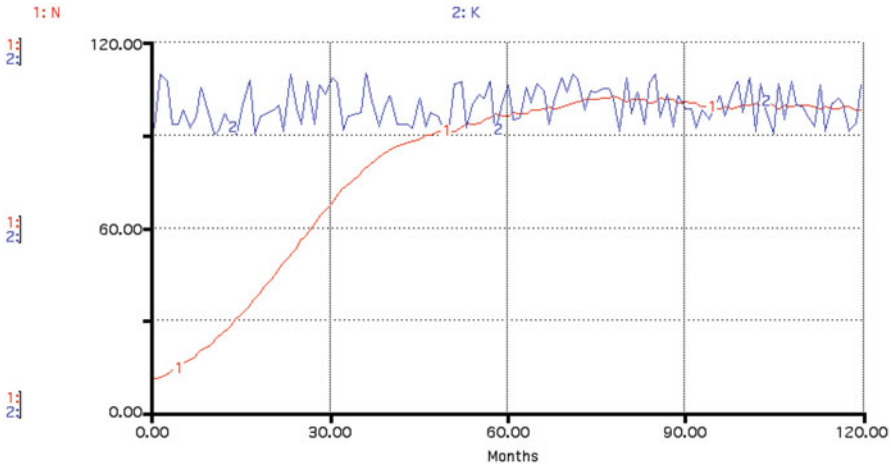

Fig. 2.5

2.4 Population with Varying Carrying Capacity Model Equations

$N(t) = N(t - dt) + (\Delta N) * dt$
INIT N = 10
INFLOWS:
$\Delta N = R*N*(1-N/K)$

K = 100 + SINWAVE(10,12)
R = .1

2.5 Sensitivity and Error Analysis with STELLA

Let us reflect for a moment on the models that we developed so far. We have hypothesized about the workings of a dynamic system—the influences on births and deaths in a population, and possible fluctuations in the maximum number of individuals of that population that can be sustained in a given environment. We have not concerned ourselves with real data describing real populations in a real environment. Rather, we were interested in the general features of such systems.

The modeling approach that we chose here is distinct from a data-driven, statistical approach. Statistical, or as they are sometimes called, empirical models are a kind of disembodied representation of some well-studied phenomenon. They have no connection to reality other than the purely mathematical. The systematical alternative, the kind we have used in this and similar books [1–3], strives to

represent as much as possible the reality of the dynamic phenomenon. Some refer to this form of modeling as the mechanical approach, but this term seems to us too wooden and likely to leave the impression that we think nature is just another mechanical process, rather like the engine of an auto. We do not think so.

In systematical modeling, we build into the representation of the phenomenon that we know actually exists—such as the birth and death processes of populations. Our systematical alternative therefore starts with an advantage over the purely statistical or empirical modeling schema. This advantage allows the systematical model to be used in more related applications than the empirical model—the systematical model is more transferable to new applications. But the empirical model does have one advantage: in the process of evaluating the data gathered about the phenomenon, the mean and standard deviations of the coefficients are found. The corresponding parameters and initial values in our models are not so elaborated. Such values are at first usually found or derived from the pertinent literature, often without a given variation.

Once the systematical model has performed to meet a general sanity test, the parameters and initial values need to be flexed to determine the sensitivity of the model results with regard to the choice of parameters and initial conditions. This process is time consuming and is usually allocated to the drudgery part of modeling. But it is essential. Just how effective is a model that responds with dramatic difference when one of its parameters is changed slightly? The point is not whether sensitivity analysis needs to be done but how can it be done efficiently? Our view is that STELLA is a very efficient tool for building the structure of the systematical model and for performing sensitivity analyses.

To conduct a sensitivity analysis, for example on the parameter R of our population model, choose "Sensi Specs…" (Sensitivity Specification) from the Run pull-down menu, and choose the parameter R—by clicking on it and selecting it—as the one for which you want to perform a sensitivity analysis. Type in the dialogue box "# of Runs" 5 to generate five sensitivity runs. Then provide start and end values for R. If you chose "Incremental" as the variation type, STELLA calculates the other values from the start and end values that you specified such that there are equal incremental changes in A from run to run. Plot the five resulting curves for N in the same graph by choosing the "Graph" option in STELLA II's Sensitivity Specs menu. Run the model with the S-Run command and observe the resulting graph. Here are the results for R varying from run to run incrementally between 0.05 and 0.15, and a carrying capacity that is specified as

$$\mathrm{K} = 100 + \mathrm{SINWAVE}(10, 12) \tag{2.4}$$

The results of this model are shown in Fig. 2.6. Here, we have created one page in our graph pad that summarizes the five consecutive runs. You can do this by clicking on the graph that already existed, then selecting a new page for that graph pad by clicking the triangle that is labeled "New", and then specifying that this should serve as a "Comparative" graph (see Fig. 2.7).

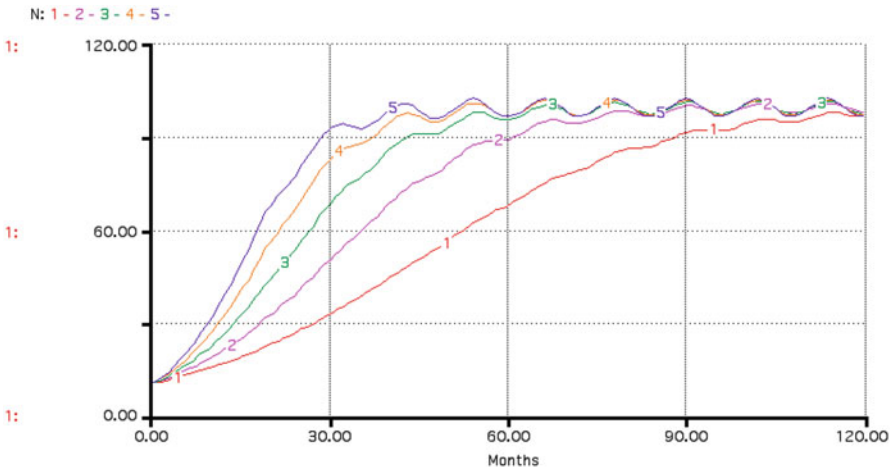
N: 1 - 2 - 3 - 4 - 5 -
1: 120.00
1: 60.00
1: 0.00
0.00
30.00
60.00
90.00
120.00
Months

Fig. 2.6

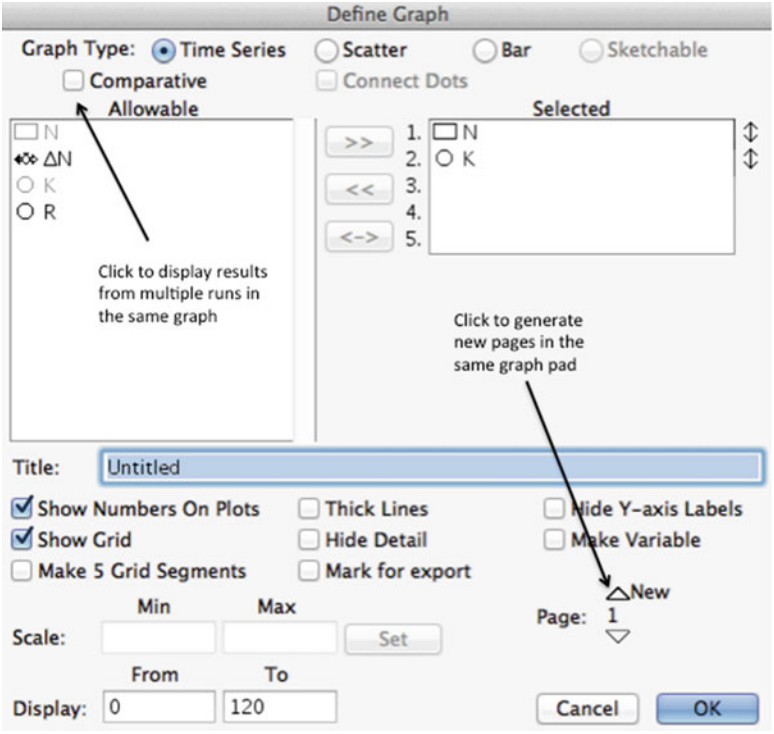
Define Graph
Graph Type: Time Series
Scatter
Bar
Sketchable
Comparative
Connect Dots
Allowable
N
ΔN
K
R
Selected
1. N
2. K
3.
4.
5.
>>
<<
<->
Click to display results from multiple runs in the same graph
Click to generate new pages in the same graph pad
Title: Untitled
Show Numbers On Plots
Thick Lines
Hide Y-axis Labels
Show Grid
Hide Detail
Make Variable
Make 5 Grid Segments
Mark for export
New
Page: 1
Min
Max
Scale:
Set
From
To
Display: 0
120
Cancel
OK

Fig. 2.7

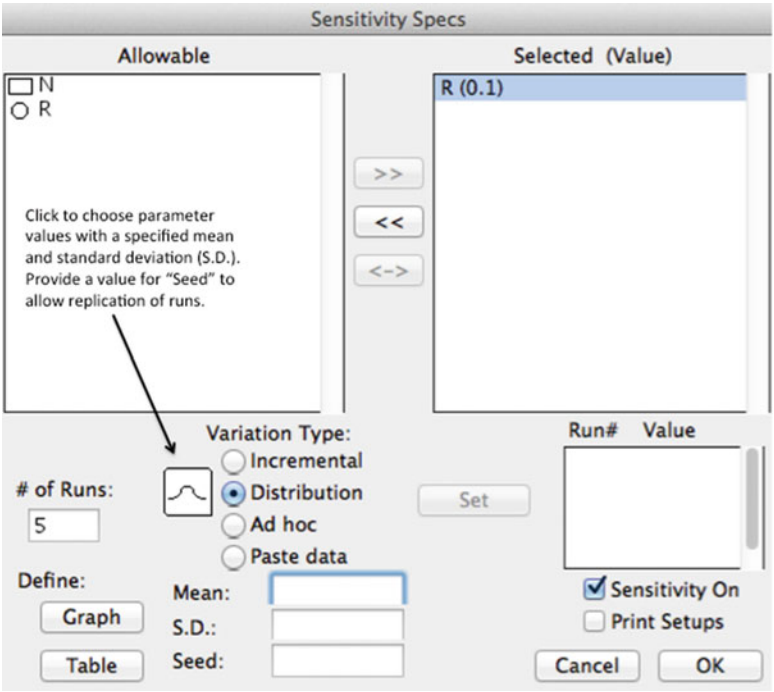

Fig. 2.8

If you wish to let values of R vary from run to run along a normal distribution with known mean and standard deviation, choose, within "Sensi Specs" the "Distribution" option instead of "Incremental". When you specify "Seed" as a positive integer, you ensure that the model will replicate a particular random number sequence in subsequent sensitivity runs. Specify 0 as the seed and a "random" seed will be selected (Fig. 2.8). If you do not wish to make use of the normal distribution of the random numbers used in the sensitivity analysis, click on the bell curve button. This curve bell-shaped button will change its appearance when you click on it, and then allow you to specify a minimum, maximum and a seed for your sensitivity analysis.

Another choice for the specification of sensitivity runs in STELLA is not to change parameters in incremental intervals or along distributions. You can specify ad hoc values for each of the consecutive runs.

You can easily specify a whole series of parameter variations in STELLA and make the hundreds of runs needed to reasonably explore the combinations for their collected sensitivity. Examination of the results can lead you to those parameter groups that can cause trouble. You might consider eliminating such a combination by a structural change in the model or by investing more effort in narrowing the real range of these parameters through extended research.

Another rather interesting approach is to change each of the parameters to a sine or cosine function varying the mean value of the parameter, similar to what we have done in the preceding section of this chapter. Each parameter is assigned its own frequency and the model is run with all parameters and initial values varying in this way. A spectral analysis can be performed on the variations in the main variable with the hope of finding certain critical frequencies, leading you directly to the parametrical culprits.

Generally though, big computer-based models create a demand for big computers as they are needed to sift through the parameter and initial value specification problem. There is no easy way around this situation. The problem is actually larger than the parameter problem discussed here. There are many sources of modeling error. Gertner et al. [4] and Gertner and Guan [5] wisely advocate the use of Error Budgets as a way of pinning down the critical areas of error sources. He and his colleagues have developed the methods of breaking down the source of error in several categories (Input Measurement, Sampling, Components of the model (sets of equations), Grouping and Computational). They are able to isolate the sources of variation in the main variable. With such information the model can be effectively revised or the data collection effort intelligently redirected.

For very large spatial dynamic models with thousands of cells, the testing problem is very great, seemingly impossibly large. But efficient testing algorithms have been and are being developed.[1]

In presenting your models and their results, you should always include the variations in the main variables of interest with changes in the critical parameters. This display reveals to the critical observer that you have a respect for the trouble that can be caused by what is still unknown about the process you study. The best thing that can happen to modelers is to have one of their models used to aid important decision-making. No good decision maker will use a model that has not been screened for its error potential.

2.6 Difference and Differential Equations

Let us more closely investigate how STELLA treats the equation we specify in our models. If we set DT = 1, then state variables are updated every full time period, such as every year, month or week. In this case, we have a model of discrete time. As DT is lowered, we still have a discrete time model, but more closely approach the case of continuous time. The model of this section illustrates the differences between the two cases.

[1] For a review of this approach and the general strategy of developing, sensitivity testing and using these large spatial models, see: http://ice.gis.uiuc.edu, generally and: http://ice.gis.uiuc.edu/TortModel/tortoise.html, specifically for the error budgeting process.

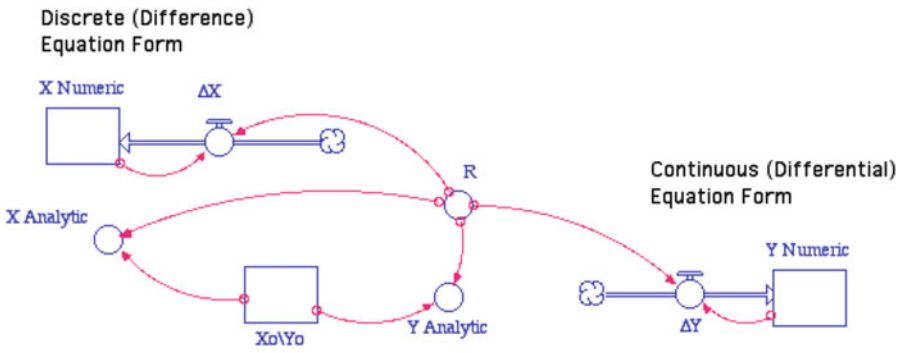

Fig. 2.9

Start with the following difference equation:

$$X(t+1) = R * X(t) \quad t = 1, 2, 3, 4 \ldots \tag{2.5}$$

The analytic solution to this equation is:

$$X(t) = Xo * R\hat{}t \quad t = 1, 2, 3, 4 \ldots \tag{2.6}$$

In STELLA, this difference equation is

$$\Delta X = X(t+1) - X(t) = (R-1) * X(t) \tag{2.7}$$

which yields what we term Y Numeric in the model of Fig. 2.9.

The continuous form version of this phenomenon is:

$$\Delta Y = R * Y, \tag{2.8}$$

yielding Y Numeric.

The analytic solution to this continuous time equation is:

$$Y(t) = Yo * EXP(R * t) = Y\ Analytic. \tag{2.9}$$

X Numeric and X Analytic are the same if DT = 1. As DT approaches 0, these equations drift apart. But the analytic solution to the difference equation is good only for DT = 1 (Fig. 2.10). Conversely, when DT = 1, the Y Numeric and Y Analytic are far apart. Run the model with a DT = 1/1024, and you will find that numeric and analytical equations converge, as of course they should.

Figure 2.11 shows how much the two numeric solutions agree at DT = 1. Thus there is a substantial difference between the difference and differential (discrete vs. continuous) equations when they result in exponential solutions.

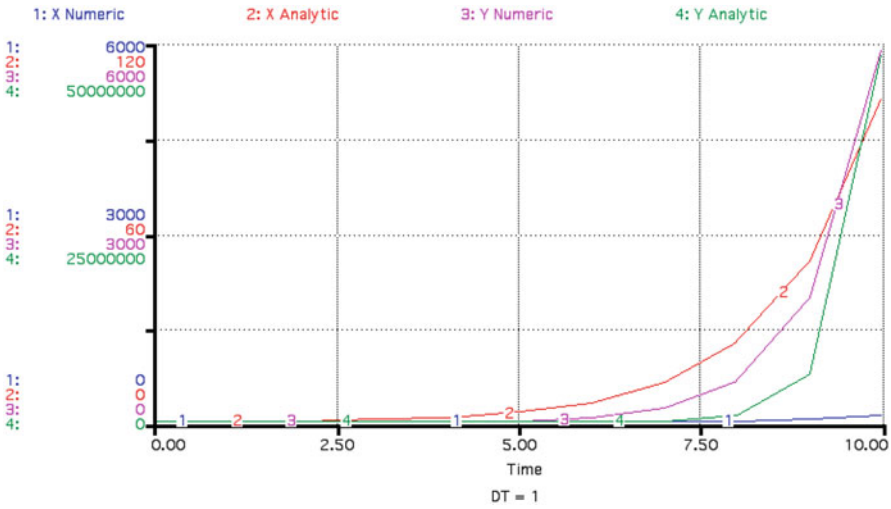

Fig. 2.10

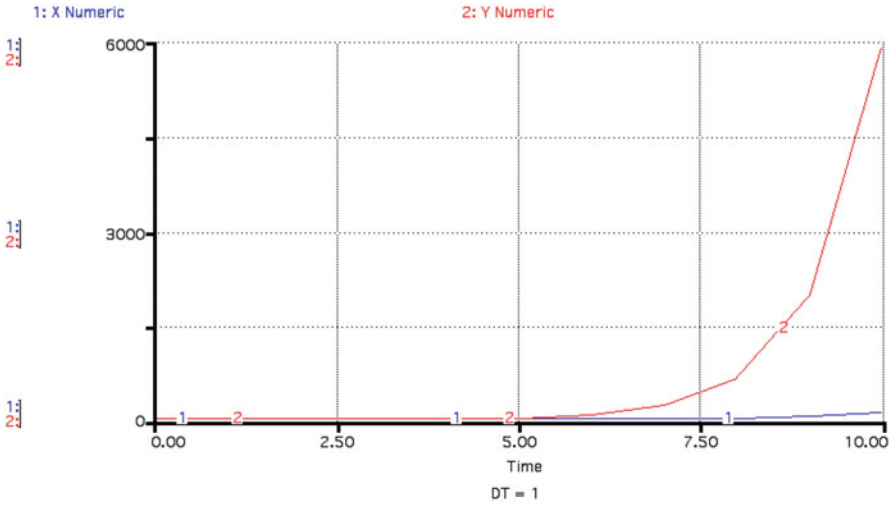

Fig. 2.11

2.7 Difference and Differential Equation Model Equations

$$\text{Xo\Yo}(t) = \text{Xo\Yo}(t - dt)$$
$$\text{INIT Xo\Yo} = 0.1$$
$$\text{X\_Numeric}(t) = \text{X\_Numeric}(t - dt) + (\Delta X) * dt$$
$$\text{INIT X\_Numeric} = \text{Xo\Yo}$$

INFLOWS:
$\Delta X = (R-1)*X\_Numeric$

$Y\_Numeric(t) = Y\_Numeric(t - dt) + (\Delta Y) * dt$
INIT Y_Numeric = Xo\Yo
INFLOWS:
$\Delta Y = R*Y\_Numeric$

$R = 2$
X_Analytic = Xo\Yo*R^time
Y_Analytic = Xo\Yo*EXP(R*time)

References

1. Hannon B, Ruth M (2001) Dynamic modeling, 2nd edn. Springer, New York
2. Hannon B, Ruth M (2009) Dynamic modeling of diseases and pests. Springer, New York
3. Ruth M, Hannon B (2012) Modeling dynamic economic systems, 2nd edn. Springer, New York
4. Gertner G, Cao X, Zhu H (1991) A quality assessment of a Weibull based growth projection system. For Ecol Manage 71(3):235–250
5. Gertner G, Guan B (1991) Using an error budget to evaluate the importance of component models within a large-scale simulation model. In: Proceedings of conference on mathematical modeling of forest ecosystems. J.D. Verlag, Frankfurt am Main, pp 62–74

Chapter 3
Risky Population

I know that history at all times draws strangest consequence from remotest cause.

(T.S. Eliot, Murder in the Cathedral, Part I, 1935)

3.1 Risky Population Model

In the previous chapter we have seen how to model simple deterministic and random processes that influence population dynamics. We emphasized the need to thoroughly test your models before you move on and expand them. This chapter provides a novel expansion to the traditional model of population dynamics. Other expansions follow in the next chapters. Each of those expansions is kept to a minimum complexity yet the resulting dynamics can be rather surprising. A central utility of this model is that it produces what could be called a carrying capacity of an environment as reflected in the birth, death, and behavioral characteristics of the population, as determined by that environment. Simple population model that claim to represent resource limits often do it by simply specifying the carrying capacity as a model input parameter [such as K in Eq. (2.1) of Chap. 2]. Here the carrying capacity is derived.

For the following model assume that a population grows exponentially by virtue of a birth rate and dies according to a death rate. If the birth rate is 7 % and the death rate is 4 %, then the population grows exponentially at the rate of 3 %. That is a simple matter and we have addressed this model in our most elementary examples of the previous chapters.

But we all know that, as the population size (or density) grows and no migration can take place, the chance of a sudden unanticipated rise in the death rate increases.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_3,

Perhaps the calamity is due to an infectious disease or, in the case of some human populations, to war. We leave out natural causes like earthquake and meteor impacts because these disasters are not brought on by the population itself. Likewise as some argue, a larger population causes an inventive focus to alleviate some of the human dilemma. We need to express the death rate in such a way that both the negative and the beneficial effects of rising population can be felt. The question is: when the birth rate is higher than the nominal death rate, what are the population dynamics when higher populations can have the effect of increasing and at other times decreasing the death rate? The answer is rather surprising.

Let us set the birth rate to a constant rate of 7 %. To mimic unanticipated changes in the death rate, we introduce a nominal death rate, NOMINAL DR, which we set to

$$\text{NOMINAL DR} = .04 \tag{3.1}$$

We make use of this nominal death rate as the mean value in a normal distribution, DR DISTRIBUTION. To specify a normal distribution with a mean of nominal death rate we make use of the built-in function NORMAL. We must specify the mean and standard deviation as its arguments. We define

$$\text{DR DISTRIBUTION} = \text{NORMAL}(\text{NOMINAL DR}, 0.005 * \text{POPULATION}) \tag{3.2}$$

with an arbitrarily set standard deviation of 0.005∗POPULATION. Thus, as the POPULATION increases, so does the standard deviation around the mean NOMINAL DR.

Next we define a death rate signal, DR DIST CONTROL that constrains the normally distributed DR DISTRIBUTION between 0.01 and 1:

$$\begin{aligned}\text{DR DIST CONTROL} = {} & \text{IF}(\text{DR DISTRIBUTION} \geq 0.01) \ \text{AND} \\ & (\text{DR DISTRIBUTION} \leq 1) \\ & \text{THEN DR DISTRIBUTION ELSE } 0.01\end{aligned} \tag{3.3}$$

We assume that the absolute minimum of the death rate is 0.01.

Finally, the DEATH RATE itself is set up to allow the following three test cases:

(a) The DR DIST CONTROL is defined in Eq. (3.3) above,
(b) A case where the standard deviation is allowed to only increase the death rate, and
(c) An exponentially declining nominal death rate whose standard deviation (0.005∗POPULATION) is allowed to only increase the death rate.

For the first case, we define

$$\text{DEATH RATE} = \text{DR DIST CONTROL} \tag{3.4}$$

Consequently, the death rate varies with normal distribution from 0.01 to 1 in every time period (e.g., year, decade, generation) with the nominal death rate

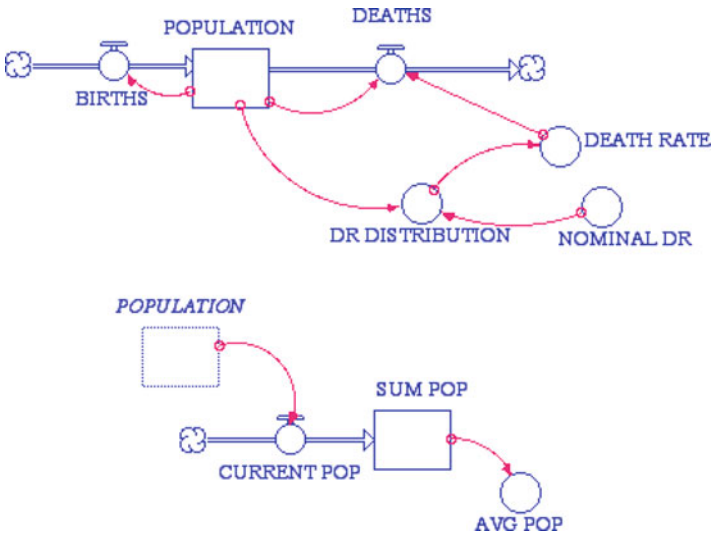

Fig. 3.1

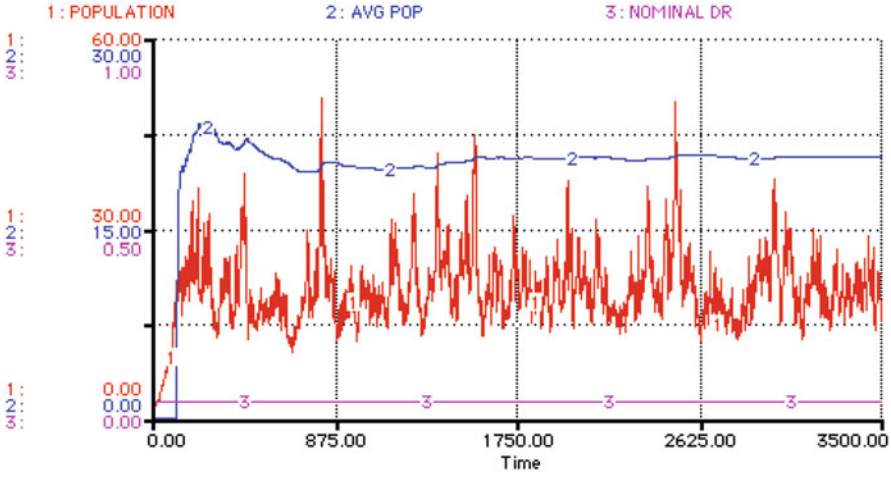

Fig. 3.2

(here 4 %) as the mean of this distribution. The STELLA diagram is shown in Fig. 3.1. It contains a module that calculates the average population size AVG POP over time by summing the population over time and dividing by time.

The results of this model are shown in Fig. 3.2.

The remarkable result is that the population reaches a somewhat steady condition—it is not expanding off to infinity at a net 3 % growth rate as implied by the net birth rate of 7 %. At first the population grows exponentially but it then

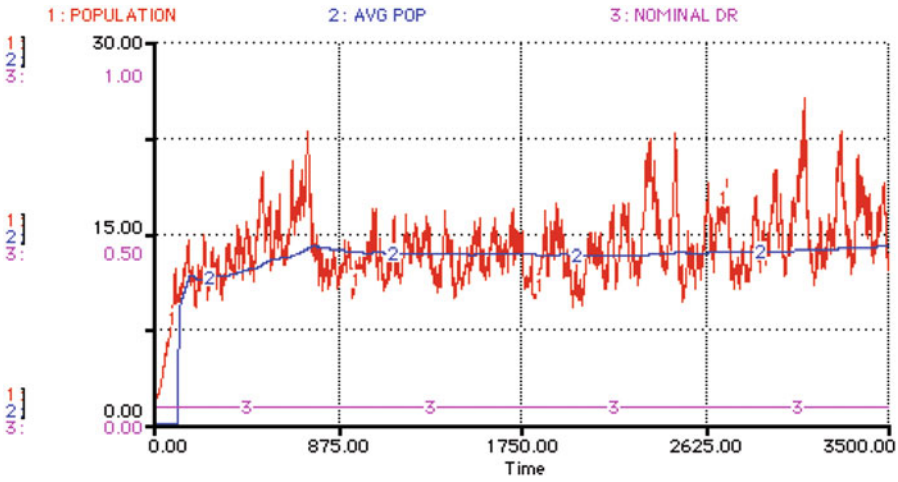

Fig. 3.3

hits a kind of limit shown by the average population after about 1,500 time units. The population becomes large enough to be constrained by the sudden switches in its death brought on by the sheer size of the population, even allowing that many of these switches are death rate-reducing.

How do the results of our model change if the DEATH RATE is allowed only to increase above the NOMINAL DR (up to 1.0), from period to period? This is the second test case described above. To investigate this case we change the definition of the DEATH RATE in Eq. (3.4) to

$$\text{DEATH RATE} = \big(\text{IF DR DIST CONTROL} > \text{NOMINAL DR THEN DR DIST CONTROL ELSE NOMINAL DR}\big) \tag{3.5}$$

To specify the model in this way, note that you will need a connector from NOMINAL DR to DEATH RATE. Here we find a lower mean population size of not quite 14, down from the previous model run of about 20 (Fig. 3.3). Can you explain why?

The results above were all derived for a fixed nominal death rate. Yet, the nominal death rate may slowly decline over time as the population grows. This is certainly the case for the human population that, through advancements in medicine and political treaties, has significantly reduced the death rate in parts of the world. How do our results change if we have on the one hand a decline in the nominal death rate and on the other hand an increase in the standard deviation above the mean DR DISTRIBUTION? We model the decline in the nominal death rate as

$$\text{NOMINAL DR} = \text{EXP}(-.01 * \text{TIME}) * .03 + .01 \tag{3.6}$$

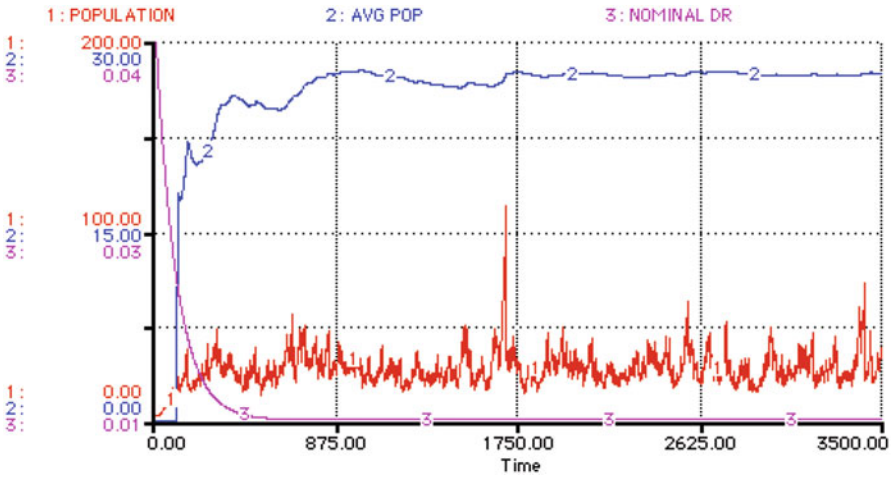

Fig. 3.4

where EXP is Euler's number, and TIME is a built-in function that takes on the same value as the current period of the model run. With this specification, the nominal death rate exponentially declines from 4 to 1 %. This is an example for our third case mentioned above, and the results it generates are shown in Fig. 3.4. Now, the mean population size is approximately 27.

We note that at first the population rises in the expected exponential way and then the effect of the standard deviation takes over, causing an average population about 26. None of these populations continue to grow. After about t = 2,000, a mean value is achieved that holds for the rest of the run, in all cases although the first and third cases clearly show a greater variation.

Can you develop any more scenarios? Should the death rate be allowed to vary so greatly from period to period? Perhaps the death rate should be a state variable and thus change more slowly. Can you set up such a model? Should this distribution be normal? Is it a bit unreal to cut the normal distribution off at 0.01 on the low side and 1.0 on the high side? How about a normal distribution that has different standard deviations for the different sides of the mean? What about the idea of making the standard deviation a nonlinear function of the population size? What about the possibility of a delay of death rate decreases with no delay of death rate decreases? Try out some of these modifications of the model, make an educated guess before you run them, then explain your results.

Now that we have seen some of the effects of randomness on population dynamics, we will return to deterministic models—models that yield the same results from run to run. The absence of randomness, however, not necessarily means that we will be able to make precise forecasts of a system's behavior once we know its history and current state. Rather, unforeseen, chaotic events may occur. This is the topic of the following chapter.

3.2 Risky Population Model Equations

POPULATION(t) = POPULATION(t − dt) + (BIRTHS − DEATHS) ∗ dt
INIT POPULATION = 2

INFLOWS:
BIRTHS = .07∗POPULATION
OUTFLOWS:
DEATHS = DEATH_RATE∗POPULATION
SUM_POP(t) = SUM_POP(t − dt) + (CURRENT_POP) ∗ dt
INIT SUM_POP = 0

INFLOWS:
CURRENT_POP = IF TIME > 100 THEN POPULATION ELSE 0
AVG_POP = IF TIME ≠ 100 then SUM_POP/(time−100) else 0
DEATH_RATE = (IF DR_DIST_CONTROL > NOMINAL_DR THEN DR_DIST_CONTROL ELSE NOMINAL_DR)∗1 +0∗DR_DIST_CONTROL +0∗NOMINAL_DR
DR_DISTRIBUTION = NORMAL(NOMINAL_DR,0.005∗POPULATION)
DR_DIST_CONTROL = IF (DR_DISTRIBUTION ≥ 0.01) AND (DR_DISTRIBUTION ≤ 1) THEN DR_DISTRIBUTION ELSE 0.01
NOMINAL_DR = (EXP(−.01∗TIME)∗.03 + .01)∗1 + .04∗0

Chapter 4
Steady State, Oscillation, and Chaos in Population Dynamics

And then so slight, so delicate is death
That there's but the end of a leaf's fall
A moment of no consequence at all.

(Mark Swann)

4.1 The Emergence of Chaos in Population Models

Let us return to the simple population models of Chap. 2, in the absence of randomness, and explore the behavior of a very basic deterministic model as a parameter value gets pushed outside the realm that is typically considered in these models. Denote the size of the population in time period t as N(t) and the net change in the population size during that period as ΔN. The exogenous parameter influencing the net flow is R. The net flow ΔN updates the stock N:

$$\Delta N = N(t + DT) - N(t). \tag{4.1}$$

The "reproductive rule" in this model is

$$N(t + DT) = R * N(t) * (1 - N(t)), \tag{4.2}$$

and consequently

$$\Delta N = R * N(t) * (1 - N(t)) - N(t). \tag{4.3}$$

Compare this equation to Eq. (2.1) of Chap. 2 and describe the differences. Also see the discussion of discrete versus continuous STELLA flow equation in Chap. 2.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_4,

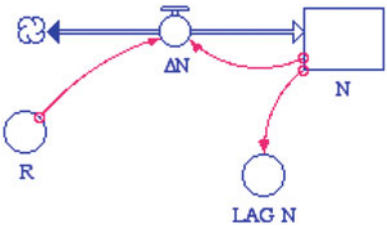

Fig. 4.1

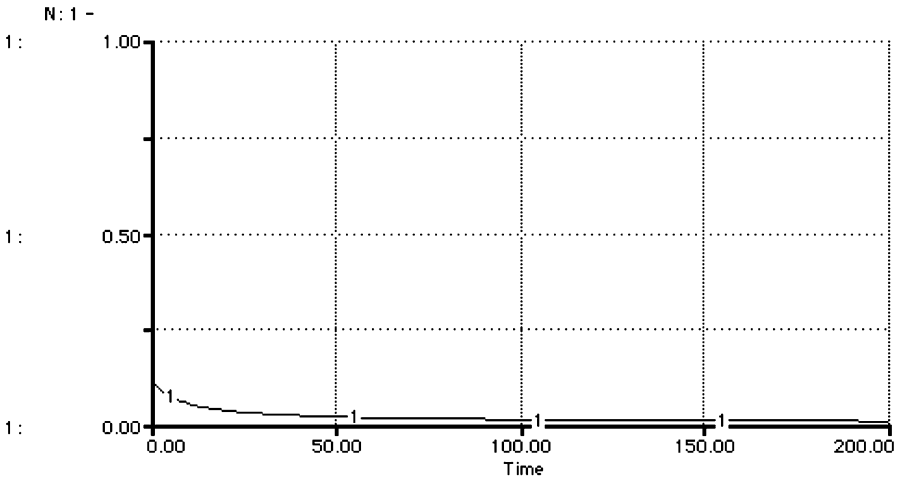

Fig. 4.2

The STELLA model shown in the following diagram has as its main component the ΔN equation updating the stock N at each period of time. We also calculate the stock N delayed by DT as LAG N. To calculate this lagged value, we make use of the built-in function DELAY, which requires as its input the variable of which a delay should be calculated—in our case N—the lag length—DT—and initial value. If you do not specify an initial value, STELLA will assume the initial value of the delayed variable is zero. Here, we specify

$$\text{LAG N} = \text{DELAY}(\text{N}, \text{DT}) \tag{4.4}$$

The STELLA model is shown in the diagram of Fig. 4.1. Note that the control on N is specified as a bi-flow, allowing additions into and subtractions from the stock.

Set up the model with an initial value for $\text{N} = 0.1$, $\text{DT} = 1$, and $\text{R} = 1$. Make an educated guess before you run the model. Figure 4.2 shows what you should get. The population size declines along the logistic curve.

Next increase R for subsequent runs to 2, then 3. You can do that with STELLA's sensitivity methods. Again make a guess before you run the model. The results are plotted in Fig. 4.3.

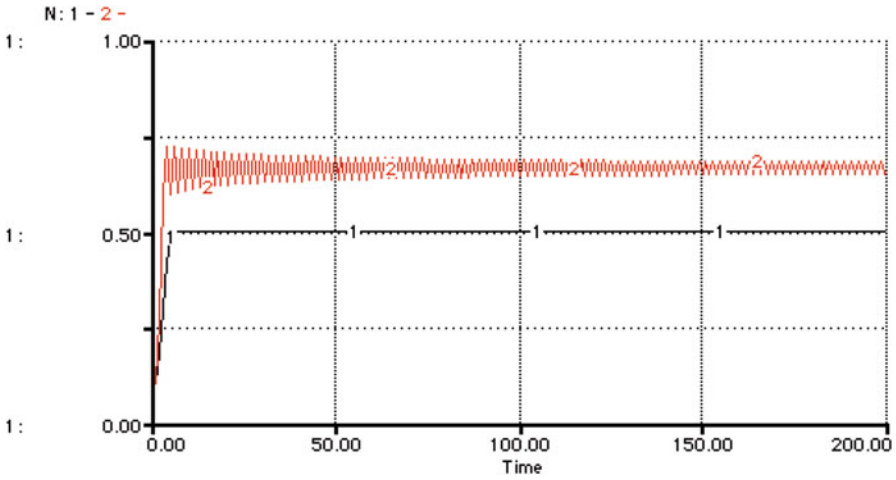

Fig. 4.3

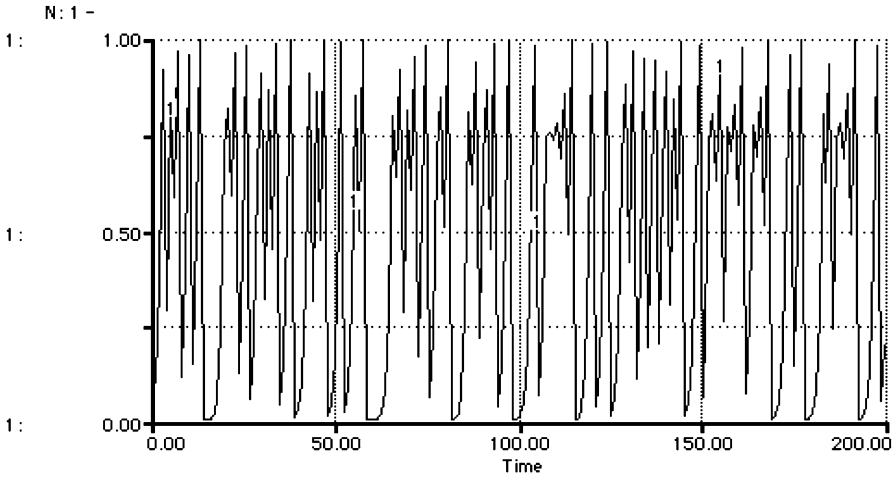

Fig. 4.4

The first of these runs generates a steady-state population of N = 0.5, and the second run yields a damped oscillation. With R = 2 we have a situation in which the population size can be thought of overshooting some carrying capacity, then gets correct and falls below that carrying capacity, only to overshoot again—albeit to a smaller extent—in the next period.

Increase R to 4. You should find that the population has left its regular pattern and becomes chaotic—the curve never repeats itself (Fig. 4.4). Pause the model halfway through its run and make a guess on the future path. Can you predict where it is going, solely based on your observation of its past behavior and current position?

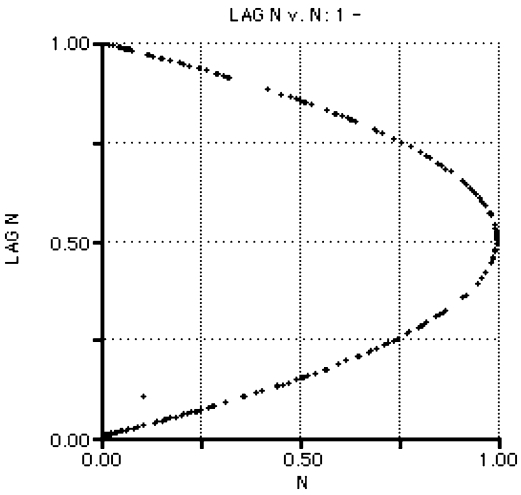

Fig. 4.5

Such prediction is impossible. Run the model over and over again, and you will find that always the same path is chosen. The system that we are dealing with here is not random, it is deterministic. Yet, prediction from past and current states is impossible. Recognize also that if you change even very slightly the initial conditions, e.g., $N = 0.101$ instead of $N = 0.1$, a very different path emerges. This should illustrate the sensitivity of nonlinear dynamic systems to initial conditions, and sensitize you to the limitations of real data—which always comes with measurement errors—in forecasting a system's behavior.

Even though there is seemingly no regularity in the system's behavior, all values lie within a well-defined range. Generate a scatter plot—a diagram of the system's phase space—by plotting LAG N against N, and observe the results. To set up the scatter plot, create a graph and choose "Scatter" as the Graph Type. Here are the results (Fig. 4.5).

Can you find the value for R at which chaos seems to begin? Now lower the time step to $DT = 0.5$ and find the value for R at which chaos begins again. Keep shortening DT and you will find that there is a relation between the size of DT and the smallest R necessary to produce chaotic behavior:

DT	R necessary for chaos
1	3.58
0.5	6.12
0.25	11.29
0.125	21.56
0.0625	42.13
0.03125	83.24

A pattern emerges. When DT is halved, R is doubled and lessened by one. That is, if $DT(n+1) = 0.5 * DT(n)$ then $R(n+1) = 2 * R(n) - 1$. A function R(DT) to calculate the critical R is

Fig. 4.6

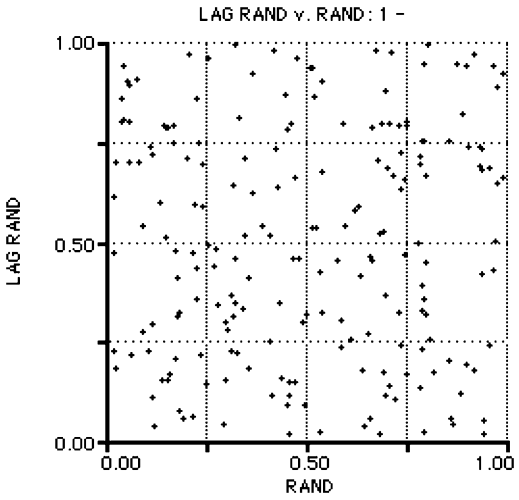

Fig. 4.7

$$R(DT) = (4.57/DT) - [1 + 2 + 4 + 8 + \cdots 1/DT] \tag{4.5}$$

or

$$R(DT) = (R(1) + 1)/DT - (1 + 2 + 4 + 8 + \cdots 1/DT), \tag{4.6}$$

the boundary between chaos and finitely numbered solutions for the spectrum of discrete steps.

This model shows you that the R for DT = 0 is infinity. This result is correct since chaos is typically not noticed on the continuous level. Chaos occurs on the continuous level only if you are stuck with a specific DT in your particular problem, and the parameters lie within the critical range. For a full discussion of this and other versions of chaos, see Jenson [1].

Compare the chaotic time paths of your model to a truly random number. We defined such a number in the model as RAND and calculated its delayed value (Fig. 4.6).

The random number, plotted against its delayed value, is shown in Fig. 4.7. Re-run the model several times and see how the graph changes. The difference between chaotic—but deterministic—behavior and random behavior should become apparent.

At this point we should ask if chaos occurs in nature. We find that indeed it does. Water drips from a faucet chaotically, heart beats and brain wave variations show chaos. Both living and non-living systems seem to show chaos. Why? To what

advantage is such a result to these systems? Kaufman [2], among others, has proposed that all systems seem to evolve toward higher and higher efficiencies of operation. Many systems are so highly disturbed by variations in their environment that their efficiencies are not ever very high. However, if these disturbances can be held to a minimum, then the evolution of the system becomes more complete, more efficient but closer to the border of chaotic behavior. Earthquakes and avalanches are examples of energy storing systems that continuously redistribute the incoming stresses more and more efficiently until a breaking point is reached and the border to chaos is opened. Does this mean that the brain and the heart have somehow evolved close to some maximum efficiency for such organisms? We don't know the answer to this question. We do know that the scale of measurement here matters. For example, if we were to watch the pattern on a patch of natural forest over many centuries, we would see the rise and sharp fall of the biomass levels, unpredictably. Forest fires and insects find ample hosts in such forest patches once they have developed a large amount of dry biomass bound up in relatively few species. The patch evolves or succeeds to greater and greater efficiency of light energy conversion by getting larger and fewer species. But the patch also becomes more vulnerable to fire and pests, and eventually collapses. Yet if we look at the total biomass on a large collection of such biomasses, whose collapses are not synchronized, this total biomass remains relatively constant. Thus chaotic-like behavior in the small is not seen in the large. Could this mean that natural systems have "found" chaos in their search for greater efficiencies and have "learned" to stagger the chaotic events, allowing faster rebound and large-scale stability? We don't know the answers here either, but we think the implications are fascinating. We will return to these questions with our models on self-organization and catastrophe, presented in Part VII of this book.

4.2 Chaotic Population Model Equations

N(t) = N(t − dt) + (ΔN) ∗ dt
INIT N = 0.1

INFLOWS:
ΔN = R∗N∗(1 − N) − N {X_t plus one − X = delta X}
LAG_N = DELAY(N, DT)
LAG_RAND = DELAY(RAND, DT)
R = 1
RAND = RANDOM(0,1)

4.3 Simple Oscillator

The model of the previous section assumed that changes in population size occur instantaneously in response to the current population. Alternatively, we may assume that those changes are a function of the population size one time period delayed.

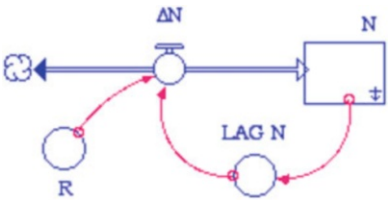

Fig. 4.8

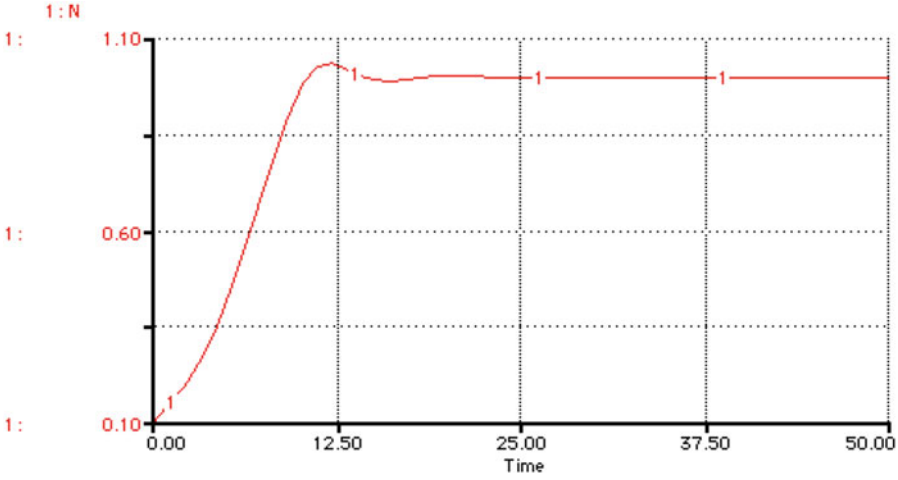

Fig. 4.9

Oscillations can occur in simple first-order differential equations provided they are nonlinear and contain lags. Let us define the differential equation that guides the change in population size as

$$\Delta N = R * LAG\ N * (1 - LAG\ N) \tag{4.7}$$

with

$$LAG\ N = DELAY(N, 1) \tag{4.8}$$

where DELAY is the built-in function that generates a delayed value of its argument—in our case the state variable N—over a set time frame—in our case 1 full time period.

The STELLA model is shown in Fig. 4.8.

Set $R = 0.5$ and $DT = 1$ and the model will generate population dynamics that oscillate but quickly settle down to a steady-state level. The results of the model are shown in Figs. 4.9 and 4.10. In steady state, LAG $N = N$, and thus the graph in phase-space collapses to a point.

Now increase R to 1, make an educated guess of the results of the model, and run it. Figures 4.11 and 4.12 show what you should get. Did you expect this result and

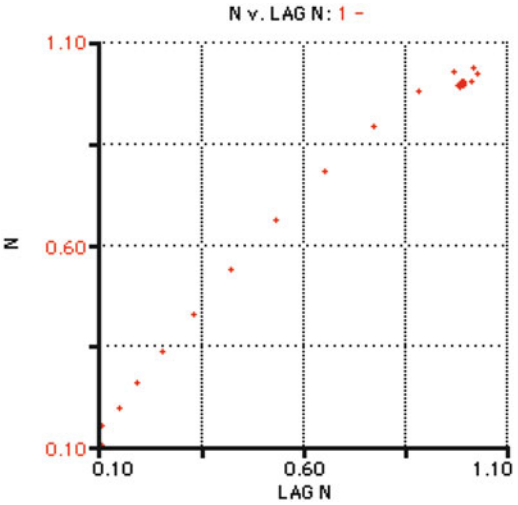

Fig. 4.10

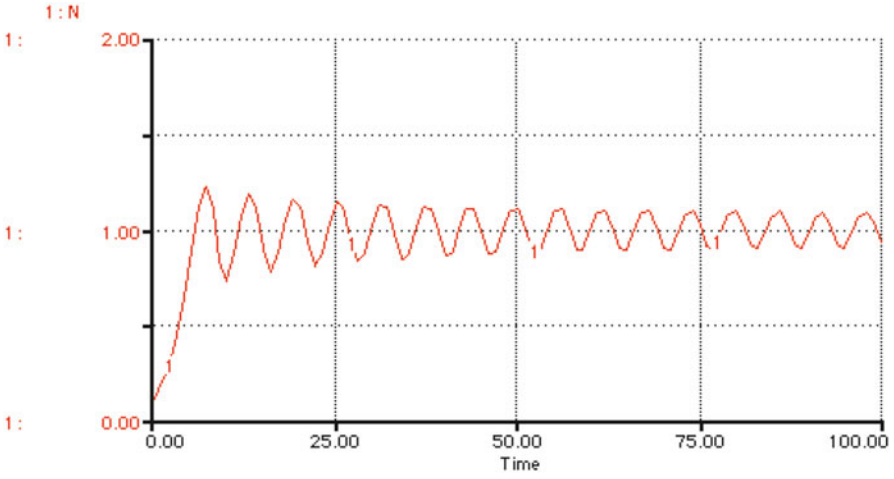

Fig. 4.11

can you explain it? Here a pattern emerges and yet is it chaos? Note how the arcs of the limit curve fill in with a definite pattern. This pattern of filling in arcs is due to the lagged N.

Keep increasing R for subsequent runs and observe the results. The plot in Fig. 4.13 is for R = 1.3.

If the DT is changed to 0.5, and R set to 1, you will find again an oscillation—shown in Figs. 4.14 and 4.15. But if you increase R to 1.3, you will not find chaos.

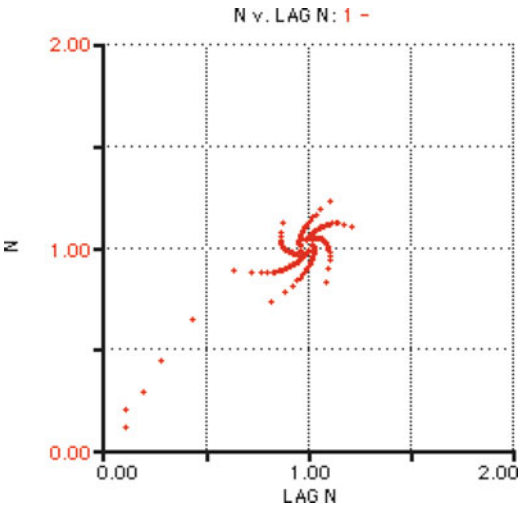

Fig. 4.12

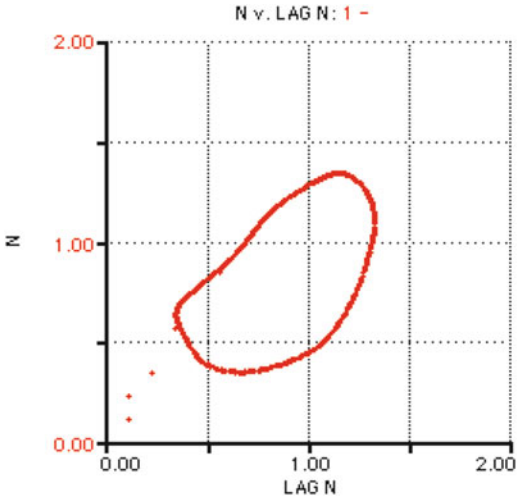

Fig. 4.13

Rather, the number of limbs increases from 6 to 10 and the limit loop becomes smaller. See Figs. 4.16 and 4.17 for this case. What is the origin of these limbs?

Can you find the value of R at which chaos emerges? If the lag time is increased to two, the model becomes unstable. Could you then stabilize it somehow?

Oscillatory behavior has been observed in chemical systems. We will model one of the most prominent chemical oscillators—the Brusselator—in Chap. 10. It has far-reaching implications for the understanding of real-world systems, and we will discuss these implications in that later chapter.

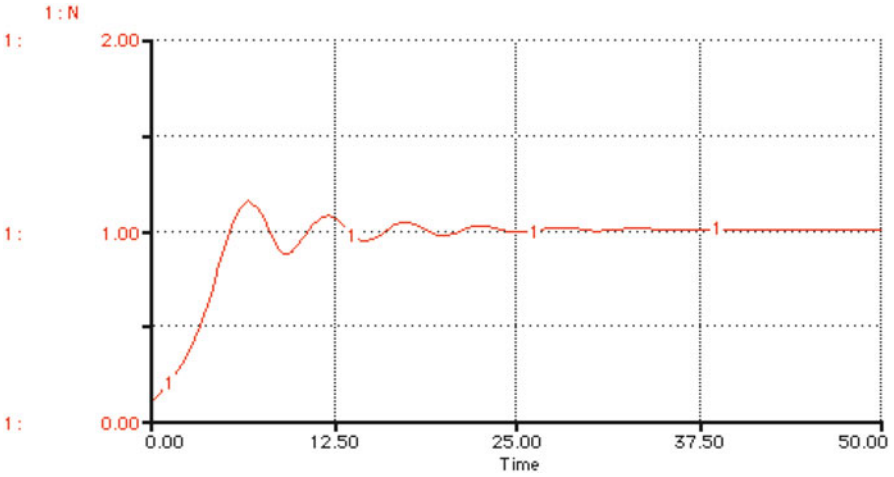

Fig. 4.14

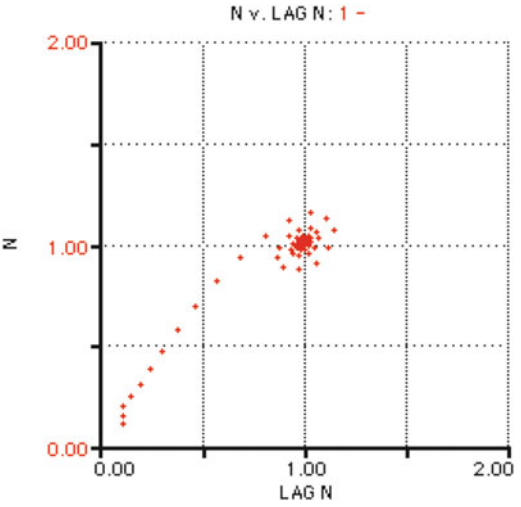

Fig. 4.15

4.4 Simple Oscillator Model Equations

$N(t) = N(t - dt) + (\Delta N) * dt$
INIT N = .1

INFLOWS:
$\Delta N = R*LAG\_N*(1-LAG\_N)$
LAG_N = DELAY(N,1)
R = 1.3
K = 1

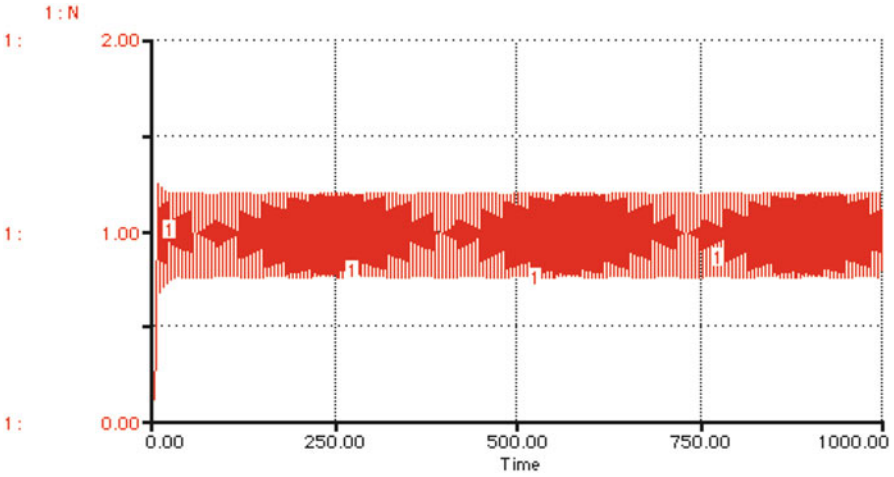

Fig. 4.16

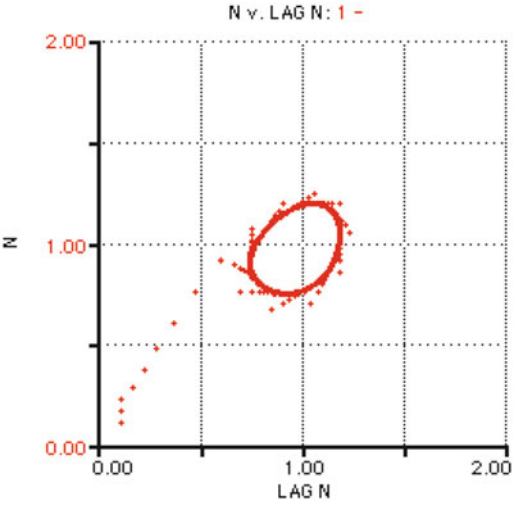

Fig. 4.17

References

1. Jenson RV (1987) Classical chaos. Am Sci 75:168–181
2. Kaufman S (1993) The origins of order: self organization and selection in evolution. Oxford University Press, New York

Chapter 5
Spatial Dynamics

There is no coming into being of aught that perishes, nor any end for it... but only mingling, and separation of what has been mingled.

(Empedocles)

5.1 Spatial Dynamics Model

In the previous chapters we have modeled population dynamics in response to births and deaths, and we have assumed that the respective flows originate or disappear in "clouds." If we deal with migration in the context of population dynamics we may want to explicitly model the source and recipients of the flows of migrants. This is the topic of this chapter. The model developed here is quite general and provides a basis for our discussion of spatial dynamics in later chapters.

Here is what might be called a "mobility framework," where migration from one state to another depends only on the current status of the donor state. Mobility might be the movement of animals between various resource points, chemical diffusion, the circulation of money, or the location of people in towns on a landscape.

Mobility is defined by flows that are calculated as the difference between the value of the state variables at two points on the landscape. This difference is then multiplied by a fixed coefficient that reflects the strength of migration. The STELLA model is shown in Fig. 5.1.

Figure 5.2 shows the egalitarian or the strict diffusion solution—no matter what the initial status of each of the states, all stocks find the same equilibrium. This is due to the fact that each of the exchanges is multiplied by the same coefficient.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_5,

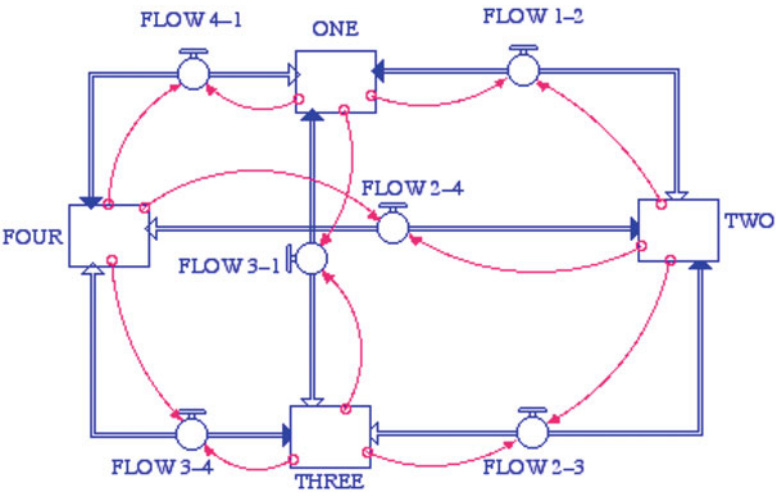

Fig. 5.1

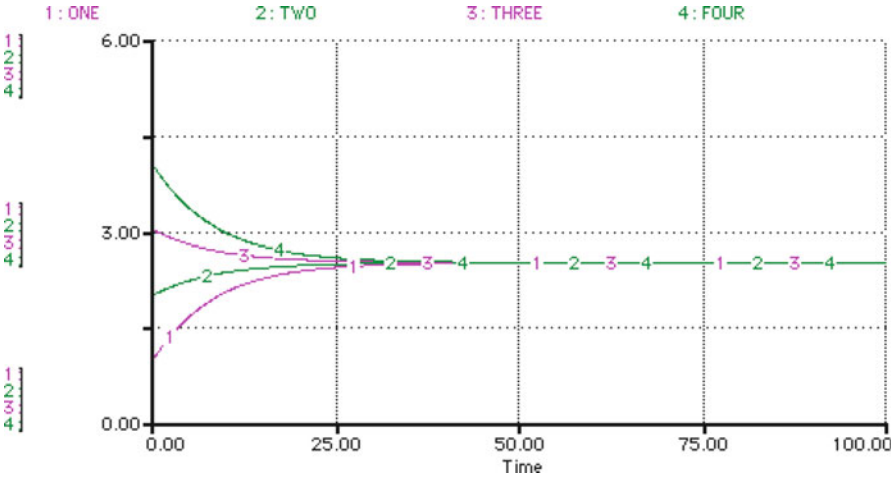

Fig. 5.2

The final level is just the average of the four initial stock values, since there is no loss from the system.

This is a simple but profound view of the long-term processes of thermodynamics. While the system starts with low entropy (high differentiation), it ends with its highest possible entropy (no differentiation). Show that with delays, this system can be kept from this zero entropy state. The delays in a real system require an external input of energy and this is the root cause of the system's apparent reluctance to dissipate.

Remove the two internal and one of the external flows. Does this truncated system show the same result? Why? Compare the two systems through time.

Fig. 5.3

ONE	2.5
TWO	2.5
THREE	2.5
FOUR	2.5

Fig. 5.4

The drivers in this system are actually the concentration differences expressed in the six flows. Here we assumed that all volumes are 1.0. Make each stock a different volume. Even make one of them a function of time as in a collapsing balloon. Then set up converters for the four stocks that represent their dynamic concentrations. (Remember, the stocks are measuring the weight such as in grams of gas in each of them. The concentrations are then in grams per unit volume.) Have the concentration differences times a specific diffusion constant for each flow be the flow equation in each of the six cases. This is a more realistic representation of diffusion conditions.

To quickly compare the results of each model run without plotting a table, choose the numeric display—you find it among the STELLA icons to the right of the graph and table icons that we have used before and a "status indicator" that we won't use here (Fig. 5.3)—and place it in the STELLA diagram.

Double-click on the numeric display icon, select one of the system's state variables, and click on OK. Repeat this procedure for the other state variables. These numeric displays function like a counter and show you the value of parameter as the model runs. If you specified the displays to maintain the ending balance, you can quickly compare the results from model run to model run. For example, the egalitarian result is shown in our model for coefficients equal to 0.03 (Fig. 5.4).

Return to the case of coefficients smaller than one. What are the effects of time lags on the time it takes for the system to equilibrate? Will the same equilibrium be reached as in the absence of those lags? Can these new time constants cause oscillation? Why? There are six connectors in this model. What is the minimum number to allow this same equilibrium to be reached?

5.2 Spatial Dynamics Model Equations

$$\text{FOUR}(t) = \text{FOUR}(t - dt) + (\text{FLOW\_3–4} + \text{FLOW\_2–4} - \text{FLOW\_4–1}) * dt$$
$$\text{INIT FOUR} = 4$$

INFLOWS:
FLOW_3–4 = .03∗(THREE-FOUR)
FLOW_2–4 = .03∗(TWO-FOUR)
OUTFLOWS:
FLOW_4–1 = .3∗(FOUR-ONE)
ONE(t) = ONE(t - dt) + (FLOW_4–1 + FLOW_3–1 - FLOW_1–2) ∗ dt
INIT ONE = 1

INFLOWS:
FLOW_4–1 = .3∗(FOUR-ONE)
FLOW_3–1 = .03∗(THREE-ONE)
OUTFLOWS:
FLOW_1–2 = .03∗(ONE-TWO)
THREE(t) = THREE(t − dt) + (FLOW_2–3 − FLOW_3–4 − FLOW_3–1) ∗ dt
INIT THREE = 3

INFLOWS:
FLOW_2–3 = .03∗(TWO-THREE)
OUTFLOWS:
FLOW_3–4 = .03∗(THREE-FOUR)
FLOW_3–1 = .03∗(THREE-ONE)
TWO(t) = TWO(t − dt) + (FLOW_1–2 − FLOW_2–3 − FLOW_2–4) ∗ dt
INIT TWO = 2

INFLOWS:
FLOW_1–2 = .03∗(ONE-TWO)
OUTFLOWS:
FLOW_2–3 = .03∗(TWO-THREE)
FLOW_2–4 = .03∗(TWO-FOUR)

Part II
Physical and Biochemical Models

Chapter 6
Law of Mass Action

Let us now consider the character of the material Nature whose necessary results have been made available... for a final cause.

(Aristotle)

6.1 Law of Mass Action Model

The law of mass action is a powerful concept that describes the average behavior of a system that consists of many interacting parts such as molecules that react with each other, or viruses that are passed along from a population of infected individuals to nonimmune ones. The law of mass action has been derived first for chemical systems but subsequently found high use in epidemiology and ecology. In this chapter, we will discuss the law of mass action in the context of a simple chemical system. In later chapters, we will apply it to issues as diverse as enzyme–substrate interactions, the spread of a disease, or the colonization of landscape patches.

Let us consider the case of oxygen O reacting with hydrogen molecules H to form water H_2O. The stock of each substance is given as a concentration, measured in moles per cubic meter. For simplicity, we assume initial conditions of 200 moles of hydrogen per cubic meter and 100 moles of oxygen atoms per cubic meter. Thus, there is enough of each initial stock to just form 100 molecules of water. The stochiometric equation for this reaction is

$$O + 2H \rightarrow H_2O \tag{6.1}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_6,

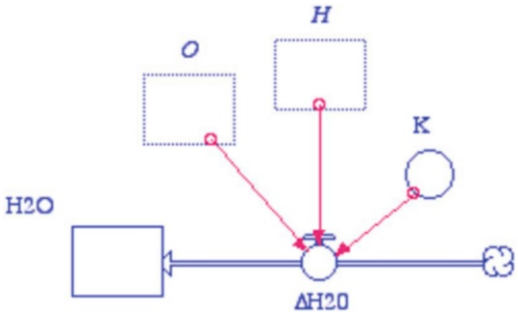

Fig. 6.1

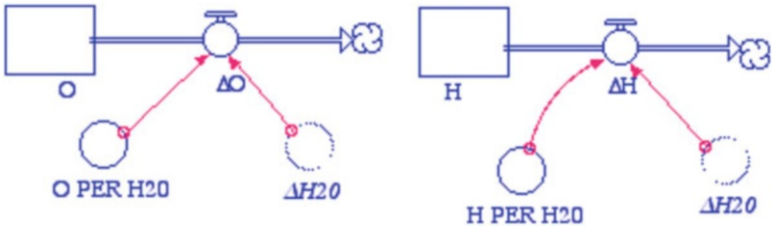

Fig. 6.2

Obviously, not all of the oxygen molecules will instantaneously find hydrogen molecules to engage in the reaction. Rather, the reaction will take place over time. The reaction velocity will be high early on as the concentrations of each of the reactants are high. With declining concentrations, the reaction velocity declines. The law of mass action expresses the rate of formation of the product H_2O from the reactants O and H:

$$\Delta H_2O = K * H * O \tag{6.2}$$

with K as the reaction rate constant, measured in 1 per second. This law provides chemistry with dynamics. It was first used many centuries ago and the theoretical basis for it was established in the late nineteenth century. If the pressure and/or temperature vary during the reaction, H and O have exponents to modulate the rate for such effects.

The process by which H_2O is generated in our model is shown in Fig. 6.1.

The concentrations of the reactants decrease as the reaction proceeds. To calculate the respective outflows from the stocks O and H, we need to specify the number of moles that enter the reaction to form one mole of the product. For O and H these are, respectively, 1 and 2. The part of the model that calculates removal of O and H for the formation of H_2O is shown in Fig. 6.2.

The results of our model are shown in Fig. 6.3 for a hypothetical value of $K = 0.005$. As we would expect, the concentrations of both reactants decline as the product is formed. Also, the rate of reaction declines. Will eventually all the

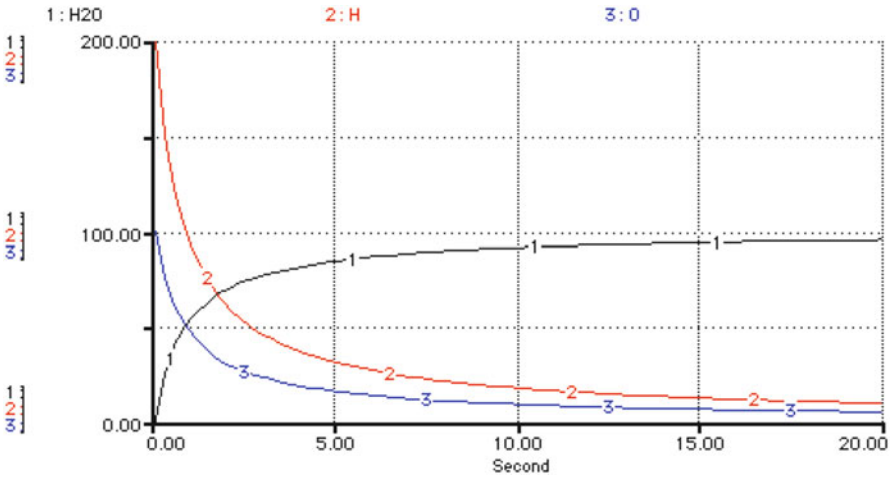

Fig. 6.3

reactants be used up and exactly 100 moles of water per cubic meter be formed? Make an educated guess and then run the model for a longer time frame and observe the results.

The model above implicitly assumes that temperatures and pressures remain constant throughout the reaction. However, the formation of water from oxygen and hydrogen produces a significant amount of heat that, in turn, increases the reaction velocity. Can you introduce this effect in the model? Expand the model to the case of a simple photosynthetic process in which carbon dioxide and water react to form a glucose and oxygen:

$$6CO_2 + 6H_2O \rightarrow C_6H_{12}O_6 + 6O_2 \tag{6.3}$$

Can you change your model to capture the metabolic process in which the glucose reacts with oxygen to form water and carbon dioxide? Find in the literature characteristic reaction rate constants for both the formation and metabolism of various types of glucose and model the respective chemical reactions.

6.2 Law of Mass Action Model Equations

H(t) = H(t − dt) + (− ΔH) ∗ dt
INIT H = 200 {Moles per Cubic Meter}

OUTFLOWS:
ΔH = ΔH2O∗H_PER_H2O {Decrease in H concentration as a result of H2O formation; measured in Moles per Cubic Meter per Second}
H2O(t) = H2O(t − dt) + (ΔH2O) ∗ dt
INIT H2O = 0 {Moles per Cubic Meter}

INFLOWS:
ΔH2O = K∗H∗O {Increase in H2O concentration; simple second-order reaction overall and first order reaction with respect to each of the elements; measured in Moles per Cubic Meter per Second}
O(t) = O(t − dt) + (− ΔO) ∗ dt
INIT O = 100 {Moles per Cubic Meter}

OUTFLOWS:
ΔO = O_PER_H2O∗ΔH2O {Decrease in 0 concentration as a result of H2O formation; measured in Moles per Cubic Meter per Second}
H_PER_H2O = 2 {Moles H required per Mole H2O formed - from stoichiometry of reaction}
K = .005 {1/Second-concentration.}
O_PER_H2O = 1 {Moles O required per Mole H2O formed - from stoichiometry of reaction}

Chapter 7
Catalyzed Product

Many bodies . . . have the property of exerting on other bodies an action which is very different from chemical affinity. By means of this action they produce decomposition in bodies, and form new compounds into the composition of which they do not enter. This new power, hitherto unknown, is common both in organic and inorganic nature. I shall call it catalytic power. I shall also call Catalysis the decomposition of bodies by this force.

(Berzelius, Edin. New Phil. Jrnl. XXI., 1836)

7.1 Catalyzed Product Model

In the previous chapter we have used the law of mass action to describe chemical change of two substances interacting with each other and forming a product that is chemically distinct from the two reactants. In this chapter we expand on that model and deal with the case in which—after a series of reactions—one of the reactants reemerges to enter the reaction anew. For example, some enzyme E may enter a chemical reaction from which an intermediate product I results. This intermediate product, in turn, may enter a reaction from which E is released in unchanged form together with a new product F. An additional substance active in this process is the substrate D, which is converted into the product F by action of the enzyme E.

Such catalyzed reactions are common in biological processes. One example of these reactions is the production of fructose (F) from dextrose (D). In this process, the enzyme (E) mechanically locks onto the substrate molecule, breaks it into a new molecule, fructose, and is released again after the chemical reaction occurred (see [1]).

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_7,

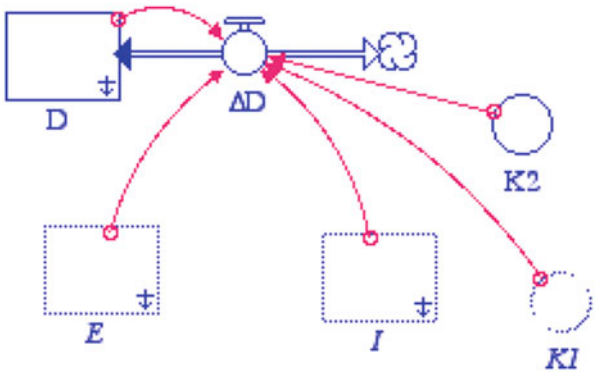

Fig. 7.1

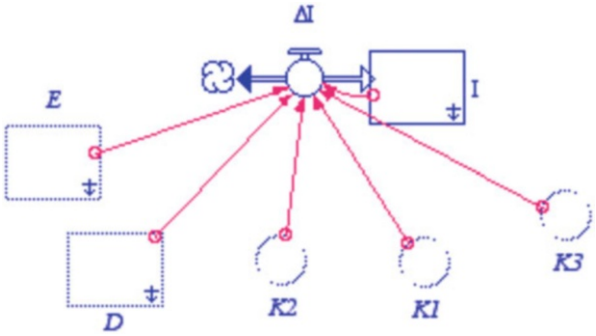

Fig. 7.2

Denote the reaction rate constants as K1, K2, and K3. With this notation, the basic reaction equation is:

$$\mathrm{E} + \mathrm{D} \underset{\mathrm{K2}}{\overset{\mathrm{K1}}{\leftrightarrow}} \mathrm{I} \overset{\mathrm{K3}}{\rightarrow} \mathrm{E} + \mathrm{F} \tag{7.1}$$

The four basic differential equations that define the rate expressions, or flows, are:

$$\mathrm{dD}/\mathrm{dt} = \mathrm{K1} * \mathrm{D} * \mathrm{E} - \mathrm{K2} * \mathrm{I} \tag{7.2}$$

$$\mathrm{dI}/\mathrm{dt} = \mathrm{K1} * \mathrm{S} * \mathrm{E} - (\mathrm{K2} + \mathrm{K3}) * \mathrm{I} \tag{7.3}$$

$$\mathrm{dE}/\mathrm{dt} = -\mathrm{dI}/\mathrm{dt} \tag{7.4}$$

$$\mathrm{dF}/\mathrm{dt} = \mathrm{K3} * \mathrm{I} \tag{7.5}$$

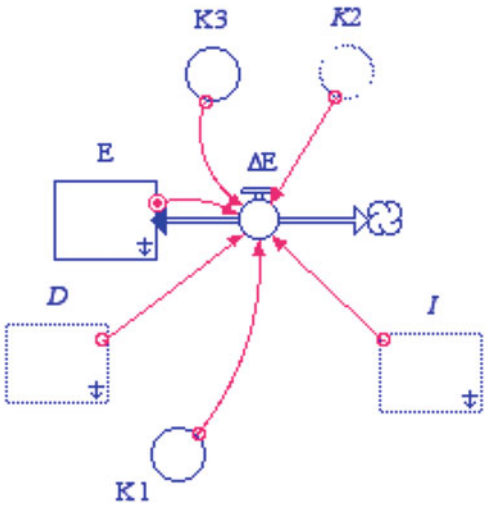

Fig. 7.3

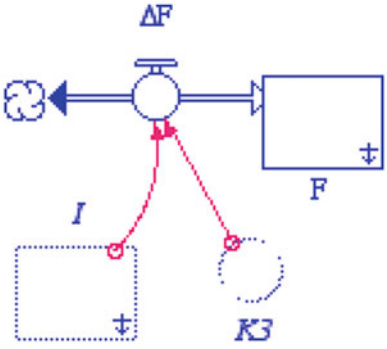

Fig. 7.4

The differential equations are used in four modules of the STELLA model as shown in Figs. 7.1, 7.2, 7.3, and 7.4. Each of the state variables is treated as a concentration.

The results (Fig. 7.5) show a temporary decline in the concentration of the enzyme as it gets locked up in the production of I. As the concentration of the intermediate product declines—in response to the formation of the final product F—the concentration of E increases again. As the substrate D gets depleted, no more reactions take place and the system settles down to a set of equilibrium concentrations for E and F. What is the continuous addition rate of D such that withdrawal can be achieved for F at a rate of 0.1 units per time step?

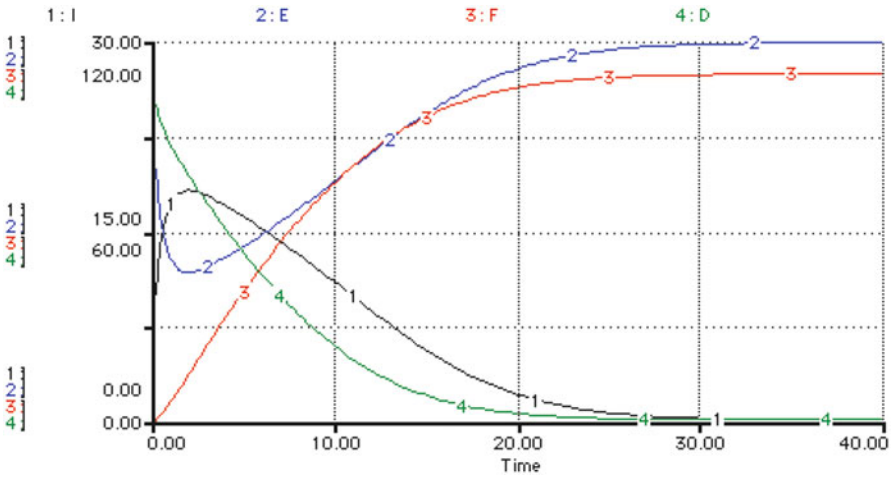

Fig. 7.5

For the model results above, we have arbitrarily set K1 = K2 = 0.01 and K3 = 0.5. Change the reaction rates and observe the results. Can you make this model with fewer stocks, say just ones for I and F?

In this chapter we modeled in a general way the catalyzed reactions of enzymes and substrates. In the following chapter, we broaden our focus and deal with an entire cell that takes up nutrients from its environment and excretes waste products into its environment.

7.2 Catalyzed Product Model Equations

D(t) = D(t − dt) + (−ΔD) ∗ dt
INIT D = 100 {Moles per Cubic Meter}

OUTFLOWS:
ΔD = K1∗D∗E−K2∗I {Moles per Cubic Meter per Time Period}
E(t) = E(t − dt) + (−ΔE) ∗ dt
INIT E = 20 {Moles per Cubic Meter}

OUTFLOWS:
ΔE = K1∗D∗E−(K2+K3)∗I {Moles per Cubic Meter per Time Period}
F(t) = F(t − dt) + (ΔF) ∗ dt
INIT F = 0 {Moles per Cubic Meter}

INFLOWS:
ΔF = K3∗I {This is the Michaelis-Menten enzyme model; P_Rate measured in Moles per Cubic Meter per Time Period}
I(t) = I(t − dt) + (ΔI) ∗ dt
INIT I = 10 {Moles per Cubic Meter}

INFLOWS:
ΔI = K1∗D∗E−(K2+K3)∗I {Moles per Cubic Meter per Time Period}
K1 = .01 {1/Time Period}
K2 = .01 {1/Time Period}.
K3 = .5 {1/Time Period}

Reference

1. Spain JD (1982) Basic microcomputer models in biology. Addison-Wesley, Reading

Chapter 8
Two-Stage Nutrient Uptake

In the nutrient-rich waters of the Thames type, a burst of algal growth may sometimes cease before any serious depletion of the mineral nutrient in the water has apparently taken place.

(Nature, 21 Sept. 1946)

8.1 Two-Stage Nutrient Uptake Model

In the previous model we have investigated a chemical reaction that may take place within a cell. In this chapter, we model in more detail the activities of an entire cell that receives nutrients from its environment, uses these nutrients for growth and maintenance, and then excretes waste products back into its environment.

Assume a cell with an internal nutrient concentration Q is immersed in a media with a nutrient concentration N. The growth of the cell biomass X is directly dependent on the internal rather than the external nutrient concentration. Nutrient uptake is proportional to the cell biomass. The proportionality factor is MU. An outline of the structure of this system is given in Fig. 8.1. The model is called the Caperon–Droop model. For a reference on the origins of this model see Spain [1].

Through respiration and mortality, the nutrient is passed back into solution outside the cell. The rate of return from the cell is proportional to the biomass of the cell. The proportionality factor is R. There is a minimum level of the internal nutrient concentration, Q_0, which is needed before the cell will grow at all. Thus, the change in cell biomass, ΔX, is

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_8,

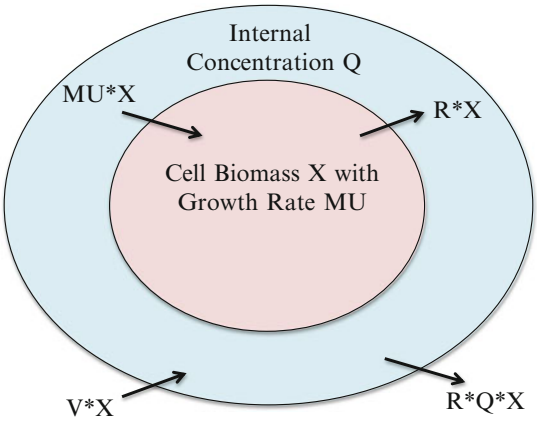

Fig. 8.1

$$\Delta X = \text{IF Q} \geq \text{Q\_0 THEN}(\text{MU} - \text{R}) * \text{X ELSE 0.} \tag{8.1}$$

The change in the nutrient concentration outside the cell, ΔN, depends on the nutrient passing through the cell wall,

$$\Delta N = \text{IF N} > 0 \text{ THEN R} * \text{Q} * \text{X} - \text{V} * \text{X ELSE 0.} \tag{8.2}$$

R∗Q∗X is the return of the nutrients to the external environment, depending on the internal nutrient concentration Q. The nutrient balance equation for the internal concentration of the nutrient is

$$\Delta Q = \text{IF Q} \geq \text{Q\_0 THEN V} - \text{MU} * \text{Q ELSE 0.} \tag{8.3}$$

Equation (8.3) is a form of the Monod equation used to predict the change in the internal concentration.

The rates V and MU are calculated from a now-standard Michaelis-Menten formula. The general form for this equation is derived from the enzyme-substrate equation discussed in the previous chapter [2]:

$$\text{V} = \text{VM} * \text{N}/(\text{KN} + \text{N}) \tag{8.4}$$

$$\text{MU} = \text{MU BAR} * (\text{Q} - \text{Q\_0})/(\text{KQ} + (\text{Q} - \text{Q\_0})) \tag{8.5}$$

with VM the maximum rate of nutrient uptake per unit biomass, KN the half saturation constant for nutrient uptake, KQ the half saturation constant for growth, and MU BAR the maximum biomass growth rate.

Our model is shown in Fig. 8.2. Recognize, that this model differs from the specification of the reaction rate in the previous chapter in that the reaction rate here is not based on the product of the concentrations of the compounds. Also, note well

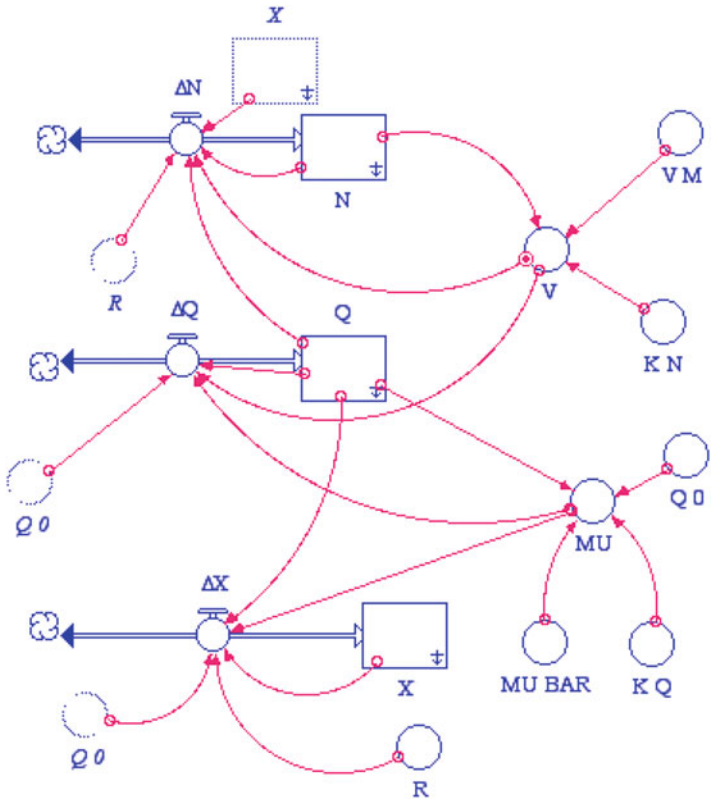

Fig. 8.2

the way in which MU is used in the ΔX and ΔQ equations. It is an interesting use of a common variable. The ΔQ equation makes more sense if you multiply through by X. Curiously, R∗X and R∗Q∗X are not included in the ΔQ equation. The R factor is apparently designed to represent the communication of the cell biomass to the main nutrient source only. This fact, as with the definition of Q, is the likely result of experimental measurement problems of the time.

In this model N and Q are concentrations. But they are different kinds of concentrations. N is measured in mgN/liter, a volumetric concentration and Q is measured in mgN/mgX, a mass-based concentration. It must be too hard to measure the volume of a cell. The units are tricky. The units of the uptake rate for N, V are mgN/mgX/hour, while the rate of formation and mortality of the biomass X are 1/hour. The units of MU are 1/time. Work out the units of ΔN, ΔQ, and ΔX to make sure that they are consistent.

The STELLA diagram of Fig. 8.2 shows the three differential equations and the supporting parameters. The graph in Fig. 8.3 shows how the concentrations and the biomass levels change with time and how the cell growth rate depends on the internal nutrient concentration. To obtain such results, we had to run the time step at

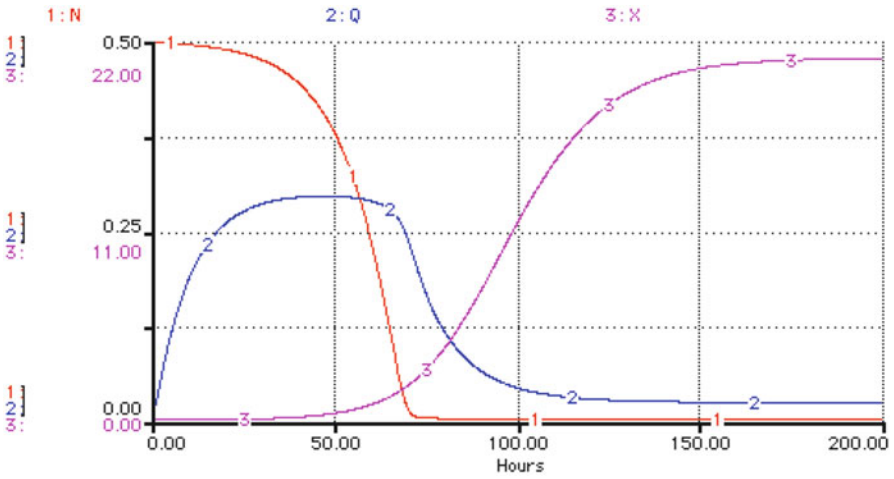

Fig. 8.3

DT = 0.125. Try running the model with Euler and DT = 1.0 and note the difference in results. Try shifting the parameters and note their effect on the concentration trajectories.

How is it possible in our model that the biomass can maintain itself in the steady state forever? Does this cell know something about perpetual motion that it is not telling us or have we missed something? Actually, the cell is using energy all the time. It has to be taking up high quality energy (ATP) and giving off low quality energy (heat). We are simply not modeling that part of the cell activity. We model only the use of a single nutrient. Neither are we modeling things going on at the smaller level where things are inevitably falling apart and being replaced, and that replacement process is not faultless.

The following chapter will expand the boundaries of systems processes beyond an individual cell to the level of an organism. There, we will distinguish different compartments among which a substance is being distributed. Models of organs and entire organisms are presented in Part IV of the book.

8.2 Two-Stage Nutrient Uptake Model Equations

$N(t) = N(t - dt) + (\Delta N) * dt$
INIT N = 0.5 {mg/liter}

INFLOWS:
ΔN = IF N > 0 THEN R∗Q∗X −V∗X ELSE 0
$Q(t) = Q(t - dt) + (\Delta Q) * dt$
INIT Q = 0.02 {mg/mg of X}

INFLOWS:
$\Delta Q = \text{IF } Q \geq Q\_0 \text{ THEN } V - MU*Q \text{ ELSE } 0$
$X(t) = X(t - dt) + (\Delta X) * dt$
INIT X = 0.01 {mg/liter. Total cell biomass per liter.}

INFLOWS:
$\Delta X = \text{IF } Q \geq Q\_0 \text{ THEN } (MU - R)*X \text{ ELSE } 0$
K_N = 0.05 {mg/liter}
K_Q = 0.03 {mgN/mgX}
MU = MU_BAR*(Q - Q_0)/(K_Q + (Q − Q_0)) {1/hour}
MU_BAR = 0.1 {1/hour}
Q_0 = 0.02 {mgN/mgX}
R = 0.01 {1/hour}
V = V_M*N/(K_N + N) {mgN/mgX/hour}
V_M = 0.03 {mgN/mgX*hour}

References

1. Spain JD (1982) Basic microcomputer models in biology. Addison-Wesley, Reading
2. Edelstein-Keshet L (1988) Mathematical models in biology. Random House, New York

Chapter 9
Iodine Compartment

Iodine was discovered accidentally, about the beginning of the year 1812, by M. Courtois, a manufacturer of saltpetre at Paris.

(Henry, Elem. Chem. I., 1826)

9.1 Iodine Compartment Model

The previous two models focused, respectively, on chemical reactions within a cell, and on the activities of an entire cell. In this chapter we model the flow of a substance—iodine—among different parts of an organism. This is a donor-controlled model.

Iodine is found in three places in the body: the thyroid gland, tissue connected to and surrounding the thyroid gland, and inorganic iodine in the circulatory system. Figure 9.1 shows how the flow balance is established for each of the stocks representing each of the places that iodine can be found. For our model (Fig. 9.2), inorganic iodine is constantly injected into the system at 150 μg/day, for example, through certain foods, or iodized salt. This exogenous iodine input is denoted D_i in Fig. 9.1.

Note that all flows are donor controlled. This of course is not the case in all models. The Extra Tissue is that immediately adjacent to the thyroid. The Inorganic Iodine is in the blood stream.

The thyroid gland is consuming inorganic iodine and some leaves the system with urine. Ultimately, the remainder leaves the digestive system as feces. Conversion rates required to specify the flows from compartment to compartment are assumed to be dependent on the amount of iodine in each place. This is a questionable assumption that nevertheless is supported by experiment.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_9,

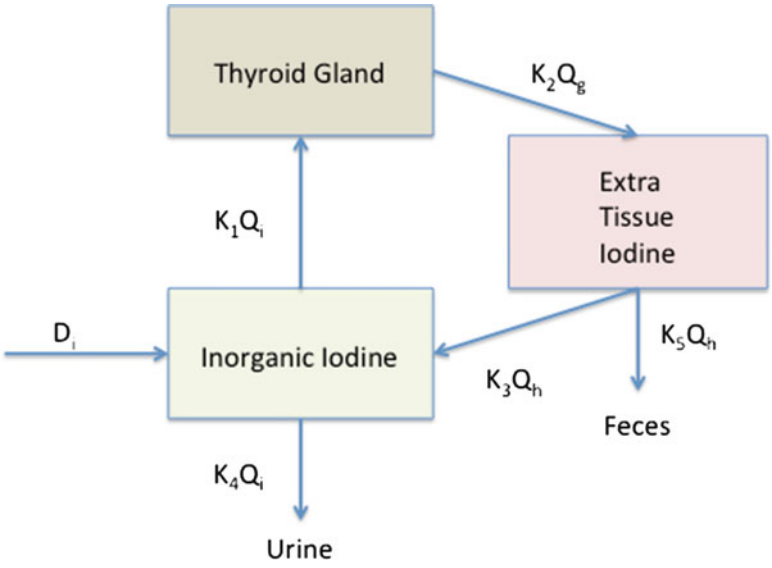
Thyroid Gland
K_2Q_g
Extra Tissue Iodine
K_1Q_i
D_i
Inorganic Iodine
K_5Q_h
K_3Q_h
Feces
K_4Q_i
Urine

Fig. 9.1

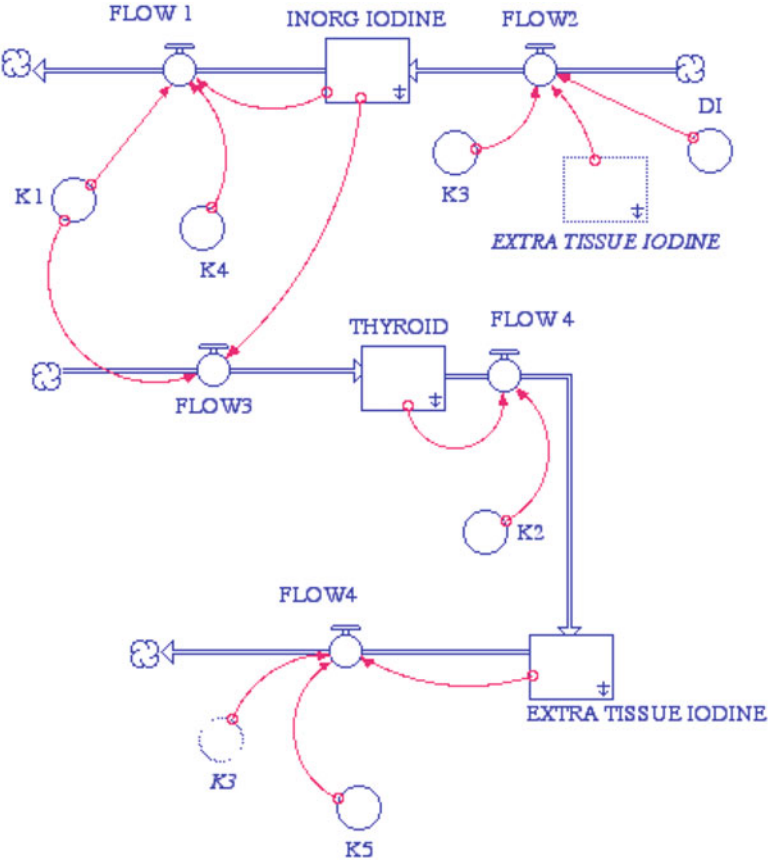
FLOW 1
INORG IODINE
FLOW2
DI
K1
K3
K4
EXTRA TISSUE IODINE
THYROID
FLOW 4
FLOW3
K2
FLOW4
EXTRA TISSUE IODINE
K3
K5

Fig. 9.2

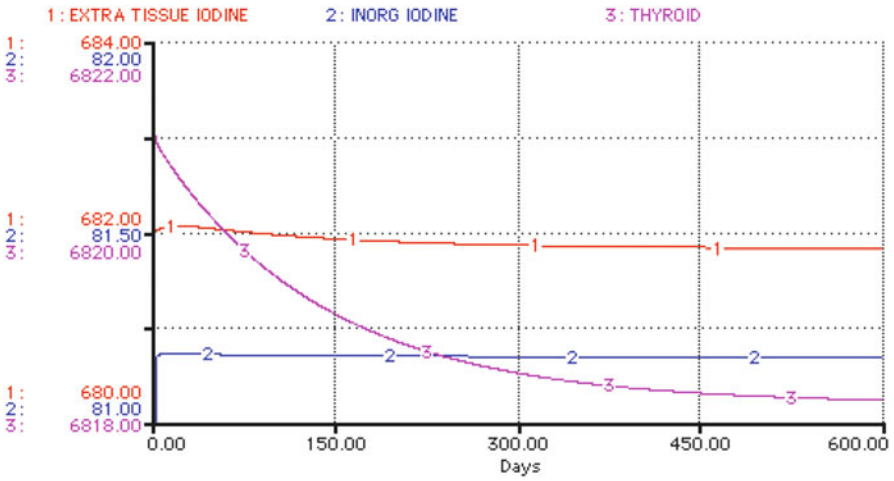

Fig. 9.3

Results of this model are shown in Fig. 9.3. The thyroid level declines to a stable level with the input rate. Change the external input rate to one tenth of the stable level and plot the result. This level represents an extremely deficient iodine level.

This very basic model of a complex and important process demonstrates how one can model simply yet accurately. Note that we did not use the idea of the mass action law here. Why not? Can you see here where some flows are recipient-controlled and some are donor-controlled? This identification of the state variable controls early in the modeling process can be very helpful.

We have begun this part of the book on physical and biochemical with simple chemical reactions and proceeded to catalyzed processes, the activities of an individual cell, and the distribution of chemical substances among different compartments. Let us return in the following chapter to chemical reactions and combine our insight into oscillatory system behavior of Chap. 4 with our knowledge about chemical processes.

9.2 Iodine Compartment Model Equations

EXTRA_TISSUE_IODINE(t) = EXTRA_TISSUE_IODINE(t − dt) + (FLOW_6 − FLOW_5) ∗ dt
INIT EXTRA_TISSUE_IODINE = 682 {micrograms}

INFLOWS:
FLOW_6 = K2∗THYROID

OUTFLOWS:
FLOW_5 = (K3+K5)∗EXTRA_TISSUE_IODINE
INORG_IODINE(t) = INORG_IODINE(t − dt) + (FLOW2 − FLOW1) ∗ dt
INIT INORG_IODINE = 81 {micrograms}

INFLOWS:
FLOW2 = DI+K3∗EXTRA_TISSUE_IODINE
OUTFLOWS:
FLOW1 = (K1+K4)∗INORG_IODINE
THYROID(t) = THYROID(t − dt) + (FLOW3 − FLOW4) ∗ dt
INIT THYROID = 6821 {micrograms}

INFLOWS:
FLOW3 = K1∗INORG_IODINE
OUTFLOWS:
FLOW4 = K2∗THYROID
DI = 150 {micrograms/day}
K1 = 0.84 {1/day}
K2 = 0.01 {1/day}
K3 = 0.08 {1/day}
K4 = 1.68 {1/day}
K5 = 0.02 {1/day}

Chapter 10
The Brusselator

In the Beginning the Heav'ns and Earth Rose out of Chaos.

(P.L. Milton, 1667)

10.1 Brusselator Model Equations

One important characteristic of real-world processes is that interactions among system components are nonlinear. As we have seen in the previous chapters, nonlinearities can give rise to rich temporal patterns. An example of nonlinearities in chemical reactions is the autocatalytic process

$$A + X \rightarrow 2X \tag{10.1}$$

whereby X stimulates its own production from A. Such autocatalytic reactions may involve a series of intermediate sates, for example if X produces a substance Y, which in turn accelerates the production of X. One such cross-catalytic reaction has been studied extensively by a group of scientists around the Nobel Laureate Ilya Prigogine in Brussels, and is known as the Brusselator [1]. It involves the following series of reaction steps:

$$A \rightarrow X \tag{10.2}$$

$$2X + Y \rightarrow 3X \tag{10.3}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_10,

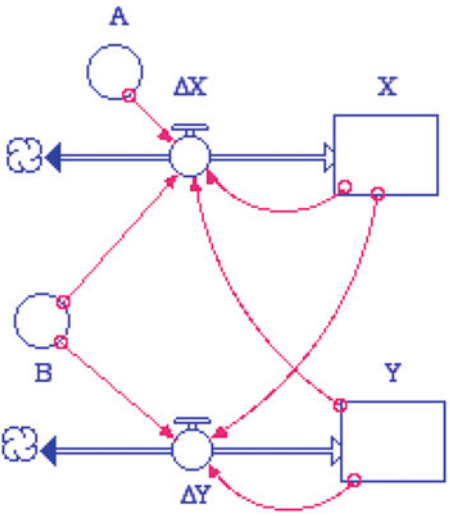

Fig. 10.1

$$B + X \rightarrow Y + D \tag{10.4}$$

$$X \rightarrow E \tag{10.5}$$

A and B are used to make D and E with intermediate products X and Y.

These reactions can be maintained far from equilibrium by continually supplying the substances A and B and extracting D and E. These additions and subtractions eliminate the back reactions by holding the concentrations A, B, D, and E constant. This assumption allows us to capture in the STELLA model the concentrations of those products as transforming variables rather than reservoirs. In contrast, the two intermediate components (X and Y) may have concentrations that change in time.

For simplicity, and without loss of generality, we set the kinetic constants equal to one. The following system of nonlinear equations results, after eliminating D, which does not *enter* any of the reactions and is continuously removed from the system. Thus, by analogy with Chap. 7:

$$dX/dt = A + X^2 * Y - B * X - X \tag{10.6}$$

$$dY/dt = B * X - X^2 * Y \tag{10.7}$$

Figure 10.1 shows the corresponding STELLA diagram. The assumption that the products A, B, D, and E are held constant through either removal or addition allows us to model those products as constants.

Set $A = 0.7$ and $B = 2$, and run the model. Choose the initial conditions $X(t = 0) = A$ and $Y(t = 0) = B/A$, run the model at a $DT = 0.0625$, and you will

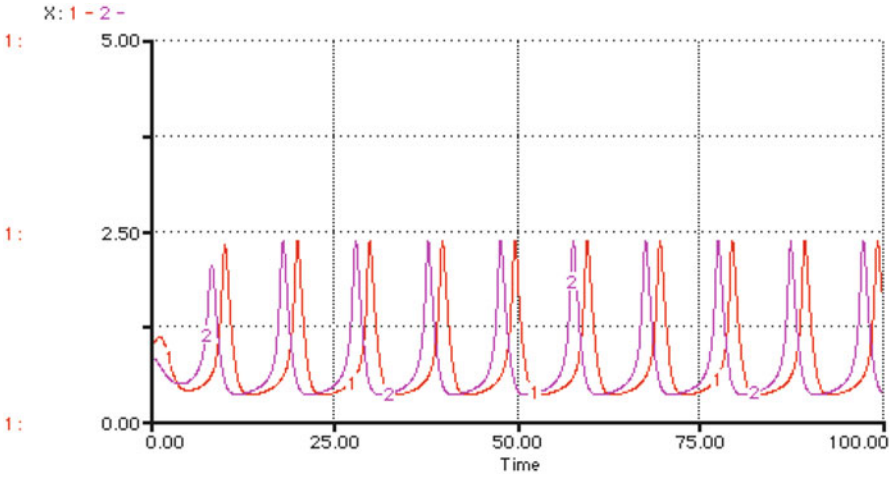

Fig. 10.2

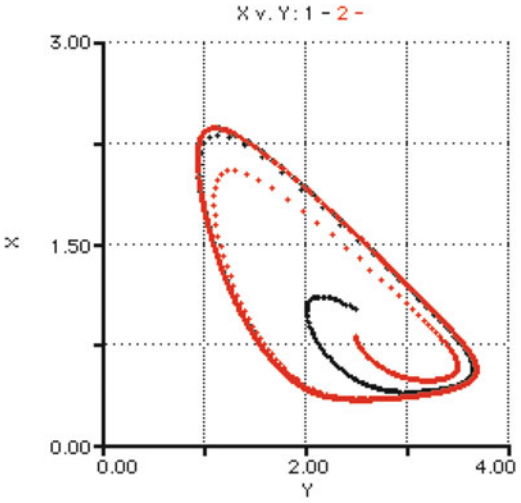

Fig. 10.3

find the steady state of this system. However, this steady state is unstable. To confirm this observation, run the model for two initial conditions close to the ones that yield steady state, for example $X(t=0)=1$, and $X(t=0)=0.8$, with $Y(t=0)=2.5$ in each of the two cases. Figure 10.2 shows the results plotted in the same graph, and in Fig. 10.3 we show the phase diagrams for these two model runs.

Small changes in the initial conditions lead away from the (unstable) steady state but ultimately lead the system to a steady limit cycle. Will this cycle be chosen by the system for initial conditions that are significantly different from those that

Table 10.1 Initial conditions for five runs

Run	X(t = 0)	Y(t = 0)
1	0	0
2	0	1
3	0	2
4	3	0.5
5	1.5	0

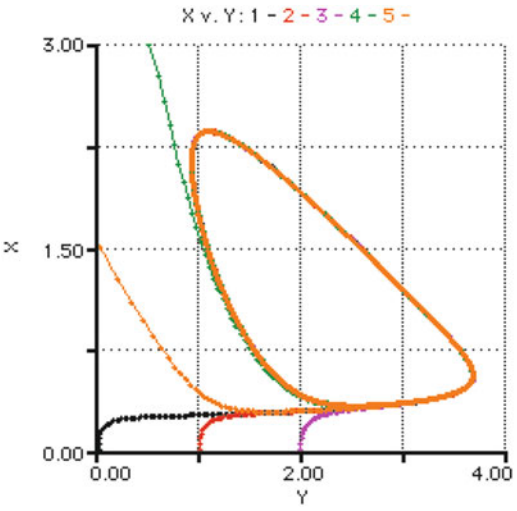

Fig. 10.4

we already chose? Here are the results for five runs with the setups shown in Table 10.1:

These runs were performed with STELLA's sensitivity analysis, making use of the *ad hoc values* option and plotting X against Y in a *comparative* scatter plot. The results show that irrespective of the initial conditions, the system converges to the same limit cycle.

Note that if $dX/dt = dY/dt = 0$, then $A = X_e$, the equilibrium X. When $A < 1$, we have a loop; $A > 1$, we have a line. The break point between a line and a loop is $A = 1$ (Fig. 10.4).

Two conditions are necessary for the limit cycle to occur—the system must be open, and interactions among system components must be nonlinear. The first of these conditions is fulfilled by withdrawing and adding the products A, B, D, and E, effectively leaving their concentrations constant. As a result, the system is maintained away from an equilibrium at which reactants get used up and the chemical reactions come to a halt. The second condition is met by Eqs. (10.6) and (10.7). Prigogine and his coworkers argue that virtually any real-world system is open, characterized by nonlinearities, and maintained out of equilibrium with its surroundings. Individual organisms receive material and energy inputs from their

environment and excrete waste products and waste heat. Similarly, entire ecosystems channel materials and energy through their systems. The constant influx of "reactants" and energy into these systems and the constant removal of waste materials and heat make it possible for these systems to function. They are clearly open and not in equilibrium with their surroundings.

Change the values for A and B, and rerun the model for alternative initial conditions. Can you find the steady-state conditions? How does the limit cycle change? How are the results affected by the choice of DT and integration methods.

Once you explored the dynamics of this system and familiarized yourself sufficiently with the models of chemical processes discussed in this part of the book, move on to learn more about the application of physical principles and tools to the understanding of biological processes. This is the topic of the following chapter.

10.2 Brusselator Model Equations

X(t) = X(t − dt) + (ΔX) ∗ dt
INIT X = 1
INFLOWS:
ΔX = A+X^2∗Y−B∗X−X

Y(t) = Y(t − dt) + (ΔY) ∗ dt
INIT Y = 2.5
INFLOWS:
ΔY = B∗X−X^2∗Y

A = .7
B = 2

Reference

1. Prigogine I (1980) From being to becoming: time and complexity in the physical sciences. W. H. Freeman and Company, New York

Chapter 11
Signal Transmission

> *"*Ginny!*" said Mr. Weasley, flabbergasted. "Haven't I taught you* anything*? What have I always told you? Never trust anything that can think for itself* if you can't see where it keeps its brain*?"*
>
> (J.K. Rowling, Harry Potter and the Chamber of Secrets)

11.1 Fitzhugh–Nagumo Neuron Model

The previous chapter showed rich dynamics for the case of *chemical* reactions in open system. In that chapter, we stressed the sensitivity of system responses to initial conditions and the role of nonlinearities in determining system behavior. The model of this chapter captures nonlinearities in the changes of the *physical* state of a cell in response to electrical impulses from its surroundings.

Figure 11.1 depicts a neuron or nerve cell. Neurons consist of a cell body, picking up possible signals from one or all of its several dendritic branches (signals from other neurons) and transmitting these received (electrical) signals down a relatively long axon pathway to its terminal. At the terminal the signal is amplified and transmitted across a synapse to the next nerve cell. The cell remains inactive until the collective input from the dendrites reaches a critical level whereupon the cell "fires"—it reacts in such a way as to amplify the collective input signals into a signal potential at its terminal end. If the collective input signal is not great enough the signal dies out in the cell due to the action of a recovery mechanism.

The Fitzhugh–Nagumo model is of a nerve cell under special laboratory conditions, where all dendritic receivers are kept at the same potential. The space-change in potential along the axon and throughout the cell is thus ignored. The only way to

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_11,

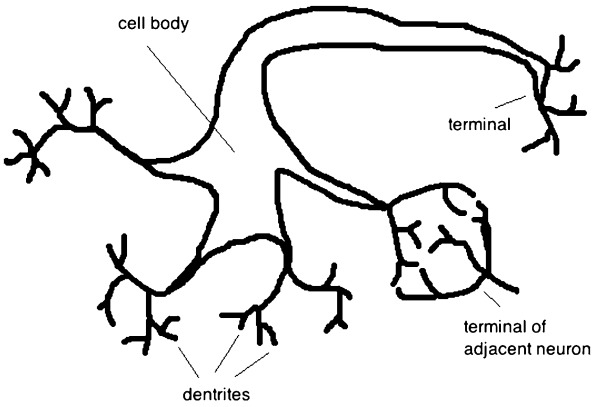

Fig. 11.1

cause a cell reaction then is to boost the external voltage to a sufficiently high level or to establish an input signal through the dendritic connections.

The Fitzhugh–Nagumo equations used here have been adapted from Brown and Rothery [1] and are

$$dV/dt = -V * (V - V1) * (V - V2) - W + E = \text{current}, \tag{11.1}$$

$$dW/dt = \text{EPSILON} * (V - C * W) \tag{11.2}$$

where V is the departure of the membrane potential from its equilibrium and W is the recovery current reflecting conductance of ions depending on voltage. These two variables are the state variables of the system. The amplifying threshold parameters V1 and V2 capture the influence of V on the rate of change of V and are held constant at 0.2 and 1.0, respectively. The parameter E reflects the electrical current to which the neuron is exposed.

Equations (11.1) and (11.2) can be combined into a single second order differential equation: the rate of change of velocity of the membrane potential voltage.

The amplifying character of the neuron is analogous to function of the transistor. The rate of change of the recovery variable W, defined in Eq. (11.2), is dependent on the difference between the departure of the membrane potential from its equilibrium V, and the recovery variable W that decays at a constant rate C. In our model, we arbitrarily set $C = 0.5$. The change in W is assumed to be proportional to $(V - C*W)$, with a proportionality factor of EPSILON. We set $\text{EPSILON} = 0.02$.

The STELLA model of the Fitzhugh–Nagumo equations is shown in Fig. 11.2. Note that in this STELLA model, the control flows are set up as biflows and the stocks are set to allow negative values. We run this model at a $DT = 0.1$, using the Euler integration method.

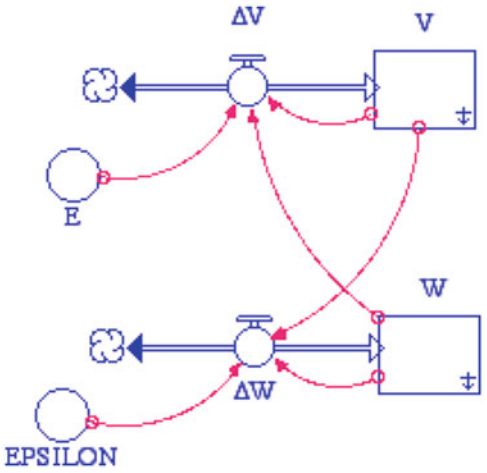

Fig. 11.2

If you set $V(t=0)=0$ and $W(t=0)=0$, both the membrane potential and recovery mechanism are at equilibrium. If you further set $E=0$, the cell does not respond because all potentials are zero.

When the neuron is exposed to either a dendritic voltage or an external potential, the cell membrane potential builds and "fires" an amplification of the combined received voltages. The dendritic signals are simulated by initial values of the membrane potential. The membrane potential response shows a critical threshold voltage potential is required. A quick examination of the differential equations shows that this amplification threshold is 0.2 (V1) and amplification ceases when the voltage exceeds 1.0 (V2)—our choice of parameters for the model.

When the dendritic potential is zero ($V(t=0)=0$, $W(t=0)=0$), the effect of applied voltages (E) can be seen. For currents over 0.23, the membrane potential cycles up and down. This result shows an unstable condition that also can be observed in physical experiments. The results for $E=0.23$ are shown in Fig. 11.3. This is the lowest applied voltage that will produce cycling. Can you find the value for E that leads to the largest amplification of the input signal?

Figure 11.4 the case of amplification of the dendritic signal: $E=0$, $V(t=0)=0.4$ and all other variables the same as above. The cell responds when the initial membrane potential is set to some positive potential. The membrane potential asymptotically approaches zero from its initial value, after rising, then overshooting zero and dropping below the equilibrium potential (zero on our scale). Try EPSILON $=0.002$ and note the different level of response. Demonstrate that the amplification range of the cell is actually between $0.2 \leq V(t=0) \leq 1.0$. Why do you suppose the nerve cell amplifies the combination of the input signals?

The first model run above showed cycling of V and W for $E \geq 0.23$. Keep increasing EPSILON for subsequent runs and observe the results. You will find

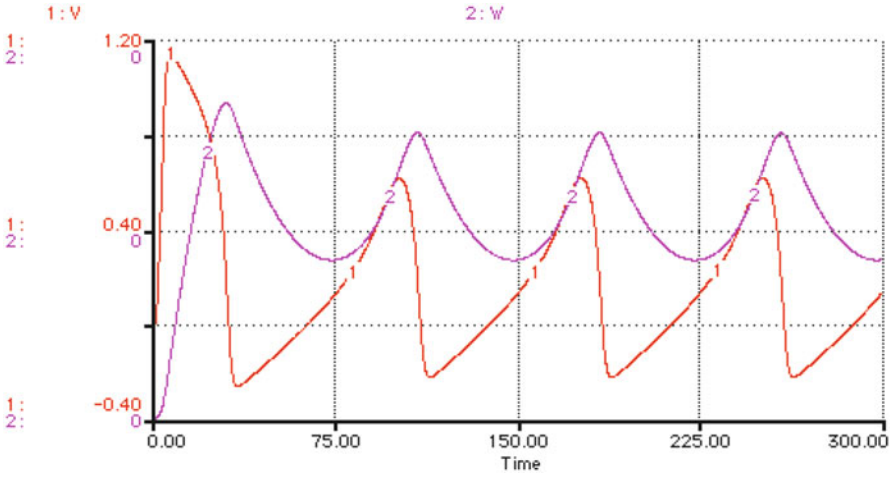

Fig. 11.3

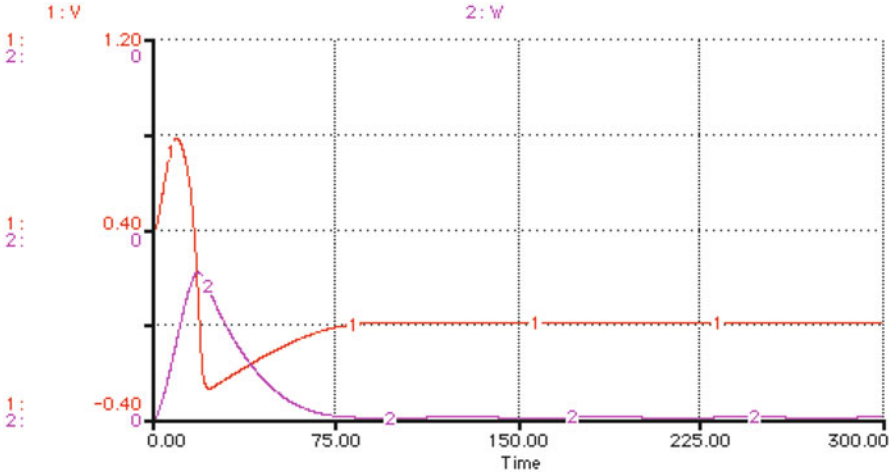

Fig. 11.4

that when the applied current in the model is greater than 1.3, cycling ceases, only to return again when EPSILON reaches the range of 25–32; first cycling returns and then chaos ensues. This chaos phenomenon is probably only of interest to modelers. Could the nerve cell actually become chaotic if it could withstand these apparently high currents? It depends on whether or not the actual functioning of the cell is discrete in time. If the cell has some operating period required below which no action takes place, then chaos of the membrane potential should be experimentally demonstrable.

Following this model of the behavior of an individual cell in response to the impulses from its environment, we will model the behavior of an entire organism in response to changes in its physical surroundings. This is the topic of the following chapter.

11.2 Fitzhugh–Nagumo Neuron Model Equations

V(t) = V(t − dt) + (ΔV) ∗ dt
INIT V = 0
INFLOWS:
ΔV = −V∗(V−.2)∗(V−1)−W+E

W(t) = W(t − dt) + (ΔW) ∗ dt
INIT W = 0
INFLOWS:
ΔW = EPSILON∗(V−.5∗W)

E = 0
EPSILON = .02

Reference

1. Brown D, Rothery P (1993) Models in biology: mathematics, statistics and computing. Wiley, Chichester, pp 320–326

Part III
Genetics Models

Chapter 12
Mating and Mutation of Alleles

There is such an unerring power at work, or Natural Selection, which selects exclusively for the good of each organic being.

(Darwin, Letters, 1857)

12.1 Model of Mating and Mutation of Alleles

Genetic theory provides us with insight into the changes of the genetic makeup of organisms. These insights are of particular interest in the context of conservation biology where genetic diversity is seen as one advantage of a species to survive in a changing environment. The more different genetic information is present, the better prepared a species to deal with a range of environmental factors.

Several processes influence the genetic makeup of organisms in a species. Among those influences is the combination of various alleles into a genotype from "parents" that carry those alleles. Another factor determining genetic makeup is the random mutations of one type of allele into another. These two cases are dealt with in the model of this chapter. The following chapter will then explore the impacts of natural selection and fitness in conjunction with mutation on genotype distribution.

To model the process of genotype mixing and mutation we restrict ourselves—without loss of generality—to the case of two alleles A and B, which are drawn randomly from an initial pool of 200 A alleles and 300 B alleles. The results of the simple mating of two alleles are explained by the Hardy–Weinberg law: this law states that the genotype frequencies are determined in a random mating process. These genotype frequencies are for AA, p^2; for AB, 2∗p∗q; for BB, q^2, where p and q are the A and B allele frequencies, respectively. In our sample problem, p = 200/500 or 0.4 and q = 300/500 or 0.6 for our initial pool of 200 A alleles and

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_12,

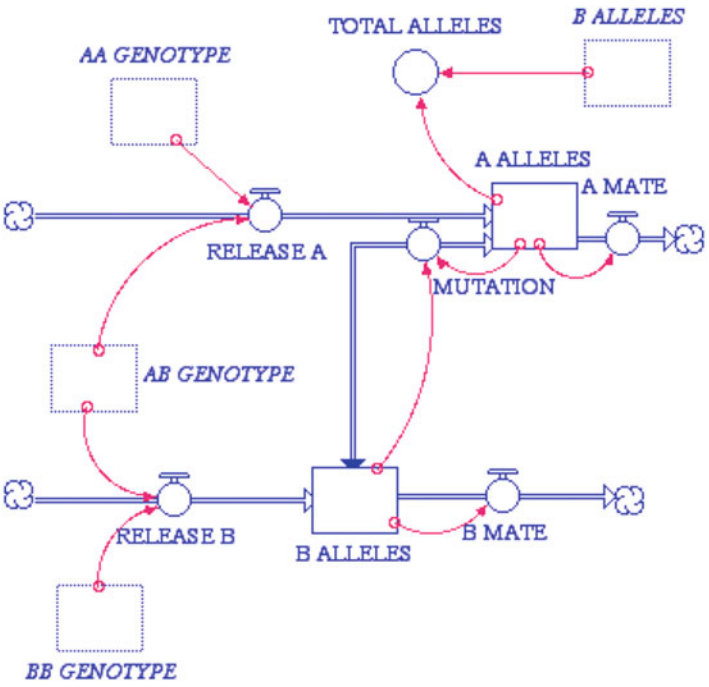

Fig. 12.1

300 B alleles. From 500 alleles we can have 250 genotypes, so the Hardy–Weinberg law tells us that we should end up with $0.4*0.4*250$ or 40 AA genotypes, $2*0.4*0.6*250$ or 120 AB genotypes and finally, $0.6*0.6*250$ or 90 BB genotypes.

Of course, the allele frequencies change from generation to generation as mutation occurs. Here, we assume that mutation is a random process and can occur in both directions. For the model we assume that a random fraction of A alleles turns into B alleles and vice versa, and we only deal with the net of the mutation in both directions. We define

$$\begin{aligned}\text{NET MUTATION} = {} & \text{RANDOM}\,(-0.05, 0.05) \\ & * (\text{A ALLELES} + \text{B ALLELES})\end{aligned} \tag{12.1}$$

where A ALLELES and B ALLELES are the stocks of alleles of each type that are available for mating to form the next generation of genotypes. Each of these stocks is emptied when a new generation of genotypes is formed. We call the respective outflows A MATE and B MATE. The inflows into the stocks are based on the alleles that have been temporarily "tied up" in genotypes. For example, one AA GENOTYPE releases two A alleles and one AB GENOTYPE releases only one A allele. As alleles are dumped into the respective stocks, mutation occurs as specified in Eq. (12.1). The STELLA model that deals with this part of the process of allele mixing and mutation is shown in Fig. 12.1.

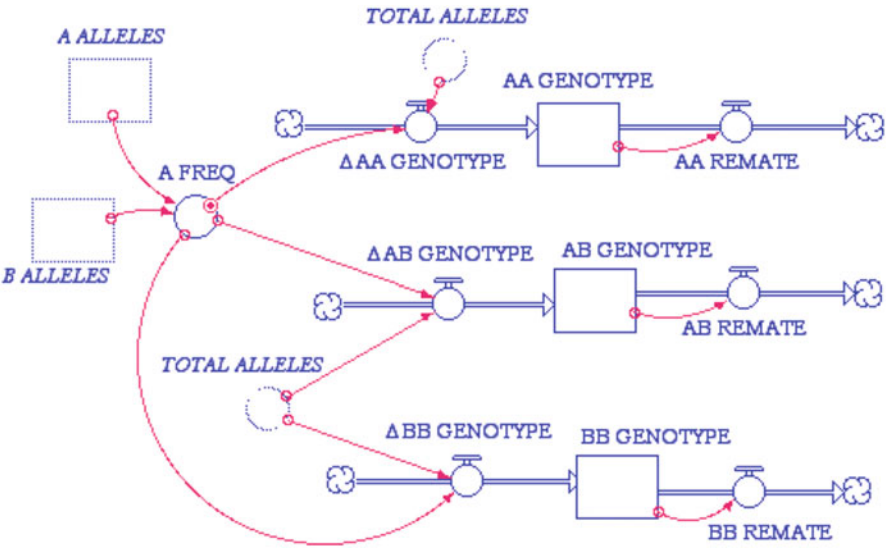

Fig. 12.2

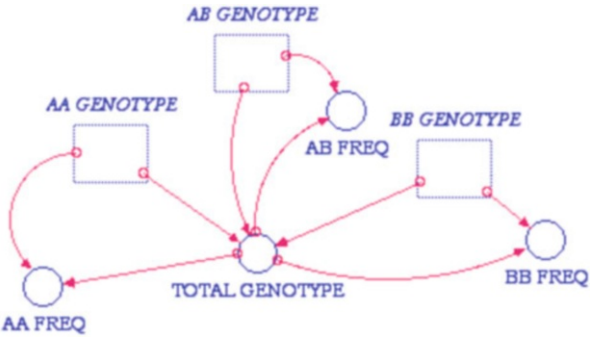

Fig. 12.3

The alleles are combined according to the Hardy–Weinberg law. The Hardy–Weinberg law is used here to calculate the inflows into the stocks of the three genotypes (Fig. 12.2). The outflows are set equal to the stocks, reflecting the fact that after a generation was formed the will be available again for re-mating.

Figure 12.3 shows the calculation of the relative frequencies of the genotypes. These are plotted in Fig. 12.4. The results of the model are different from run to run because of the random mutation process that takes place. What will happen when the total number of alleles decreases? Run the model several times and observe the results. How do the results change if mutation is more likely to occur in one direction than another? What are the implications of a decreasing number of alleles

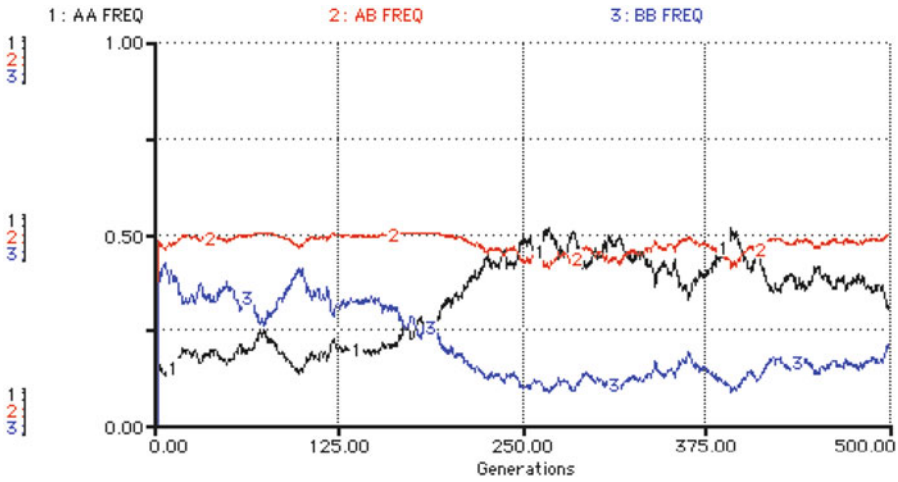

Fig. 12.4

and asymmetric mutation for conservation biology? Can you set up the problem without making explicit use of the Hardy–Weinberg law?

Use this model to introduce survival and fertility rates for the genotypes and combine these rates to form a fitness measure. Together with mutation these effects consort to give the new genotype frequencies. Geneticists and conservation biologist think the existence of the heterozygote, the AB genotype, is a measure of health of the system of genotypes. The strength of its presence is sometimes referred to as "Hybrid vigor" and is obviously desired as it carries both alleles.

12.2 Mating and Mutation of Alleles Model Equations

AA_GENOTYPE(t) = AA_GENOTYPE(t − dt) + (Δ_AA_GENOTYPE − AA_REMATE) ∗ dt
INIT AA_GENOTYPE = 0

INFLOWS:
Δ_AA_GENOTYPE = TOTAL_ALLELES/2∗A_FREQ^2
OUTFLOWS:
AA_REMATE = AA_GENOTYPE
AB_GENOTYPE(t) = AB_GENOTYPE(t − dt) + (Δ_AB_GENOTYPE − AB_REMATE) ∗ dt
INIT AB_GENOTYPE = 0

INFLOWS:
Δ_AB_GENOTYPE = 2∗TOTAL_ALLELES/2∗A_FREQ∗(1 − A_FREQ)

OUTFLOWS:
AB_REMATE = AB_GENOTYPE
A_ALLELES(t) = A_ALLELES(t − dt) + (RELEASE_A + MUTATION − A_MATE) ∗ dt
INIT A_ALLELES = 200

INFLOWS:
RELEASE_A = 2∗AA_GENOTYPE +AB_GENOTYPE
MUTATION = RANDOM(−0.05,0.05)∗(A_ALLELES+B_ALLELES)
OUTFLOWS:
A_MATE = A_ALLELES
BB_GENOTYPE(t) = BB_GENOTYPE(t − dt) + (Δ_BB_GENOTYPE − BB_REMATE) ∗ dt
INIT BB_GENOTYPE = 0

INFLOWS:
Δ_BB_GENOTYPE = TOTAL_ALLELES/2∗(1 − A_FREQ)^2
OUTFLOWS:
BB_REMATE = BB_GENOTYPE
B_ALLELES(t) = B_ALLELES(t − dt) + (RELEASE_B − B_MATE − MUTATION) ∗ dt
INIT B_ALLELES = 300

INFLOWS:
RELEASE_B = 2∗BB_GENOTYPE + AB_GENOTYPE
OUTFLOWS:
B_MATE = B_ALLELES
MUTATION = RANDOM(−0.05,0.05)∗(A_ALLELES+B_ALLELES)
AA_FREQ = IF TIME > 0 THEN AA_GENOTYPE/TOTAL_GENOTYPE ELSE 0
AB_FREQ = IF TIME > 0 THEN AB_GENOTYPE/TOTAL_GENOTYPE ELSE 0
A_FREQ = A_ALLELES/(A_ALLELES+ B_ALLELES)
BB_FREQ = IF TIME > 0 THEN BB_GENOTYPE/TOTAL_GENOTYPE ELSE 0
TOTAL_ALLELES = A_ALLELES + B_ALLELES
TOTAL_GENOTYPE = AA_GENOTYPE+AB_GENOTYPE+BB_GENOTYPE

Chapter 13
Artificial Worms

Any intelligent fool can make things bigger, more complex, and more violent. It takes a touch of genius – and a lot of courage – to move in the opposite direction.

(Albert Einstein)

13.1 Artificial Worm Model

Natural selection works in the presence of chance or randomness. As random mutations take place, the form or function of organisms changes. With changes in the "appearance" of individual organisms may come an enhanced ability to survive and to pass on the respective traits to subsequent generations. However, because mutations continue to take place, and because the environment within which natural selection occurs is not constant, it is not automatic that the fitness of offspring increases from one generation to the next.

The following model—briefly mentioned in Cohen and Stewart [1]—illustrates the workings of natural selection and randomness for a population of six worms of different length. Each individual worm can, in principle, mate with any one of the others. The decision of who mates with whom is randomly made. Of the three randomly chosen pairs of mating worms, only the longer one of the two will survive to bear exactly two offspring. The offspring, in turn, are either shorter or longer (randomly decided) than their parent. Their actual length is determined by a coin toss. If the coin shows head, one of the offspring will be exactly 1 unit longer than the parent, and the other one will be exactly 2 units longer than the parent. In the case that the coin shows tail, the two offspring are, respectively, 1 and 2 units shorter than the parent. The moment the offspring are generated, the parent dies.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_13,

As a result, a total of six new worms are present whose length may differ from that of their six parents. Will the length of the new generations of worms be similar to that of their parents, or will over generations worms get longer or shorter?

Because of the randomness that determines the choice of mating partners and the randomness that determines the length of the offspring we do not know in advance how long each of the next-generation worms will be. What we do know, however, is that of the pairs that mate, only the longer one survives. So, a mechanism is built in to the dynamics of this artificial, evolutionary process that favors longer worms, and we would therefore expect that as time passes, later-born worms are on average longer than their predecessors.

In order to model the evolution of our worms, note that we need to solve three interrelated problems. First, we need to randomly pair the worms. Then we need to decide who in each pair is longer, and "kill" the shorter ones off. Third, we need to toss a coin to determine the length of the offspring and add these new worms to our system while we remove the information about their parents, i.e. let the remaining three parents die.

The first of these three modeling problems can be solved by assuming that we cast six dice such that at the end all six dice show a different number. Then we always pair the same dice—the first die with the second, the third with the fourth, and the fifth with the sixth. For example, if the dice give the numbers 2, 4, 1, 6, 3, 5, then we take this to mean that we should mate WORM 2 with WORM 4, WORM 1 with WORM 6, and WORM 3 with WORM 5. To make things easier, we may actually fix one die—say the first one—and cast only the other five die. That little trick still leaves the process perfectly random, because any time five die are cast such that their numbers are different from each other, the number for the sixth is automatically determined anyway. We may as well start with that one. Here, we arbitrarily set it to 1.

Once all dice have been cast such that they each show a different number, their sum is 21 ($1+2+3+4+5+6=21$). At that time, the worms are officially paired and can begin their fight to the death (with the winner reproducing). Then the system is re-set and the stocks, which contain the results of the round of casting the dice, are emptied, and the dice are cast anew.

The dice of this model are represented as stocks named STOCK 1, STOCK 2, etc., whose contents are held constant until all dice show random numbers between 2 and 6, and all are different from each other. The exception is the first die, STOCK 1, whose value is fixed at 1.

Take, for example, the second die. The converter DIE 2 in Fig. 13.1 generates a random number between 1 and 7:

$$\text{DIE 2} = \text{INT}(\text{RANDOM}(1,7)) \tag{13.1}$$

By only taking the integer of the random number we generate a string of numbers between 1 and 6. The odds that exactly a 7 is generated are minuscule. Next, we need to make sure that we only accept a number that has not already been cast by another die—we want to avoid mating one worm with itself. We do this, for

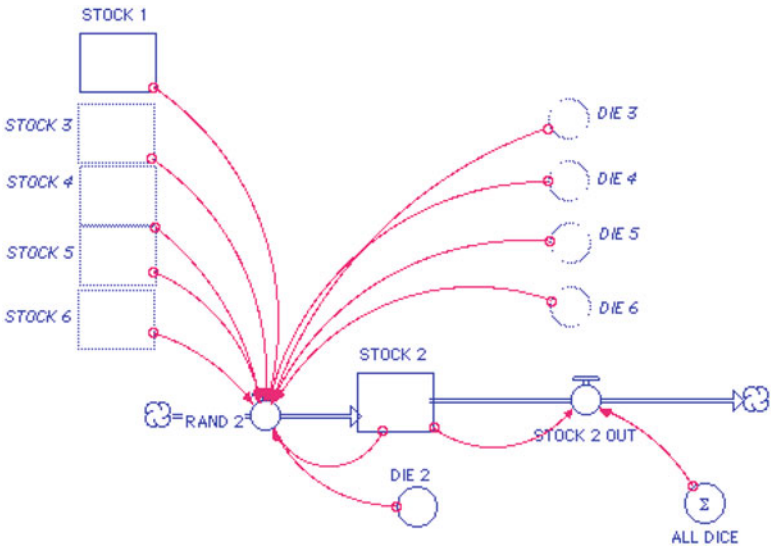

Fig. 13.1

example, for DIE 2, with the following conditional statement which states that as long as STOCK 2 is zero and as long as DIE 2 shows a different number from those recorded in the other stocks, we can accept that die's number. If not, we continue to cast that die until its number differs from any of the others, and we then add it to the stock.

$$\begin{aligned}\text{RAND 2} = \text{IF } (&\text{STOCK 2} = 0 \text{ AND DIE 2} \neq \text{STOCK 1 AND DIE 2} \neq \text{STOCK 3}\\ &\text{AND DIE 2} \neq \text{STOCK 4 AND DIE 2} \neq \text{STOCK 5 AND DIE 2} \neq \text{STOCK 6}\\ &\text{AND DIE 2} \neq \text{DIE 6 AND DIE 2} \neq \text{DIE 5 AND DIE 2} \neq \text{DIE 4}\\ &\text{AND DIE 2} \neq \text{DIE 3})\\ &\text{THEN DIE 2 ELSE 0}\end{aligned} \tag{13.2}$$

Once all dice have been cast such that they each show a different number, their sum is 21. We calculate that sum in the ALL DICE converter, using STELLA's "Summing Converter" option. Once all dice show a different number all worms are randomly, yet unambiguously, paired with each other to begin their fight to the death. The system is re-set by emptying STOCK 2, STOCK 3 etc., and the dice are cast anew.

$$\text{STOCK 2 OUT} = \text{IF ALL DICE} = 21 \text{ THEN STOCK 2 ELSE 0} \tag{13.3}$$

So far, we have only concerned ourselves with the means by which we can randomly pair up the six worms. But what about their length? To keep track of

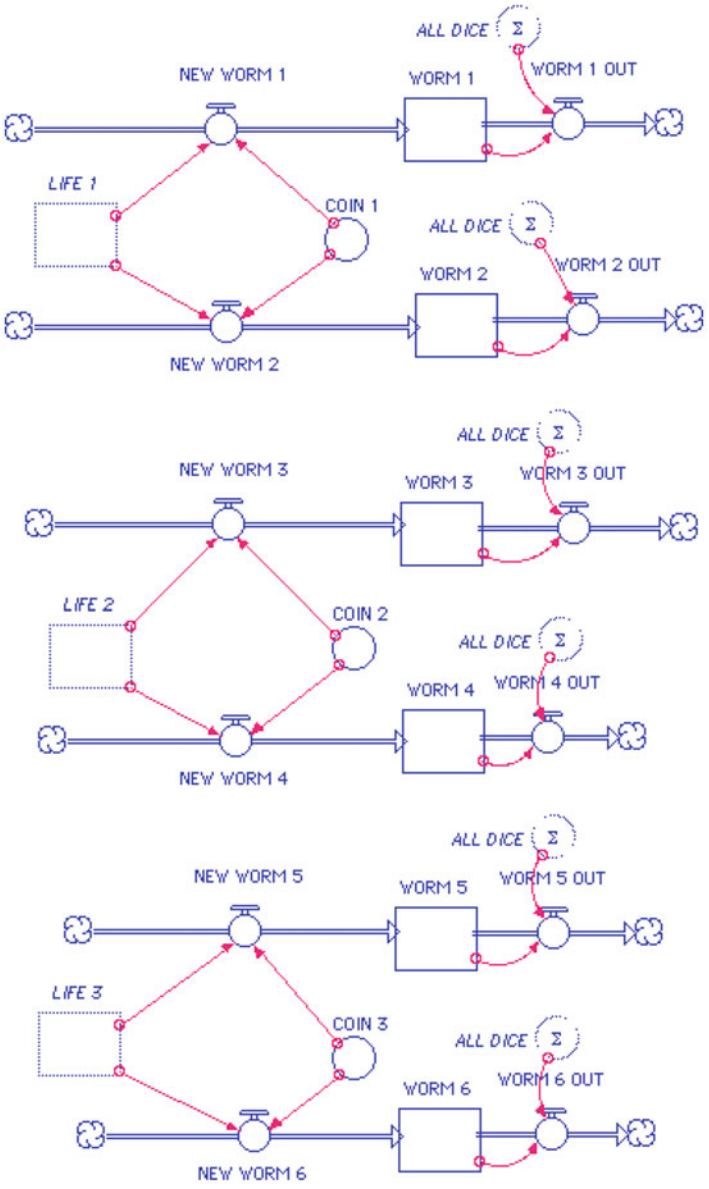

Fig. 13.2

worm length, we generate six new state variables, and we assign them some arbitrary initial values for the lengths of the worms. For simplicity, we set here the initial values of WORM 1 = 1, WORM 2 = 2, WORM 3 = 3, and so on.

Note in Fig. 13.2 how we set up the model structure such that we always update the same pair of stocks for the worms. Who these worms are, however, is

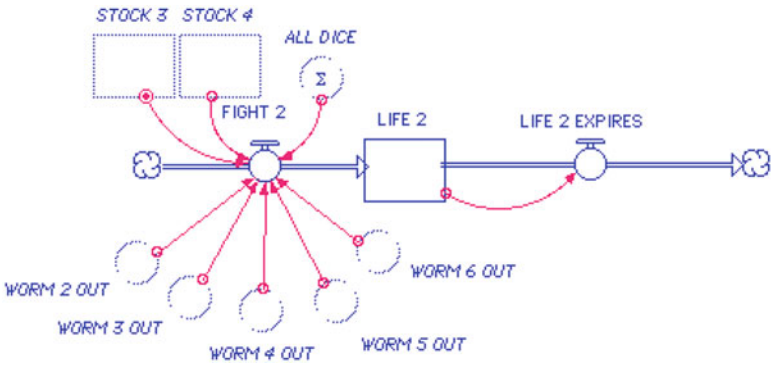

Fig. 13.3

determined by casting the dice—it is not as fixed as the graphical model representation may suggest. For example, WORM 3 and WORM 4 are always those worms that were identified as the first and second worms in the second pair of worms that is formed for the mating process, but from one round of casting dice to the next, these are different worms whose lengths may differ from those of their parents.

Outflows from the WORM 1, WORM 2, etc. stocks occur once all dice are cast, i.e. when ALL DICE = 21. Those worms are now ready to fight for their life and then generate offspring. For example, the outflow from the WORM 1 stock is

$$\text{WORM 1 OUT} = \text{IF ALL DICE} = 21 \text{ THEN WORM 1 ELSE } 0 \qquad (13.4)$$

The inflows are determined on the basis of a coin toss and who of the two worms in a pair survived the fight for life. We will discuss the inflows into WORM 1, WORM 2, etc. below, after we dealt with the processes by which we determine the winners of the fights and update the respective other stocks in our model.

Take, for example, the second pair of worms—the pair that got determined by tossing the third and fourth dice (Fig. 13.3). The stock LIFE 2 retains information about the length of the winner of a fight. That stock is initially set to zero, as are LIFE 1 and LIFE 3—the winners of the other two fights.

The inflows into the stocks LIFE 1, LIFE 2, and LIFE 3 are called FIGHT 1, FIGHT 2, and FIGHT 3, and they contain the guts of this model. Here, we check which of the worms got matched up in the dice-casting part of this model and then compare their lengths. For example, if the third die showed the number 2 and the fourth die a number 3, then we know that the second and third worm were paired up, and we will compute the maximum of the lengths of the two. But if the third die showed a number 2 and the second a number 4, then WORM 2 and WORM 4 were paired up and their lengths need to be compared. We repeat this process until we exhaust all combinatorial possibilities. If none of the possibilities arose, that is a sign that the process of pairing the worms by casting the dice has not yet been completed. In this case the value of FLIGHT 2 is set to zero and the stock LIFE 2 remains unchanged, as shown in equation 13.5:

```
FIGHT 2 = IF (STOCK 3=2 AND STOCK 4=3 AND ALL DICE=21)
        THEN MAX(WORM 2 OUT,WORM 3 OUT)
        ELSE IF (STOCK 3=2 AND STOCK 4=4 AND ALL DICE=21)
        THEN MAX(WORM 2 OUT,WORM 4_OUT)
        ELSE IF (STOCK 3=2 AND STOCK 4=5 AND ALL DICE=21)
        THEN MAX(WORM 2 OUT,WORM 5 OUT)
        ELSE IF (STOCK 3=2 AND STOCK 4=6 AND ALL DICE=21)
        THEN MAX(WORM 2 OUT,WORM 6 OUT)
        ELSE IF (STOCK 3=3 AND STOCK 4=2 AND ALL DICE=21)
        THEN MAX(WORM 3 OUT,WORM 2 OUT)
        ELSE IF (STOCK 3=3 AND STOCK 4=4 AND ALL DICE=21)
        THEN MAX(WORM 3 OUT,WORM 4 OUT)
        ELSE IF (STOCK 3=3 AND STOCK 4=5 AND ALL DICE=21)
        THEN MAX(WORM 3 OUT,WORM 5 OUT)
        ELSE IF (STOCK 3=3 AND STOCK 4=6 AND ALL DICE=21)
        THEN MAX(WORM 3 OUT,WORM 6 OUT)
        ELSE IF (STOCK 3=4 AND STOCK 4=2 AND ALL DICE=21)
        THEN MAX(WORM 4 OUT,WORM 2 OUT)
        ELSE IF (STOCK 3=4 AND STOCK 4=3 AND ALL DICE=21)
        THEN MAX(WORM 4 OUT,WORM 3 OUT)
        ELSE IF (STOCK 3=4 AND STOCK 4=5 AND ALL DICE=21)
        THEN MAX(WORM 4 OUT,WORM 5 OUT)
        ELSE IF (STOCK 3=4 AND STOCK 4=6 AND ALL DICE=21)
        THEN MAX(WORM 4 OUT,WORM 6 OUT)
        ELSE IF (STOCK 3=5 AND STOCK 4=2 AND ALL DICE=21)
        THEN MAX(WORM 5 OUT,WORM 2 OUT)
        ELSE IF (STOCK 3=5 AND STOCK 4=3 AND ALL DICE=21)
        THEN MAX(WORM 5 OUT,WORM 3 OUT)
        ELSE IF (STOCK 3=5 AND STOCK 4=4 AND ALL DICE=21)
        THEN MAX(WORM 5 OUT,WORM 4 OUT)
        ELSE IF (STOCK 3=5 AND STOCK 4=6 AND ALL DICE=21)
        THEN MAX(WORM 5 OUT,WORM 6 OUT)
        ELSE IF (STOCK 3=6 AND STOCK 4=2 AND ALL DICE=21)
        THEN MAX(WORM 6 OUT,WORM 2 OUT)
        ELSE IF (STOCK 3=6 AND STOCK 4=3 AND ALL DICE=21)
        THEN MAX(WORM 6 OUT,WORM 3 OUT)
        ELSE IF (STOCK 3=6 AND STOCK 4=4 AND ALL DICE=21)
        THEN MAX(WORM 6 OUT,WORM 4 OUT)
        ELSE IF (STOCK 3=6 AND STOCK 4=5 AND ALL DICE=21)
        THEN MAX(WORM 6 OUT,WORM 5 OUT)
        ELSE 0
```

(13.5)

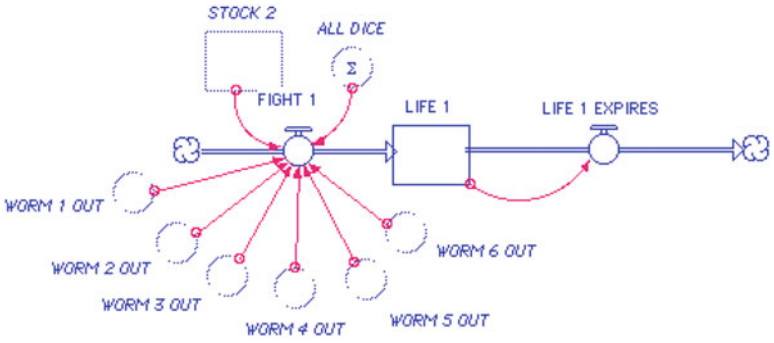

Fig. 13.4

The FIGHT 3 flow is analogous to the one for FIGHT 2, but both of these differ slightly from the FIGHT 1 flow (Fig. 13.4). To calculate FIGHT 1 we do not need a ghost of STOCK 1, because we always know the value of that stock—earlier we set it to 1.

Once we calculated the length of the winner of the fights in each pair of worms, that worm will die, too. We model the winner's death with the outflows from the respective LIFE stocks, such as

$$\text{LIFE 2 EXPIRES} = \text{LIFE 2}. \tag{13.6}$$

The LIFE 1, LIFE 2, and LIFE 3 stocks are used to calculate the lengths of the next generation of worms. For each pair of offspring we flip a coin, as we have done in previous chapters:

$$\text{COIN 1} = \text{RANDOM}(0, 1) \tag{13.7}$$

$$\text{COIN 2} = \text{RANDOM}(0, 1) \tag{13.8}$$

$$\text{COIN 3} = \text{RANDOM}(0, 1) \tag{13.9}$$

Then we compute the length of the new worms based on the outcome of the coin toss. We interpret numbers below .5 as tail and in that case make the length of one new worm 1 unit less than that of the winner of the fight, and we make the other offspring 2 units shorter. For numbers above .5 we make the offspring 1 and 2 units longer, respectively. For example,

$$\begin{aligned}\text{NEW WORM 1} = \text{IF } &(\text{LIFE 1} > 0 \text{ AND COIN 1} < .5) \text{ THEN LIFE 1} - 1 \\ &\text{ELSE IF } (\text{LIFE 1} > 0 \text{ AND COIN 1} > .5) \text{ THEN LIFE 1} + 1 \\ &\text{ELSE 0}\end{aligned} \tag{13.10}$$

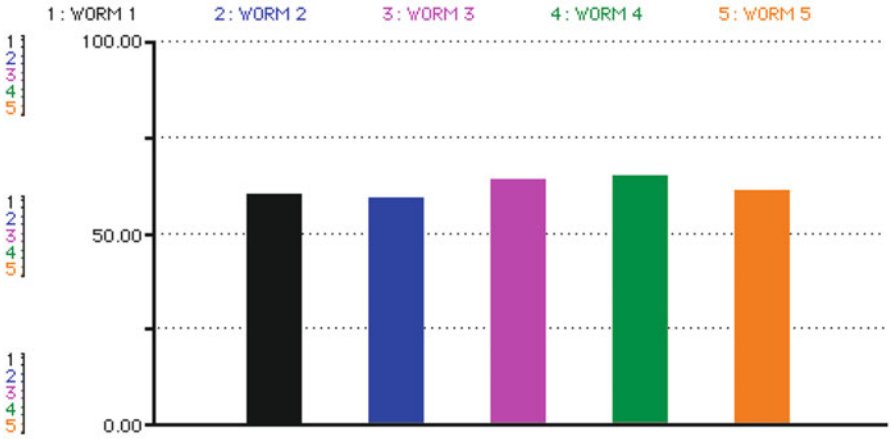

Fig. 13.5

$$\begin{aligned}\text{NEW WORM 2} = \text{IF } &(\text{LIFE 1} > 0 \text{ AND COIN 1} < .5) \text{ THEN LIFE 1} - 2 \\ &\text{ELSE IF } (\text{LIFE 1} > 0 \text{ AND COIN 1} > .5) \text{ THEN LIFE 1} + 2 \\ &\text{ELSE 0}\end{aligned} \tag{13.11}$$

Now our model is finished. We have modules that we use to generate random pairs of worms, to compare their lengths and let the longer one win a fight, and to generate offspring whose lengths depend on the length of the winner in each fight. Figure 13.5 shows in the form of a bar chart the result of the evolution of our worms after running the model for 1,000 periods. To generate a bar chart, simply double-click on an open graph pad and select "Bar." Only a maximum of five variables can be plotted.

Note that the length of the model run should not be interpreted in terms of the number of generations of worms. At times, the process of casting dice leads immediately to numbers that are different from each other for all six dice. At that moment, a new generation of worms can be formed, and only in this case is one time period equal to the length of one worm generation. If the dice are cast and not all dice show different numbers, we need to continue to cast them until they do. This may take several periods in the model and, as a consequence, the new generation of worms will be formed after more than one period in the model.

Of course in the world of laboratory and field experiments, the most likely picture we would have is a seasonal or breeding cycle study of the length of worms. In actual measurement of the worms we would get just one of the pictures in the series that compose Fig. 13.5. We would categorize the lengths in the field most likely in terms of mean length and standard deviation. Our model allows one to actually compute these values continuously and thus provides a way to validate the model. In the absence of such data, we have a model here that provides a dynamic example of natural selection dynamics.

13.2 Artificial Worm Model Equations

LIFE_1(t) = LIFE_1(t − dt) + (FIGHT_1 − LIFE_1_EXPIRES) ∗ dt
INIT LIFE_1 = 0
INFLOWS:
FIGHT_1 = IF (STOCK_2 = 2 AND ALL_DICE= 21) THEN MAX (WORM_1_OUT,WORM_2_OUT)
ELSE IF (STOCK_2 = 3 AND ALL_DICE= 21) THEN MAX(WORM_1_OUT, WORM_3_OUT)
ELSE IF (STOCK_2 = 4 AND ALL_DICE= 21) THEN MAX(WORM_1_OUT, WORM_4_OUT)
ELSE IF (STOCK_2 = 5 AND ALL_DICE= 21) THEN MAX(WORM_1_OUT, WORM_5_OUT)
ELSE IF (STOCK_2 = 6 AND ALL_DICE= 21) THEN MAX(WORM_1_OUT, WORM_6_OUT)
ELSE 0
OUTFLOWS:
LIFE_1_EXPIRES = LIFE_1

LIFE_2(t) = LIFE_2(t − dt) + (FIGHT_2 - LIFE_2_EXPIRES) ∗ dt
INIT LIFE_2 = 0
INFLOWS:
FIGHT_2 = IF (STOCK_3=2 AND STOCK_4=3 AND ALL_DICE=21) THEN MAX(WORM_2_OUT,WORM_3_OUT)
ELSE IF (STOCK_3=2 AND STOCK_4=4 AND ALL_DICE=21) THEN MAX (WORM_2_OUT,WORM_4_OUT)
ELSE IF (STOCK_3=2 AND STOCK_4=5 AND ALL_DICE=21) THEN MAX (WORM_2_OUT,WORM_5_OUT)
ELSE IF (STOCK_3=2 AND STOCK_4=6 AND ALL_DICE=21) THEN MAX (WORM_2_OUT,WORM_6_OUT)
ELSE IF (STOCK_3=3 AND STOCK_4=2 AND ALL_DICE=21) THEN MAX (WORM_3_OUT,WORM_2_OUT)
ELSE IF (STOCK_3=3 AND STOCK_4=4 AND ALL_DICE=21) THEN MAX (WORM_3_OUT,WORM_4_OUT)
ELSE IF (STOCK_3=3 AND STOCK_4=5 AND ALL_DICE=21) THEN MAX (WORM_3_OUT,WORM_5_OUT)
ELSE IF (STOCK_3=3 AND STOCK_4=6 AND ALL_DICE=21) THEN MAX (WORM_3_OUT,WORM_6_OUT)
ELSE IF (STOCK_3=4 AND STOCK_4=2 AND ALL_DICE=21) THEN MAX (WORM_4_OUT,WORM_2_OUT)
ELSE IF (STOCK_3=4 AND STOCK_4=3 AND ALL_DICE=21) THEN MAX (WORM_4_OUT,WORM_3_OUT)
ELSE IF (STOCK_3=4 AND STOCK_4=5 AND ALL_DICE=21) THEN MAX (WORM_4_OUT,WORM_5_OUT)

```
ELSE IF (STOCK_3=4 AND STOCK_4=6 AND ALL_DICE=21) THEN MAX
   (WORM_4_OUT,WORM_6_OUT)
ELSE IF (STOCK_3=5 AND STOCK_4=2 AND ALL_DICE=21) THEN MAX
   (WORM_5_OUT,WORM_2_OUT)
ELSE IF (STOCK_3=5 AND STOCK_4=3 AND ALL_DICE=21) THEN MAX
   (WORM_5_OUT,WORM_3_OUT)
ELSE IF (STOCK_3=5 AND STOCK_4=4 AND ALL_DICE=21) THEN MAX
   (WORM_5_OUT,WORM_4_OUT)
ELSE IF (STOCK_3=5 AND STOCK_4=6 AND ALL_DICE=21) THEN MAX
   (WORM_5_OUT,WORM_6_OUT)
ELSE IF (STOCK_3=6 AND STOCK_4=2 AND ALL_DICE=21) THEN MAX
   (WORM_6_OUT,WORM_2_OUT)
ELSE IF (STOCK_3=6 AND STOCK_4=3 AND ALL_DICE=21) THEN MAX
   (WORM_6_OUT,WORM_3_OUT)
ELSE IF (STOCK_3=6 AND STOCK_4=4 AND ALL_DICE=21) THEN MAX
   (WORM_6_OUT,WORM_4_OUT)
ELSE IF (STOCK_3=6 AND STOCK_4=5 AND ALL_DICE=21) THEN MAX
   (WORM_6_OUT,WORM_5_OUT)
ELSE 0
OUTFLOWS:
LIFE_2_EXPIRES = LIFE_2

LIFE_3(t) = LIFE_3(t − dt) + (FIGHT_3 - LIFE_3_EXPIRES) ∗ dt
INIT LIFE_3 = 0
INFLOWS:
FIGHT_3 = IF (STOCK_5=2 AND STOCK_6=3 AND ALL_DICE=21) THEN
   MAX(WORM_2_OUT,WORM_3_OUT)
ELSE IF (STOCK_5=2 AND STOCK_6=4 AND ALL_DICE=21) THEN MAX
   (WORM_2_OUT,WORM_4_OUT)
ELSE IF (STOCK_5=2 AND STOCK_6=5 AND ALL_DICE=21) THEN MAX
   (WORM_2_OUT,WORM_5_OUT)
ELSE IF (STOCK_5=2 AND STOCK_6=6 AND ALL_DICE=21) THEN MAX
   (WORM_2_OUT,WORM_6_OUT)
ELSE IF (STOCK_5=3 AND STOCK_6=2 AND ALL_DICE=21) THEN MAX
   (WORM_3_OUT,WORM_2_OUT)
ELSE IF (STOCK_5=3 AND STOCK_6=4 AND ALL_DICE=21) THEN MAX
   (WORM_3_OUT,WORM_4_OUT)
ELSE IF (STOCK_5=3 AND STOCK_6=5 AND ALL_DICE=21) THEN MAX
   (WORM_3_OUT,WORM_5_OUT)
ELSE IF (STOCK_5=3 AND STOCK_6=6 AND ALL_DICE=21) THEN MAX
   (WORM_3_OUT,WORM_6_OUT)
ELSE IF (STOCK_5=4 AND STOCK_6=2 AND ALL_DICE=21) THEN MAX
   (WORM_4_OUT,WORM_2_OUT)
ELSE IF (STOCK_5=4 AND STOCK_6=3 AND ALL_DICE=21) THEN MAX
   (WORM_4_OUT,WORM_3_OUT)
```

ELSE IF (STOCK_5=4 AND STOCK_6=5 AND ALL_DICE=21) THEN MAX (WORM_4_OUT,WORM_5_OUT)
ELSE IF (STOCK_5=4 AND STOCK_6=6 AND ALL_DICE=21) THEN MAX (WORM_4_OUT,WORM_6_OUT)
ELSE IF (STOCK_5=5 AND STOCK_6=2 AND ALL_DICE=21) THEN MAX (WORM_5_OUT,WORM_2_OUT)
ELSE IF (STOCK_5=5 AND STOCK_6=3 AND ALL_DICE=21) THEN MAX (WORM_5_OUT,WORM_3_OUT)
ELSE IF (STOCK_5=5 AND STOCK_6=4 AND ALL_DICE=21) THEN MAX (WORM_5_OUT,WORM_4_OUT)
ELSE IF (STOCK_5=5 AND STOCK_6=6 AND ALL_DICE=21) THEN MAX (WORM_5_OUT,WORM_6_OUT)
ELSE IF (STOCK_5=6 AND STOCK_6=2 AND ALL_DICE=21) THEN MAX (WORM_6_OUT,WORM_2_OUT)
ELSE IF (STOCK_5=6 AND STOCK_6=3 AND ALL_DICE=21) THEN MAX (WORM_6_OUT,WORM_3_OUT)
ELSE IF (STOCK_5=6 AND STOCK_6=4 AND ALL_DICE=21) THEN MAX (WORM_6_OUT,WORM_4_OUT)
ELSE IF (STOCK_5=6 AND STOCK_6=5 AND ALL_DICE=21) THEN MAX (WORM_6_OUT,WORM_5_OUT)
ELSE 0
OUTFLOWS:
LIFE_3_EXPIRES = LIFE_3

STOCK_1(t) = STOCK_1(t − dt)
INIT STOCK_1 = 1

STOCK_2(t) = STOCK_2(t − dt) + (RAND_2 − STOCK_2_OUT) ∗ dt
INIT STOCK_2 = 0
INFLOWS:
RAND_2 = IF (STOCK_2 = 0 AND DIE_2 <> STOCK_1 AND DIE_2<> STOCK_3 AND DIE_2<>STOCK_4 AND DIE_2<>STOCK_5 AND DIE_2<>STOCK_6 AND DIE_2<>DIE_6 AND DIE_2<>DIE_5 AND DIE_2<>DIE_4 AND DIE_2<>DIE_3) THEN DIE_2 ELSE 0
OUTFLOWS:
STOCK_2_OUT = IF ALL_DICE = 21 THEN STOCK_2 ELSE 0

STOCK_3(t) = STOCK_3(t − dt) + (RAND_3 − STOCK_3_OUT) ∗ dt
INIT STOCK_3 = 0
INFLOWS:
RAND_3 = IF (STOCK_3 = 0 AND DIE_3 <> STOCK_1 AND DIE_3 <>STOCK_2 AND DIE_3<>STOCK_4 AND DIE_3<>STOCK_5 AND DIE_3<>STOCK_6 AND DIE_3<>DIE_6 AND DIE_3<>DIE_5 AND DIE_3<>DIE_4 AND DIE_3<>DIE_2) THEN DIE_3 ELSE 0

OUTFLOWS:
STOCK_3_OUT = IF ALL_DICE = 21 THEN STOCK_3 ELSE 0

STOCK_4(t) = STOCK_4(t − dt) + (RAND_4 − STOCK_4_OUT) ∗ dt
INIT STOCK_4 = 0
INFLOWS:
RAND_4 = IF (STOCK_4 = 0 AND DIE_4 <> STOCK_1 AND DIE_4 <>STOCK_2 AND DIE_4<>STOCK_3 AND DIE_4<>STOCK_5 AND DIE_4<>STOCK_6 AND DIE_4<>DIE_6 AND DIE_4<>DIE_5 AND DIE_4<>DIE_3 AND DIE_4<>DIE_2) THEN DIE_4 ELSE 0
OUTFLOWS:
STOCK_4_OUT = IF ALL_DICE = 21 THEN STOCK_4 ELSE 0

STOCK_5(t) = STOCK_5(t − dt) + (RAND_5 − STOCK_5_OUT) ∗ dt
INIT STOCK_5 = 0
INFLOWS:
RAND_5 = IF (STOCK_5 = 0 AND DIE_5 <> STOCK_1 AND DIE_5 <>STOCK_2 AND DIE_5<>STOCK_3 AND DIE_5<>STOCK_4 AND DIE_5<>STOCK_6 AND DIE_5<>DIE_6 AND DIE_5<>DIE_4 AND DIE_5<>DIE_3 AND DIE_5<>DIE_2) THEN DIE_5 ELSE 0
OUTFLOWS:
STOCK_5_OUT = IF ALL_DICE = 21 THEN STOCK_5 ELSE 0

STOCK_6(t) = STOCK_6(t − dt) + (RAND_6 − STOCK_6_OUT) ∗ dt
INIT STOCK_6 = 0
INFLOWS:
RAND_6 = IF (STOCK_6 = 0 AND DIE_6 <> STOCK_1 AND DIE_6 <>STOCK_2 AND DIE_6<>STOCK_3 AND DIE_6<>STOCK_4 AND DIE_6<>STOCK_5 AND DIE_6<>DIE_5 AND DIE_6<>DIE_4 AND DIE_6<>DIE_3 AND DIE_6<>DIE_2) THEN DIE_6 ELSE 0
OUTFLOWS:
STOCK_6_OUT = IF ALL_DICE = 21 THEN STOCK_6 ELSE 0

WORM_1(t) = WORM_1(t − dt) + (NEW_WORM_1 − WORM_1_OUT) ∗ dt
INIT WORM_1 = 1
INFLOWS:
NEW_WORM_1 = IF (LIFE_1>0 AND COIN_1<.5) THEN LIFE_1−1
ELSE IF (LIFE_1>0 AND COIN_1>.5) THEN LIFE_1+1
ELSE 0
OUTFLOWS:
WORM_1_OUT = IF ALL_DICE = 21 THEN WORM_1 ELSE 0

WORM_2(t) = WORM_2(t − dt) + (NEW_WORM_2 − WORM_2_OUT) ∗ dt
INIT WORM_2 = 2

INFLOWS:
NEW_WORM_2 = IF (LIFE_1>0 AND COIN_1<.5) THEN LIFE_1−2
ELSE IF (LIFE_1>0 AND COIN_1>.5) THEN LIFE_1+2
ELSE 0
OUTFLOWS:
WORM_2_OUT = IF ALL_DICE = 21 THEN WORM_2 ELSE 0

WORM_3(t) = WORM_3(t − dt) + (NEW_WORM_3 − WORM_3_OUT) ∗ dt
INIT WORM_3 = 3
INFLOWS:
NEW_WORM_3 = IF (LIFE_2>0 AND COIN_2<.5) THEN LIFE_2−1
ELSE IF (LIFE_2>0 AND COIN_2>.5) THEN LIFE_2+1
ELSE 0
OUTFLOWS:
WORM_3_OUT = IF ALL_DICE = 21 THEN WORM_3 ELSE 0

WORM_4(t) = WORM_4(t − dt) + (NEW_WORM_4 − WORM_4_OUT) ∗ dt
INIT WORM_4 = 4
INFLOWS:
NEW_WORM_4 = IF (LIFE_2>0 AND COIN_2<.5) THEN LIFE_2−2
ELSE IF (LIFE_2>0 AND COIN_2>.5) THEN LIFE_2+2
ELSE 0
OUTFLOWS:
WORM_4_OUT = IF ALL_DICE = 21 THEN WORM_4 ELSE 0

WORM_5(t) = WORM_5(t − dt) + (NEW_WORM_5 − WORM_5_OUT) ∗ dt
INIT WORM_5 = 5
INFLOWS:
NEW_WORM_5 = IF (LIFE_3>0 AND COIN_3<.5) THEN LIFE_3−1
ELSE IF (LIFE_3>0 AND COIN_3>.5) THEN LIFE_3+1
ELSE 0
OUTFLOWS:
WORM_5_OUT = IF ALL_DICE = 21 THEN WORM_5 ELSE 0

WORM_6(t) = WORM_6(t − dt) + (NEW_WORM_6 − WORM_6_OUT) ∗ dt
INIT WORM_6 = 6
INFLOWS:
NEW_WORM_6 = IF (LIFE_3>0 AND COIN_3<.5) THEN LIFE_3−2
ELSE IF (LIFE_3>0 AND COIN_3>.5) THEN LIFE_3+2
ELSE 0
OUTFLOWS:
WORM_6_OUT = IF ALL_DICE = 21 THEN WORM_6 ELSE 0

```
ALL_DICE = STOCK_1 + STOCK_2 + STOCK_3 + STOCK_4 + STOCK_5 +
   STOCK_6
COIN_1 = RANDOM(0,1)
COIN_2 = RANDOM(0,1)
COIN_3 = RANDOM(0,1)
DIE_2 = INT(RANDOM(1,7))
DIE_3 = INT(RANDOM(1,7))
DIE_4 = INT(RANDOM(1,7))
DIE_5 = INT(RANDOM(1,7))
DIE_6 = INT(RANDOM(1,7))
```

Reference

1. Cohen J, Stewart I (1994) The collapse of chaos. Penguin Books, New York, pp 105–106

Chapter 14
Langur Infanticide and Long-Term Matriline Fitness

For my own part I would as soon be descended from that heroic little monkey, who braved his dreaded enemy in order to save the life of his keeper; or from that old baboon, who, descending from the mountains, carried away in triumph his young comrade from a crowd of astonished dogs—as from a savage who delights to torture his enemies, offers up bloody sacrifices, practices infanticide without remorse, treats his wives like slaves, knows no decency, and is haunted by the grossest superstitions.

(Charles Darwin, The Descent of Man)

14.1 Langur Infanticide Model

The previous two chapters concentrated on natural selection, mutation, and fitness for a given species. The focus there was at the level of genes. In this chapter, we explore at the population level factors that may affect long-term fitness. Specifically, we explore the case of infanticides, the selective pressures they exert on the population, and the impacts they have on population size.

The particular case we explore here is for Hanuman Langurs, which are found throughout the Indian Subcontinent, Sri Lanka, and north into the pine forests of the Himalayas. Their habitats vary from moderate ranges such as savannahs and woodlands, to the extreme climates of tropical rainforests and dry desert fringes. Hanuman Langurs coexist with humans and are considered sacred temple animals in India where they are often provisioned [1].

The Hanuman Langurs possess a highly complicated social system, with three types of groups: single male breeding troops, bachelor bands, and multi-male/multi-female troops. Langur troops are flexible, with troop sizes ranging from 7 to 93 individuals at a

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_14,

time, averaging between 20 and 40 individuals. Langur dominance hierarchies are linear and tend to be organized into matrilines (due to females being philopatric, or remaining in their natal group), and stabilized through coalitions between high ranking females and their subordinates. Within single male breeding troops one male dominates and mates with all the females within his group. In order to maintain this privilege, the male must defend his position from outside males. These males come from bachelor bands that are composed of immature males and adults.

Juvenile males are often forced out of a group by dominant males who are not their fathers. After being forced from the group, these individuals may spend several years in all-male bands, forming alliances that can be utilized in the takeover of harem groups. In contrast to the predominately male and female troops, multi-male/multi-female bands have a polygamist group dynamic. Group membership among multi-male/multi-female groups is very flexible within these coed bands, with males constantly joining and leaving the group [1]. Due to the instability of these multi-male/multi-female groups and the complexity of social relationships between them, we model here a single male harem situation.

For our model, longitudinal data was available for a group of provisioned harem groups in Jodphur, India [2]. Because this group is provisioned and also raids crops, we did not include any density component in the model.

The Langurs of Jodphur, on which the model is based, show no seasonality in births, although there is a peak in March and a low in November [2]. Their ability to produce infants year-round is primarily due to the lack of seasonality in access to food. In contrast, Langurs found in the Himalayas, for example, show a seasonally varying birth pattern.

After a dominant male has been replaced in the harem group by a new male, the incoming male will kill the offspring of the ousted male. This triggers the end of lactational amennorhea and the female enters estrus sooner than she would have otherwise. This is the major tenant of the "sexual selection hypothesis" that favors infanticide as a male reproductive strategy [3]. This hypothesis assumes that males are deliberately killing unrelated infants in order to shorten the interbirth interval. Over 80 % of infant deaths occur within the first 9 months.

For infanticide to be an effective male reproductive strategy a male would have to be able to distinguish his offspring from that of another male. Proof of this can be seen in the female strategy of paternity confusion were a male will not tend to attack an infant whose mother he mated with. Furthermore, an infanticidal male would have to remain around the female long enough to prevent other males from attacking his infant. Hausfater [4] has shown that a 26.5 month stay is optimal for this task. We therefore assume that the average tenure of a male to be dominant is 26.5 months.

For the average interbirth interval we assume 16.7 months in the case of no infanticide, and 12.8 months after the death of an infant. Gestation is 6.5 months, and the age at which a juvenile female is considered an adult and gives birth for her fist time is 43 months.

Each month 25 % of the females become pregnant. Since the cycle is around 1 month, any given week one quarter of the females are able to become pregnant. Pregnancy is known from the literature to be 6.5 months, and the conveyor

represents this. Lactation length can vary, but the average interbirth interval is 16.7 months (without infanticide), thus minus the 6.5 months of gestation, weaning would be 10.2 months. After the death of an infant, the interbirth interval is shortened to 12.8 months. Subtracting from the shortened interbirth interval inus the gestation period of 6.5 months defines 6.3 months as the period after which females become fertile again.

The model of Fig. 14.1 represents a single matriline of related females at various ages in the life cycle as they are born, reproduce, and die. It is essential that we are able to distinguish individuals within the cohorts of infants, juveniles, pregnant, and lactating females by the time at which they enter a stock and when they leave it—rather than lumping all of them together with their respective peers. We can therefore not use STELLA's default stock—the reservoir—which, like a water reservoir, receives input and allows for output, without distinguishing what came in first or last. Instead, we specify the stocks infants, juveniles, pregnant, and lactating females as conveyors, which work analogous to a conveyor belt that has slats on which entities are placed, and after some transit time on the conveyor these entities are released in the order in which they entered the conveyor. Simply click and hold the stock tool on the toolbar (Fig. 14.2). Going down the list of options in Fig. 14.2 you find

- The reservoir, which lumps all the entries together,
- The conveyor, which works much like a conveyor belt, taking on elements and releasing them after a set time,
- The queue, which keeps elements queued up until some specified conditions are met, and
- The oven, which has a given capacity, keeps elements for a defined duration but does not distinguish its contents by their order of arrival and simply releases them all after their "cook time."

Choose the stocks for infants, juveniles, lactating, and pregnant individuals as conveyors. Specify the transit time for infants as 9 (months) and that for juveniles as 34 (months) because the age at first birth is 43 months ($43 - 9 = 34$). Set the initial conditions of each of these conveyors at 5 (individuals). Specify the transit time for pregnant females as 6.5 (months) and that for lactating females as 10 (months). Assume we start with ten pregnant and ten lactating individuals.

Note that with the specification of the transit times, the outflows from each stock are pre-defined: The individuals, who come into the stock first also leave first, after their specified duration in the stock. Any additional flow out of the conveyors that you draw will be interpreted as a leak—such as the "leakage" of infants from natural causes and infanticide. For the leakage, you can specify when it should occur along the conveyor, such as at its very beginning or anywhere during the transit time. For infant deaths, for example, we allow the loss of babies to occur as early as at the beginning of their cohort, and as late as at the very end of the 9 months in their cohort (Fig. 14.3).

We must also select the type of leakage these flow represents—linear or exponential. If you select the linear option, the total amount to leak is equal to the inflow∗DT∗leakage_fraction and this amount is distributed evenly across the leakage zone of the conveyer (each slat leaks this amount divided by the number of slats). In contrast, if you select the exponential option, the amount that leaks from

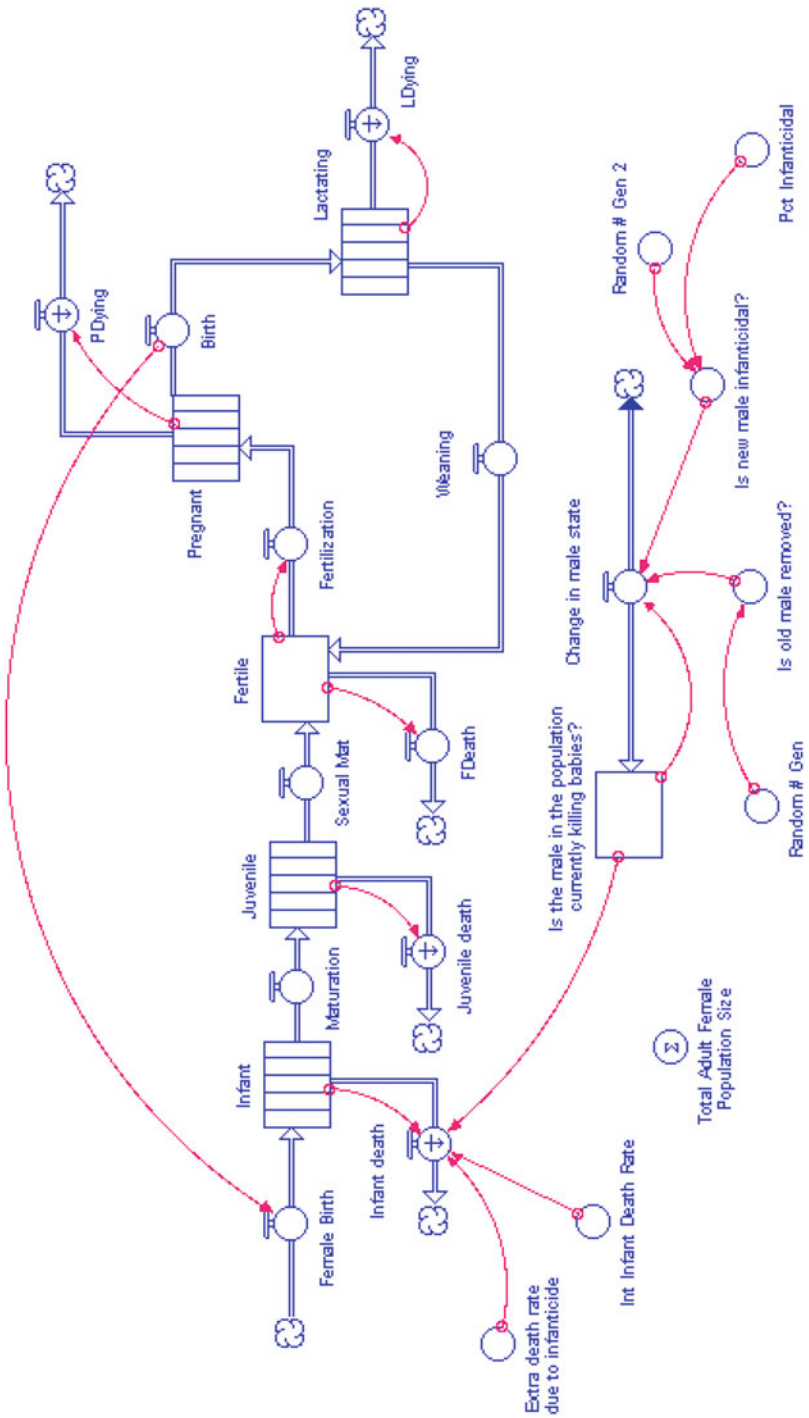
LDying
Lactating
PDying
Birth
Pregnant
Fertilization
Weaning
Fertile
FDeath
Sexual Mat
Juvenile
Juvenile death
Maturation
Infant
Infant death
Female Birth
Extra death rate due to infanticide
Int Infant Death Rate
Total Adult Female Population Size
Is the male in the population currently killing babies?
Change in male state
Random # Gen
Is old male removed?
Is new male infanticidal?
Random # Gen 2
Pct Infanticidal

Fig. 14.1

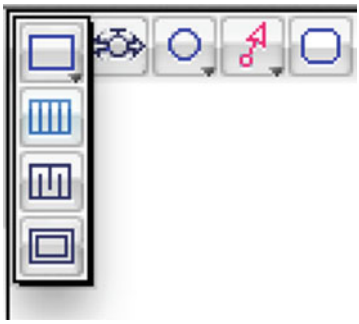

Fig. 14.2

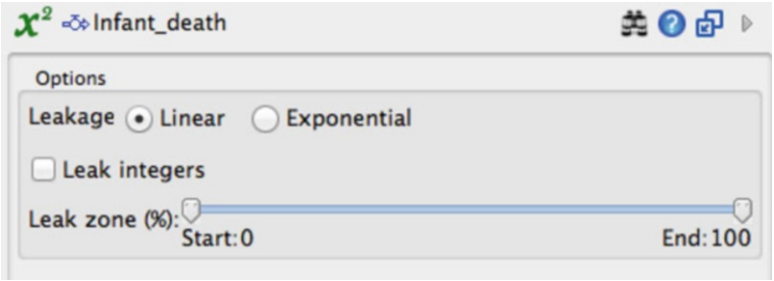

Fig. 14.3

each slat within the leakage zone is equal to the slat_contents∗leakage_fraction, generating an exponential decay in the amount leaked across the leakage zone. As a consequence, for linear leakage, the total amount leaked does not change with the length of the leakage zone, whereas for exponential leakage it does. In our model, we selected the linear option.

For infants, we specify the leak fraction as follows:

$$\begin{aligned}&\text{Leak Fraction} = \text{IFIs\_the\_male\_in\_the\_population\_currently\_killing\_babies?} > 0\\&\text{THEN}(\text{Extra\_death\_rate\_due\_to\_infanticide} + \text{Int\_Infant\_Death\_Rate}) * \text{Infant}\\&\text{ELSE Int\_Infant\_Death\_Rate} * \text{Infant}\end{aligned} \tag{14.1}$$

The matriline modeled here interacts with an outside population of males. A dominant male of a harem group encounters rivals with an unknown frequency and he will win some challenges but eventually loses his position to another. We capture this interaction in the model by creating a random number generator to yield values between 0 and 1. A converter called "Is_the_old_male_removed?" samples from this random distribution. It is specified as

$$\text{Is\_the\_old\_male\_removed} = \text{IF Random\_\#\_Gen} > .96 \ \text{ THEN } \ 1 \ \text{ ELSE } \ 0 \tag{14.2}$$

because we know that the average tenure of males in a group is 26.5 months, varying from just a few days to 74 months. We can assume that each month there is a 4 % chance of being replaced, so that after the average tenure time passes most males will have been replaced. (The DT must be set at one for this to work, if DT is set at 0.5, the 0.96 will have to be replaced by 0.98 so that the same chance is encountered).

A second unknown is whether the new male (should he win his dominance challenge) is himself infanticidal. To determine the likelihood that a male commits infanticide, we specify another random number generator in much the same way, only this time creating a component called "Pct Infanticidal" in which we can actually vary the percentage of males in the population who will exhibit this trait. A converter called "Is_new_male_infanticidal?" uses the random number generator and the percentage specified to determine the answer to this question. Its value is

$$\text{Is\_new\_male\_infanticidal} = \text{IF Random\_\#\_Gen\_2} <= \text{Pct\_Infanticidal THEN } 1 \text{ ELSE } 0'', \tag{14.3}$$

such that if 40 % are infanticidal (Pct Infanticidal = 0.4) there is a 40 % chance that the random number generated will be smaller than the 0.4 specified and thus a 40 % chance that the new dominant male is infanticidal.

Both of the conditional statements Eqs. (14.2) and (14.3) feed into a biflow connected to a stock called "Is_the_male_in_the_population_currently_killing_babies?", which we used above [Eq. (14.1)] to define the leakage fraction from the stock of infants. We set the initial condition for this stock to 0. The biflow is called "Change in Male Status" and has the following value

$$\begin{aligned}&\text{Change in Male Status} = \text{IF Is\_new\_male\_infanticidal?} + \text{Is\_old\_male\_removed?} = 2\\ &\text{THEN } 6 - \text{Is\_the\_male\_in\_the\_population\_currently\_killing\_babies?}\\ &\text{ELSE if Is\_new\_male\_infanticidal?} - \text{Is\_old\_male\_removed?} = -1\\ &\text{THEN} - \text{Is\_the\_male\_in\_the\_population\_currently\_killing\_babies?}\\ &\text{ELSE} - 1.\end{aligned} \tag{14.4}$$

This biflow allows us to make males in the population not perpetual baby killers for the entire time of their tenure. Instead, after the gestation period of around 6 months passes, new-borns are likely the male's offspring, and so this biflow causes the stock to "count down" until the male gains no benefit from killing babies: If a new male is successful and is infanticidal, then a six is read in the stock. For each month that the resident male remains in charge, this value decreases by one until it goes to zero—the stock is non-negative—and the male is no longer killing babies. If a newly dominant male is not infanticidal the sum of the two conditional statements is negative. The stock "Is_the_male_in_the_population_currently_killing_babies?" influences the infant death rate and the weaning tenure of the females.

Fig. 14.4

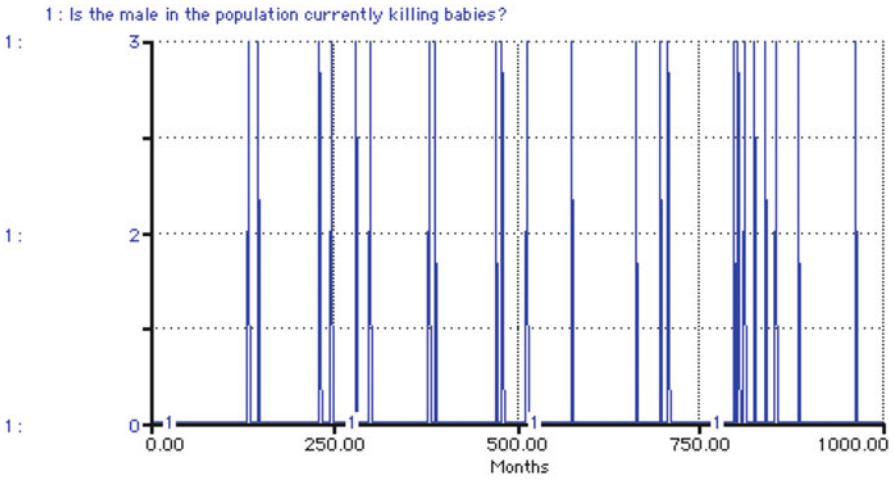

Fig. 14.5

Since the model deals only with total female population size, the total birthrate is divided by 2. Death of infants from natural causes is a function of Int_Infant_Death_Rate, which we assume to be 4 %, and the actions of an infanticidal male. The latter is estimated from the literature to be 0.366 [2].

To calculate the total adult female population size in our model, we could have created a converter and connected into it with action connectors the stocks of fertile, pregnant, and lactating individuals. Instead, we made use here of STELLA's Summing Converter (shown in Fig. 14.4 and used in the lower left-hand side of Fig. 14.1). You can select the Summing Converter by clicking and holding the Converter tool on the toolbar and place it anywhere in your model. Upon opening it, you simply need to specify which model components should be summed up, and STELLA does the rest.

Figure 14.5 shows how often males are replaced and that the modeled tenure lengths fit well with the known average and range. The total female population size is shown for four runs in Fig. 14.6—first without infanticide and for subsequent runs with (randomly) occurring infanticides, assuming 50 % of males to be infanticidal. Figure 14.7 compares total female population sizes for three different percentages of infanticidal males—0, 50, and 100 %.

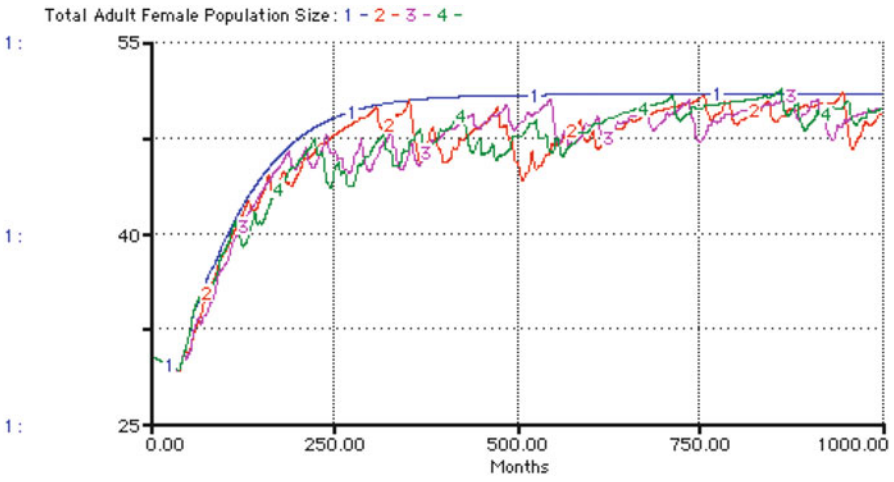

Fig. 14.6

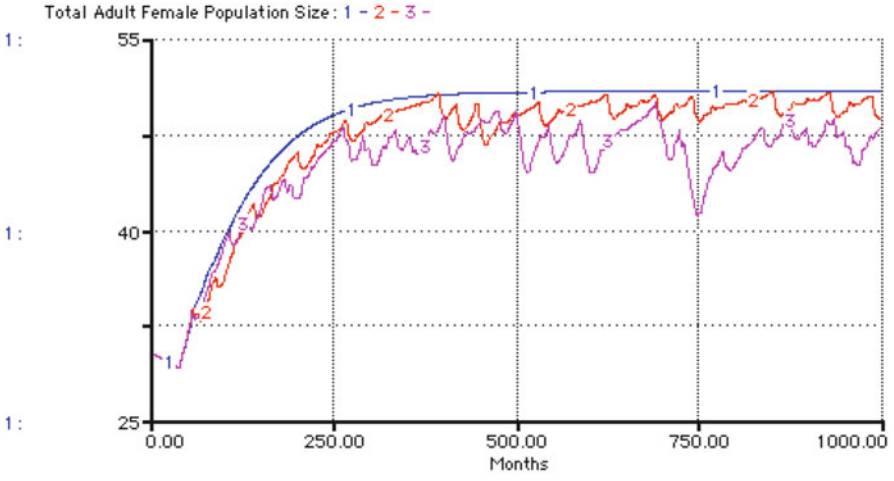

Fig. 14.7

The results show a population of females that increases and then tends to level off. Infanticide causes this population size to fluctuate at a level below that of a population with no infanticide, but the difference between 0 and 100 % infanticidal males, although significant, is only a few individuals. Over time the success of the matriline is barely affected by the presence of infanticidal males. The model thus seems to support the sexual selection hypothesis, and shows the varying levels of impact of infanticide on different hierarchical levels with the matriline showing very little long-term effect. Individual males benefit by speeding up the interbirth interval. Individual females suffering from a loss of infants see their lifetime fitness seriously reduced.

Even a small mating advantage for infanticidal males could lead to the maintenance of this trait since the cumulative effect on the matriline is so small. Variation in male success with the infanticidal/noninfanticidal strategies leads to the preservation of the polymorphism.

14.2 Langur Infanticide Model Equations

Fertile(t) = Fertile(t − dt) + (Weaning + Sexual_Mat − Fertilization − FDeath) ∗ dt
INIT Fertile = 10

INFLOWS:
Weaning = CONVEYOR OUTFLOW
Sexual_Mat = CONVEYOR OUTFLOW
OUTFLOWS:
Fertilization = .25∗Fertile
FDeath = .004∗Fertile
Is_the_male_in_the_population_currently_killing_babies?(t) = Is_the_male_in_the_population_currently_killing_babies?(t − dt) + (Change_in_male_state) ∗ dt
INIT Is_the_male_in_the_population_currently_killing_babies? = 0

INFLOWS:
Change_in_male_state = IF Is_new_male_infanticidal? + Is_old_male_removed? = 2 THEN 6−Is_the_male_in_the_population_currently_killing_babies? else if Is_new_male_infanticidal? − Is_old_male_removed? = −1 then −Is_the_male_in_the_population_currently_killing_babies? else −1
Infant(t) = Infant(t − dt) + (Female_Birth − Maturation − Infant_death) ∗ dt
INIT Infant = 5
TRANSIT TIME = 9
CAPACITY = INF
INFLOW LIMIT = INF

INFLOWS:
Female_Birth = .5∗Birth
OUTFLOWS:
Maturation = CONVEYOR OUTFLOW
Infant_death = LEAKAGE OUTFLOW
LEAKAGE FRACTION = If Is_the_male_in_the_population_currently_killing_babies? > 0 then (Extra_death_rate_due_to_infanticide +Int_Infant_ Death_Rate)∗Infant else Int_Infant_Death_Rate∗Infant
LEAK ZONE = 0% to 100%
Juvenile(t) = Juvenile(t - dt) + (Maturation - Sexual_Mat - Juvenile_death) ∗ dt
INIT Juvenile = 5

TRANSIT TIME = 34
CAPACITY =
INFLOW LIMIT =

INFLOWS:
Maturation = CONVEYOR OUTFLOW
OUTFLOWS:
Sexual_Mat = CONVEYOR OUTFLOW
Juvenile_death = LEAKAGE OUTFLOW
LEAKAGE FRACTION = .02*Juvenile
LEAK ZONE = 0% to 100%
Lactating(t) = Lactating(t − dt) + (Birth − Weaning − LDying) * dt
INIT Lactating = 10
TRANSIT TIME = IF Is_the_male_in_the_population_currently_killing_babies? > 0 THEN 6.3 ELSE 10.2
CAPACITY =
INFLOW LIMIT =

INFLOWS:
Birth = CONVEYOR OUTFLOW
OUTFLOWS:
Weaning = CONVEYOR OUTFLOW
LDying = LEAKAGE OUTFLOW
LEAKAGE FRACTION = .004*Lactating
LEAK ZONE = 0% to 100%
Pregnant(t) = Pregnant(t − dt) + (Fertilization − Birth − PDying) * dt
INIT Pregnant = 10
TRANSIT TIME = 6.5
CAPACITY = INF
INFLOW LIMIT = INF

INFLOWS:
Fertilization = .25*Fertile
OUTFLOWS:
Birth = CONVEYOR OUTFLOW
PDying = LEAKAGE OUTFLOW
LEAKAGE FRACTION = .004*Pregnant
LEAK ZONE = 0% to 100%
Extra_death_rate_due_to_infanticide = 0.366
Int_Infant_Death_Rate = 0.04
Is_new_male_infanticidal? = If Random_#_Gen_2 <= Pct_Infanticidal then 1 else 0
Is_old_male_removed? = IF Random_#_Gen > .98 then 1 else 0
Pct_Infanticidal = 0.5
Random_#_Gen = random(0,1)
Random_#_Gen_2 = Random(0,1)
Total_Adult_Female_Population_Size = Pregnant+ + Lactating+ + Fertile

References

1. Borries C (2001) Socioecology of Hanuman langurs: the story of their success. Evol Anthropol 10(4):122–137
2. Sommer V (1994) Infanticide among the langurs of Jodhpur. In: Parmigiani S, Vom Saal FS (eds) Infanticide and parental care, vol 13. Harwood Academic Publishers, Camberwell, pp 155–198
3. Hrdy SB (1974) Male-male competition and infanticide among the langurs (*Presbytis entellus*) of Abu, Rajasthan. Folia Primatol 22(1):19–58
4. Hausfater G (1984) Infanticide in langurs: strategies, counterstrategies, and parameter values. In: Hausfater G, Hrdy S (eds) Infanticide: comparative and evolutionary perspectives. Aldine, New York

Part IV
Models of Organisms

Chapter 15
Odor Sensing

The only defect of our senses is, that they give us disproportion'd images of things.

(Hume, 1739).

15.1 Basic Model of Odor Sensing

In this chapter we develop a model of a "nose" with three receptors. Each of the three receptors is tuned to a particular type of odor, either 1, 2 or 3. The goal of the "nose" is for the dominant receptor to suppress the other two odors in order to send a clear signal to the brain regarding the type of odor being sensed. The odor changes over time according to the following rule

$$\text{ODOR} = \text{IF TIME} \leq 15 \ \text{ THEN NORMAL}(1, .4) \ \text{ ELSE NORMAL}(3, .4) \quad (15.1)$$

Our "nose" can select the mean value out of a normally distributed odor. The mean value of the odor is thought to be an integer and it is assumed that integers above 3 would merely require additional receptors. In order for our nose to work, each receptor has to have a way of forgetting its present signal, should that signal decrease. We assume that the receptor is constantly forgetting or "zeroing out" the effect of the signal. Thus, when the strength of a signal to a particular receptor declines, the strength of the signal transmitted by that receptor declines. The three receptors are set up accordingly as

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_15,

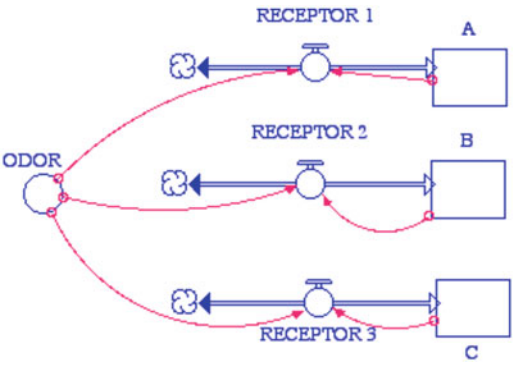

Fig. 15.1

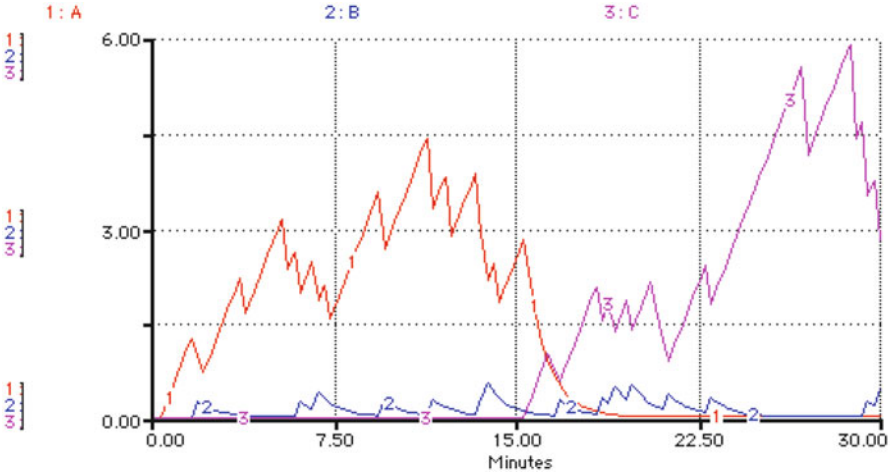

Fig. 15.2

$$\text{RECEPTOR } 1 = \text{IF } (\text{ODOR} \geq .5) \text{ AND } (\text{ODOR} \leq 1.5) \text{ THEN } 1 \text{ ELSE} - \text{A} \tag{15.2}$$

$$\text{RECEPTOR } 2 = \text{IF } (\text{ODOR} > 1.5) \text{ AND } (\text{ODOR} \leq 2.5) \text{ THEN } 1 \text{ ELSE} - \text{B} \tag{15.3}$$

$$\text{RECEPTOR } 3 = \text{IF } (\text{ODOR} > 2.5) \text{ AND } (\text{ODOR} \leq 3.5) \text{ THEN } 1 \text{ ELSE} - \text{C} \tag{15.4}$$

The complete model is shown in Fig. 15.1. Our resulting signal to the brain from the three receptors is unambiguous in spite of the fact that the incoming odor signal was mixed and overlapped the boundaries of the receptors (Fig. 15.2).

The model does not depict a perfect nose. There are too few receptors. Add more receptors and increase the range of the possible odors. Change the sensitivity of some receptor such that if there is too much odor of a particular strength or for a particular duration, that receptor gets "numb" and provides the wrong signals to the brain. How can the brain minimize those errors? Can you reprogram this model such that any continuous odor is eventually ignored, until a sudden change in the level of the odor is introduced?

15.2 Odor Sensing Model Equations

A(t) = A(t − dt) + (RECEPTOR_1) ∗ dt
INIT A = 0 {Signals}

INFLOWS:
RECEPTOR_1 = IF (ODOR ≥ .5) AND (ODOR ≤ 1.5) THEN 1 ELSE − A {Signals per Minute}
B(t) = B(t − dt) + (RECEPTOR_2) ∗ dt
INIT B = 0 {Signals}

INFLOWS:
RECEPTOR_2 = IF (ODOR > 1.5) AND (ODOR ≤ 2.5) THEN 1 ELSE −B {Signals per Minute}
C(t) = C(t − dt) + (RECEPTOR_3) ∗ dt
INIT C = 0 {Signals}

INFLOWS:
RECEPTOR_3 = IF (ODOR > 2.5) AND (ODOR ≤ 3.5) THEN 1 ELSE −C {Signals per Minute}
ODOR = IF TIME≤15 THEN NORMAL(1,.4) ELSE NORMAL(3,.4) {Odor Units}

Chapter 16
Stochastic Resonance

The thirst for something other than what we have. . .to bring something new, even if it is worse, some emotion, some sorrow; when our sensibility, which happiness has silenced like an idle harp, wants to resonate under some hand, even a rough one, and even if it might be broken by it.

(Marcel Proust, Swann's Way)

16.1 Basic Stochastic Resonance Model

In the previous chapter we modeled a "nose" that detects some odor. Let us model in this chapter an "ear" with a limited range of amplitudes of acoustic signals that it can detect. The threshold of audible amplitudes is arbitrarily set at 0.03. Let us specify a harmonic signal whose amplitude (0.01) is too low to be heard.

$$\text{HARMONIC} = \text{SINWAVE}(.01, .1) + .01 \tag{16.1}$$

Additionally, there is noise in the system.

$$\text{NOISE} = \text{RANDOM}(0, .02) \tag{16.2}$$

With the addition of noise the peak apparent amplitude to the ear is doubled, goes above the audio threshold and thus can be "heard":

$$\text{COMBINED SIGNAL} = \text{HARMONIC} + \text{NOISE} \tag{16.3}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_16,

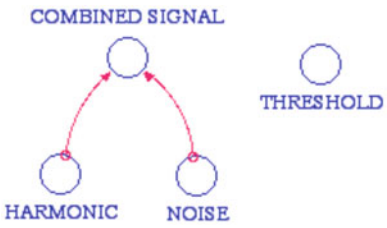

Fig. 16.1

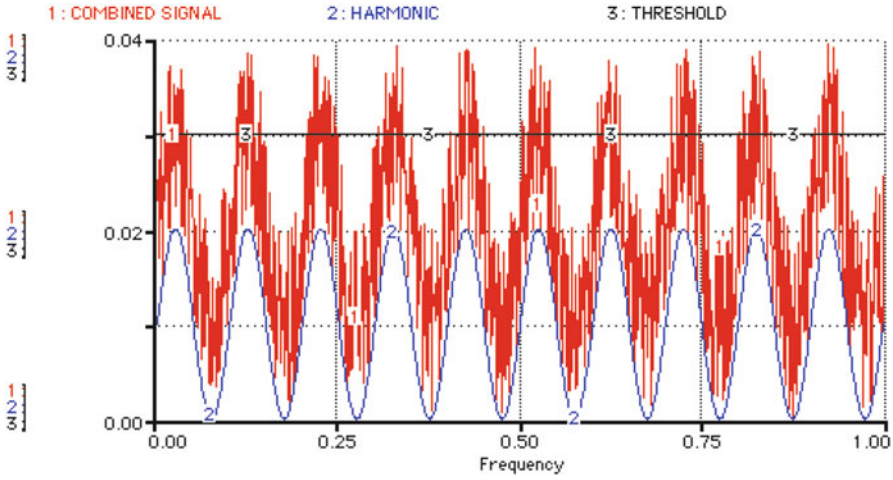

Fig. 16.2

The "frequency" of the noise is the DT. In every DT the random level of the noise is changed. If the noise is not generated often enough, the signals become impossibly masked by the noise.

Our model of stochastic resonance is shown in Fig. 16.1. Try a DT of 0.001 and of 0.0625. The results are shown, respectively, in Figs. 16.2 and 16.3. Recognize how little of the harmonic signal can be detected if the frequency is too low.

The human ear–brain smoothes over the fact that the peaks above the threshold are composed of several tightly spaced signals of noise plus signal. The same phenomenon is present in the eye-brain system. Imagine a set of leafless branches fidgeting in the wind, seen against a bright sky. Assume we are looking for a mammal moving through the branches. The movement of the branches is the "white" visual noise in this system. Our eye-brain system is extremely well equipped to filter out the dark shapes formed at random by the intersection of several branches, because this sort of shape does not persist. We easily see the mammal flitting though the branches. Adding leaves to these branches just makes the problem harder, but perhaps not impossible. So the eye-brain and ear–brain

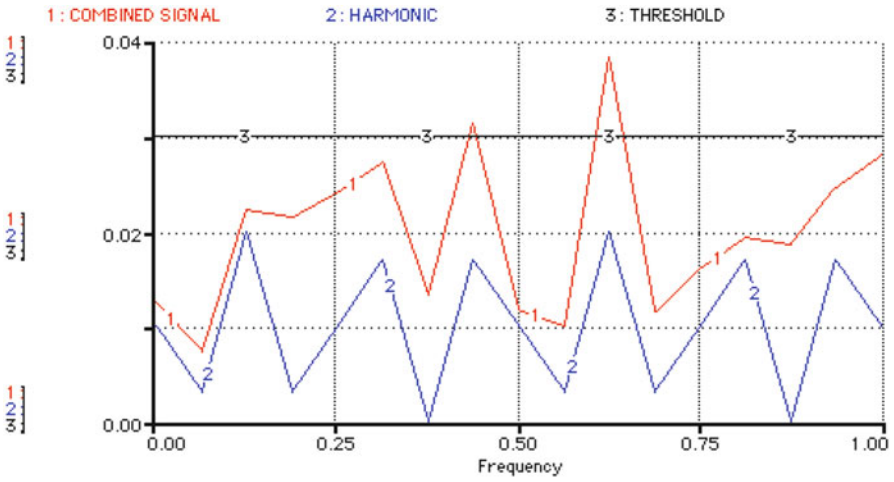

Fig. 16.3

systems must learn to allow persistence of the input signal in order to appropriately average it and make decisions based on that average value. In the case of the combined audio signal, this quality of persistence allows one to compose the parts of this combine that exceed the audio threshold into a particular frequency. In the visual example, the persistence quality allows us to ignore those signals that are random and focus our attention on the more apparently purposeful motion.

Now that we have modeled a simple "nose" and an "ear," let us turn to a more elaborate model of a four-chambered heart. This is the topic of the following chapter.

16.2 Stochastic Resonance Model Equations

COMBINED_SIGNAL = HARMONIC+NOISE
HARMONIC = SINWAVE(.01,.1)+.01
NOISE = RANDOM(0,.02)

Chapter 17
Heart Beat

I have to remind myself to breathe – almost to remind my heart to beat!

(Emily Brontë, Wuthering Heights)

17.1 Basic Heart Beat Model

In Chap. 4 we have mentioned the fact that organs, or entire organisms, may be viewed as having evolved to a critical state in which they are seemingly close to chaotic behavior. Yet, at that stage of their evolution, they may have optimized their behavior with regard to a specific task. The heart is such an organ, and we model it in a simplified form in this chapter.

The behavior of the four-chambered human heart follows a series of closely interrelated flow processes. Deoxygenated blood from the venous system is collected into the vena cava and then delivered into the right atrium. Blood then is pumped into the right ventricle past the bicuspid valve. Blood is pumped to the lungs via the pulmonary arteries where it becomes oxygenated. Blood then is delivered into the left atrium via the pulmonary vein. From there the blood is pumped into the left ventricle. The oxygenated blood is then pumped to the body's arterial system through the aorta. This pumping rate is controlled by the heart's pacemaker. Special cells in both atrial chambers have the ability to send electrical impulses that cause the atria to contract. The same impulse is also carried to the A-V node which causes ventricular contraction. The result is first an atrial contraction then, after a few millisecond delay, the ventricular contraction. A very basic rendition of the heart is laid out in Fig. 17.1.

Our model of the four-chambered heart (Fig. 17.2) is constructed to respond to changes in blood demand and to disease. A volume of blood is pumped through

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_17,

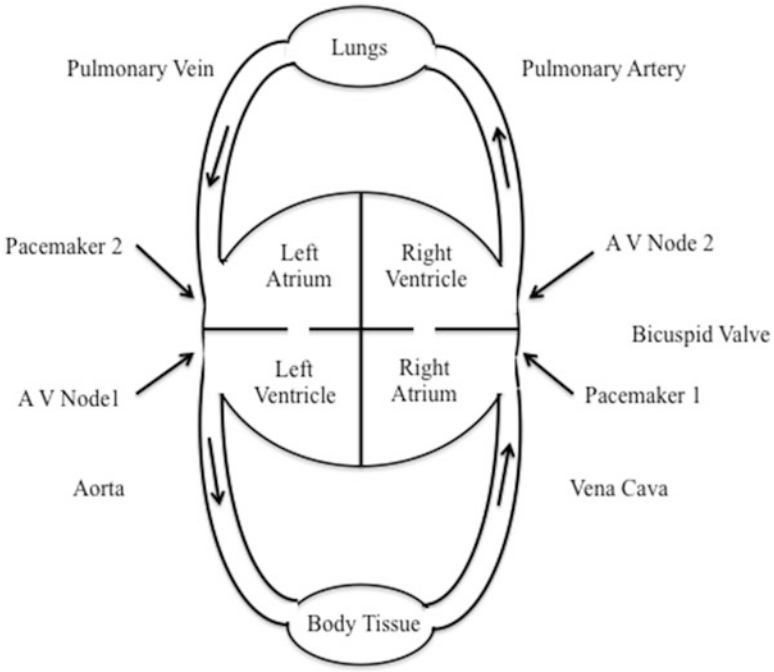

Fig. 17.1

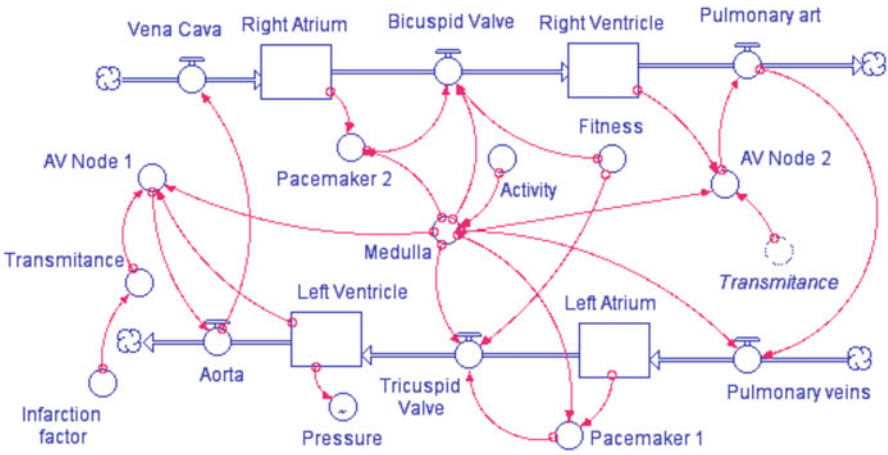

Fig. 17.2

each of the four chambers of the heart. At the core of the model are two pacemakers, each sending a Pulse that "pumps" the blood causing a flow from one chamber to the next:

$$\text{PACEMAKER} = \text{PULSE}(\text{RIGHT ATRIUM}, 1, \text{MEDULA}) \tag{17.1}$$

$$\text{PACEMAKER 2} = \text{PULSE}(\text{LEFT ATRIUM}, 1, \text{MEDULA}) \tag{17.2}$$

The Pulse's firing frequency is controlled by the medulla, part of the brain that stimulates some body functions such as breathing and heart beat. The activity of the medulla is a function of the activity of the heart, here arbitrarily set as

$$\text{MEDULA} = 8 * \text{ACTIVITY}, \tag{17.3}$$

with ACTIVITY $= 1$ for a person at rest and ACTIVITY < 1 for an active person. Increased activity increases the rate of pulses being sent by the pacemaker. FITNESS is also included in the model. The more fit an individual, the more efficient the heart is at pumping blood.

A Delay is used in both the Bicuspid and Tricuspid flows to create a short pause between atrial and ventricle firing.

$$\text{PULMONARY ART} = \text{DELAY}(\text{AV NODE\_2}, 1) \tag{17.4}$$

$$\text{AORTA} = \text{DELAY}(\text{AV NODE}, 1) \tag{17.5}$$

To make the graph of the blood flow more realistic, a Smooth function was used. The Built-in SMTH1(A,X) calculates the first order exponential smooth of a variable A, using an exponential averaging time of X. The Smooth function gives the appearance of blood gradually flowing into its chamber. To capture the gradual flow of blood in our model we define the flows connecting atrium and ventricle of the left and right chambers, BISCUSPID VALVE and TRICUSPID VALVE, with the smooth function:

$$\text{BISCUSPID VALVE} = \text{SMTH1}(\text{PACEMAKER} * \text{FITNESS}, \text{MEDULA} * .1) \tag{17.6}$$

$$\text{TRICUSPID VALVE} = \text{SMTH1}(\text{PACEMAKER 2} * \text{FITNESS}, \text{MEDULA} * .1) \tag{17.7}$$

To be able to read off the blood pressure in our model, we defined a converter PRESSURE as a graphical function of the amount of blood on the left ventricle as shown in Fig. 17.3.

Heart disease was added into the model with the infarction factor, I FACTOR. This value represents the quantity of heart tissue damage as a parameter. Increased damage influences the transmittance of the electrical impulse. Run the model for alternative I FACTOR values and observe the result.

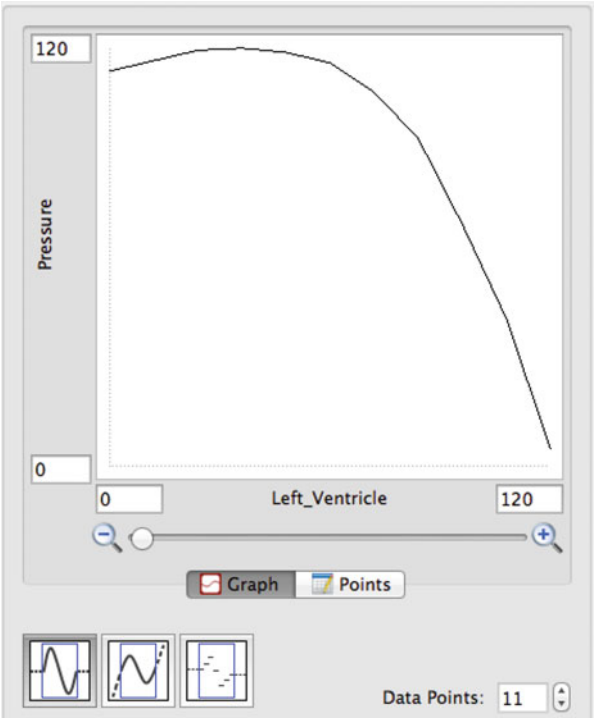

Fig. 17.3

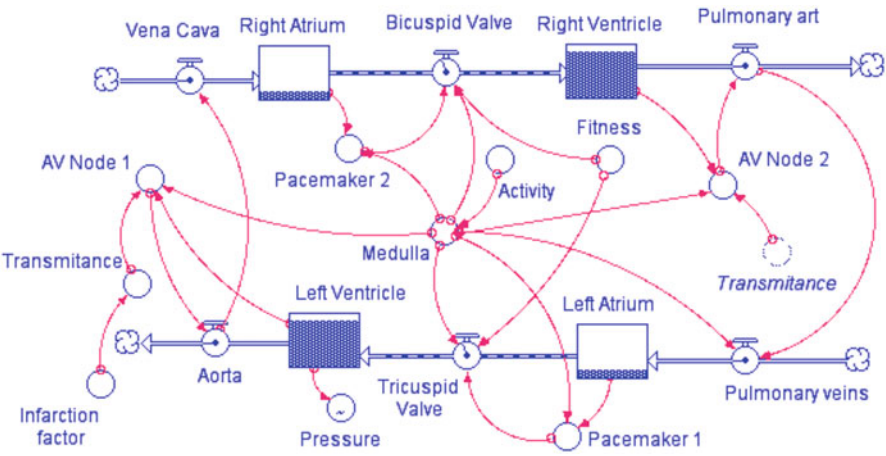

Fig. 17.4

We annotated stocks and flows in our STELLA diagram to visualize their change over time. As the model runs, the shading of these icons now varies (see Fig. 17.4). Choose "Model Prefs. . ." from the toolbar and click on the icons you wish to animate (e.g., as in Fig. 17.5).

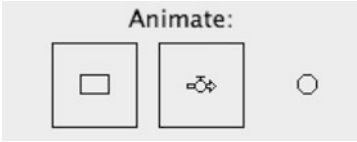

Fig. 17.5

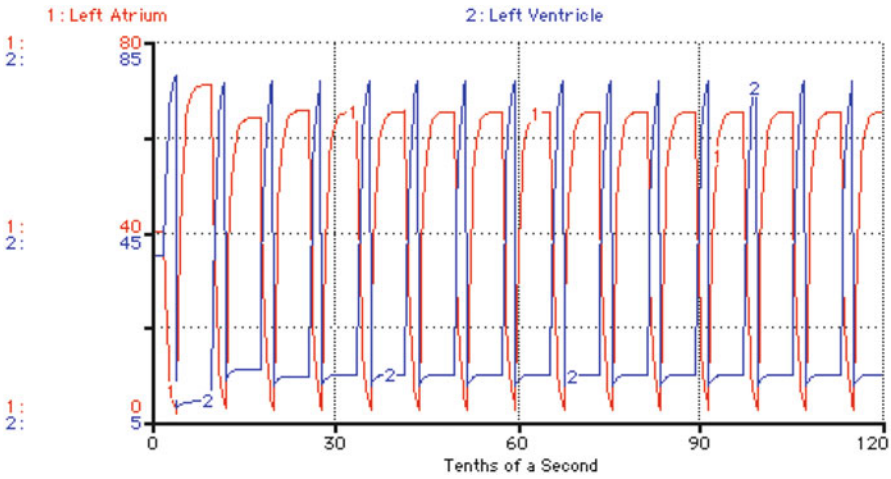

Fig. 17.6

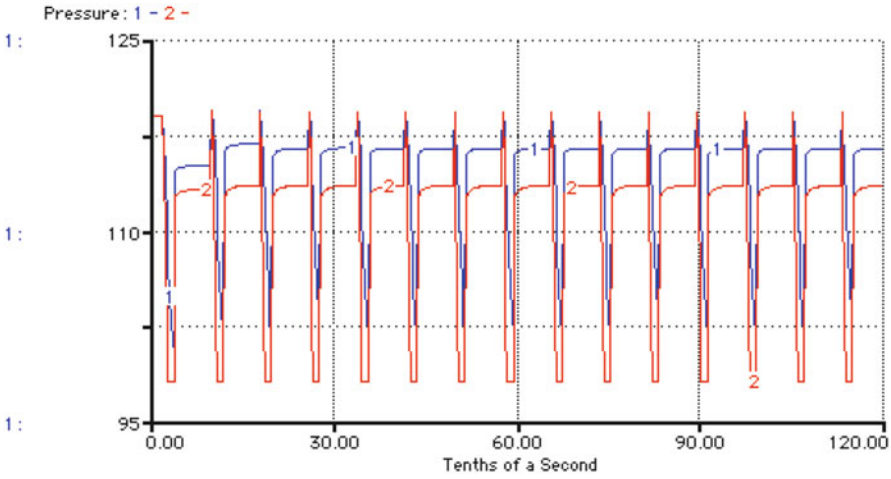

Fig. 17.7

Figure 17.6 shows the pressure of the left atrium and ventricle over time. Figure 17.7 shows the pressure in the left ventricle for two fitness rates—1 and 1.4. At all times, the pressure is lower under higher fitness rates than lower ones. Modify the model to allow for the gradual awakening of a sleeping person, and a

gradual increase in that person's stress level. How does a less-fit person's heart performance compare to that of a fitter person? What about a display of the effects of heart disease?

17.2 Heart Beat Model Equations

Left_Atrium(t) = Left_Atrium(t − dt) + (Pulmonary_veins − Tricuspid__Valve) ∗ dt
INIT Left_Atrium = 40
INFLOWS:
Pulmonary_veins = smth1(Pulmonary_art,Medulla∗.1)
OUTFLOWS:
Tricuspid__Valve = smth1(Pacemaker_1∗Fitness,Medulla∗.1)
Left_Ventricle(t) = Left_Ventricle(t − dt) + (Tricuspid__Valve − Aorta) ∗ dt
INIT Left_Ventricle = 40
INFLOWS:
Tricuspid__Valve = smth1(Pacemaker_1∗Fitness,Medulla∗.1)
OUTFLOWS:
Aorta = delay(AV_Node_1,1)
Right_Atrium(t) = Right_Atrium(t − dt) + (Vena_Cava − Bicuspid_Valve) ∗ dt
INIT Right_Atrium = 40
INFLOWS:
Vena_Cava = Aorta
OUTFLOWS:
Bicuspid_Valve = smth1(Pacemaker_2∗Fitness,Medulla∗.1) {Ventricle fills passively until halfway through the cardiac cycle.}
Right_Ventricle(t) = Right_Ventricle(t − dt) + (Bicuspid_Valve − Pulmonary_art) ∗ dt
INIT Right_Ventricle = 40
INFLOWS:
Bicuspid_Valve = smth1(Pacemaker_2∗Fitness,Medulla∗.1) {Ventricle fills passively until halfway through the cardiac cycle.}
OUTFLOWS:
Pulmonary_art = delay(AV_Node_2,1)
Activity = 1 {1 represents a person at rest, values less than 1 represent activity}
AV_Node_1 = pulse(Left_Ventricle,Transmitance,Medulla)
AV_Node_2 = pulse(Right_Ventricle,Transmitance,Medulla)
Fitness = 1.4
Infarction__factor = 1.1
Medulla = 8∗Activity
Pacemaker_1 = PULSE(Left_Atrium,1,Medulla)
Pacemaker_2 = pulse(Right_Atrium,1,Medulla)
Pressure = GRAPH(Left_Ventricle)
(0.00, 113), (12.0, 116), (24.0, 119), (36.0, 119), (48.0, 118), (60.0, 115), (72.0, 107), (84.0, 93.6), (96.0, 69.0), (108, 42.0), (120, 5.00)
Transmitance = 2∗Infarction__factor

Chapter 18
Bat Thermo-Regulation

The baby bat
Screamed out in fright,
'Turn on the dark,
I'm afraid of the light."

(Shel Silverstein)

18.1 Bat Thermo-Regulation Model

Objects of a particular temperature, surrounded by a cooler environment, tend to lose heat to their surrounding. Under the assumption that the environment is large in comparison to the object, the rate of change in the temperature of the object is determined by the difference between the object's temperature and the ambient temperature. The ambient temperature becomes the target final temperature of the object.

In this chapter we model the thermo-regulatory process of a bat. The bat loses heat based on Newton's law of cooling. This law states that the rate of change of a body's temperature is linearly proportional to the temperature difference between the object and the environment. In our case, heat loss by a bat is

$$\text{HEAT LOSS} = \text{K} * (\text{BODY TEMP} - \text{AMBIENT TEMPERATURE}) \quad (18.1)$$

Unlike the standard setting for Newtonian cooling of inanimate objects, bats are able to influence the cooling coefficient K by adjusting their fur, rolling into a more nearly spherical shape (minimum surface per unit volume), and crowding. Thus, the cooling coefficient is a function of temperature. The relationship between the

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_18,

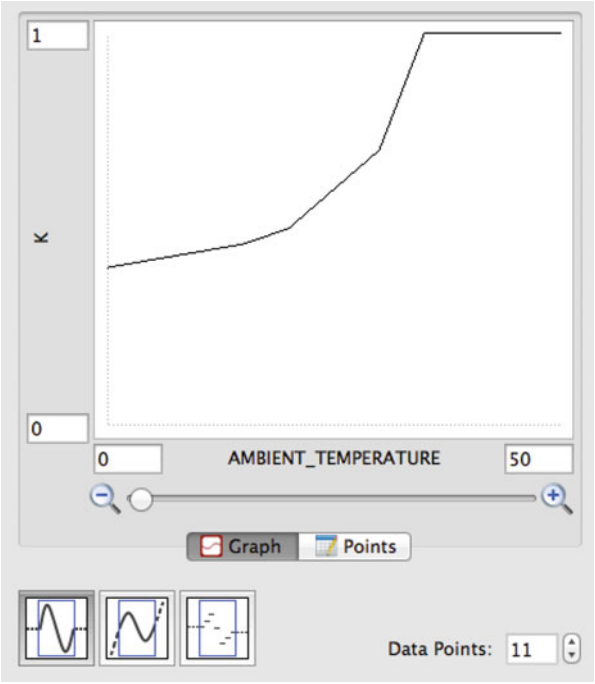

Fig. 18.1

cooling constant and ambient temperature is shown in Fig. 18.1 and the data for the model are from Spain [1].

Heat gain is a function of the difference between the body temperature and a set temperature, TN. The chemical reaction rate is halved for every 10 °C drop in body temperature. The metabolic rate maximum (10) sets the upper limit on basal metabolic rate, QR:

$$\begin{aligned} \text{QR} = \text{IF } 2 + 2 * (\text{TN} - \text{BODY TEMP}) \leq 10 \\ \text{THEN } \; 2 + 2 * (\text{TN} - \text{BODY TEMP}) \; \text{ ELSE } \; 10 \end{aligned} \tag{18.2}$$

The complete system is shown in Fig. 18.2. The parameters K and A translate heat flows into temperature changes for the bat.

Set the ambient and initial body temperature at different levels for consecutive model runs and watch the heat loss and heat gain converge while the body temperature approaches its final value. When this process is carried out for a variety of ambient temperatures the following graph results. In Figs. 18.3, 18.4, and 18.5 we plotted the temperatures for eight model runs with AMBIENT TEMPERATURE ranging from 10 to 24 °C. Note how the thermo-regulation process is proceeding normally (nearly horizontal portion of the curve) and then the curve drops suddenly. Why is this break point occurring? What process is producing temperature equilibrium in the bat at ambient temperatures below 15 °C?

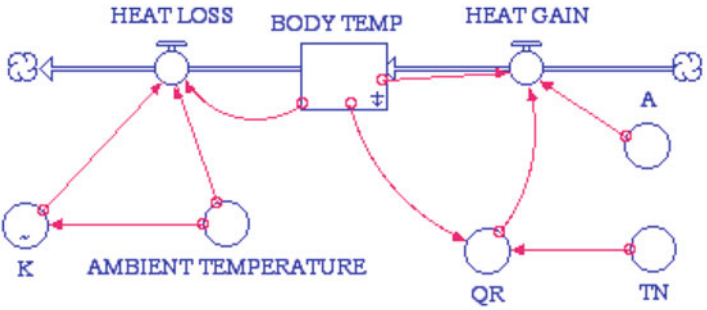

Fig. 18.2

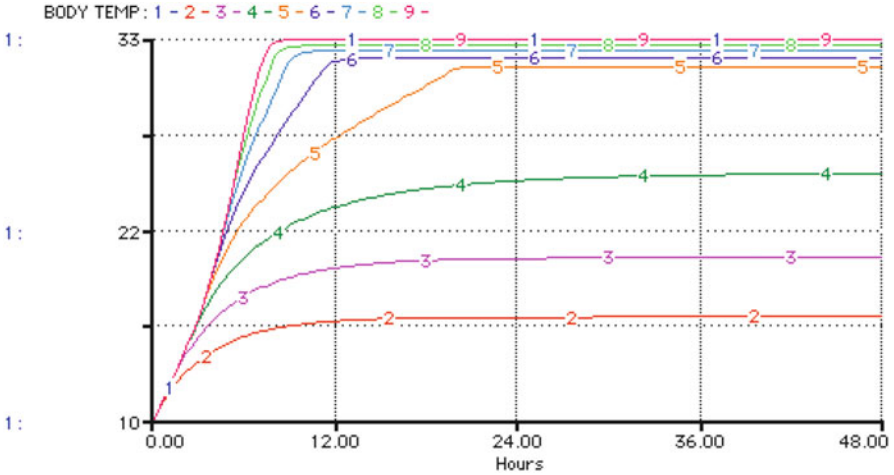

Fig. 18.3

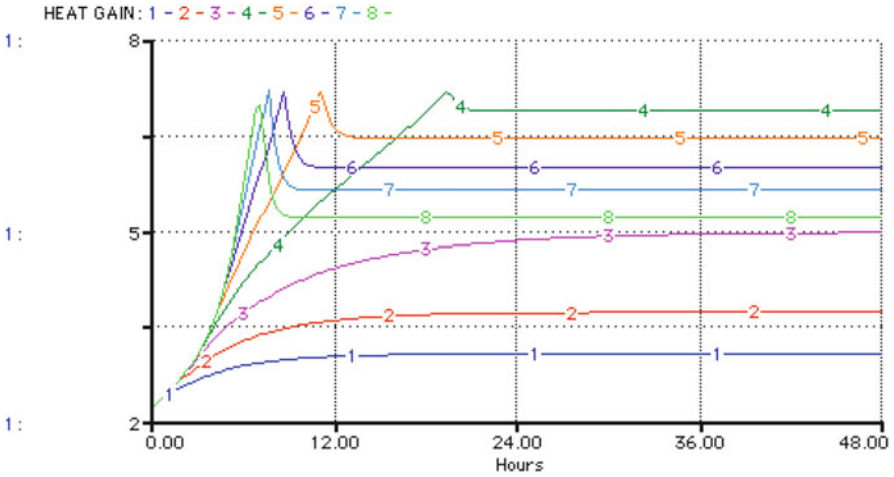

Fig. 18.4

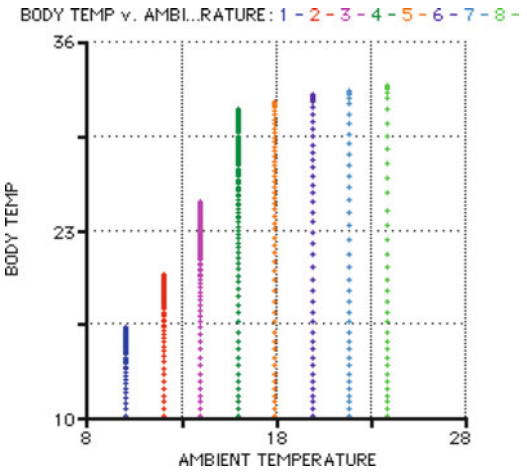

Fig. 18.5

18.2 Bat Thermoregulation Model Equations

BODY_TEMP(t) = BODY_TEMP(t − dt) + (HEAT_GAIN − HEAT_LOSS) ∗ dt
INIT BODY_TEMP = 10 {Deg C}

INFLOWS:
HEAT_GAIN = QR∗EXP(-A∗(35 − BODY_TEMP)) {Deg C per Hour}
OUTFLOWS:
HEAT_LOSS = K∗(BODY_TEMP-AMBIENT_TEMPERATURE) {Deg C per Hour}
A = .0693 {Deg C}
AMBIENT_TEMPERATURE = 14 {Deg C}
QR = IF 2 + 2∗(TN - BODY_TEMP) ≤ 10 THEN 2 +2∗(TN − BODY_TEMP) ELSE 10
TN = 35 {Deg C}
K = GRAPH(AMBIENT_TEMPERATURE)
(0.00, 0.4), (5.00, 0.42), (10.0, 0.44), (15.0, 0.46), (20.0, 0.5), (25.0, 0.6), (30.0, 0.7), (35.0, 1.00), (40.0, 1.00), (45.0, 1.00), (50.0, 1.00)

Reference

1. Spain JD (1992) BASIC microcomputer models in biology. Addison-Wesley, Reading

Chapter 19
The Optimum Plant

Plant, a Natural Body that has a vegetable Soul.

[1696 Phillips (ed. 5)]

19.1 Optimum Plant Model

Just describing or simulating the change in living organisms may simply not be good enough. We all want ultimately to predict what these organisms would do under prescribed circumstances. Scientists interested in predictions first need a good description of the behavior of the living organism. Toward that end, they frequently find it advantageous to set up optimality hypotheses of the organism's behavior and then compare the optimization results with results of experiments on the actual dynamics of the organism.

The work begun by Cohen in 1971 on the optimization of plants makes a good example of this kind of approach [1]. A good summary is provided in Roughgarden [2]. Cohen's model is the simplest model possible of optimal control in plants. The basic hypothesis is that this plant strives to produce the maximum reproductive biomass by the end of the growing season, a period that is T units long. We assume that the plant is genetically "wired" for this growing season, i.e., its genetics have been so shaped by the local environment that the plant acts as though it "knows" what the length of the growing season is. We further assume that the growing season lasts for 5 time units and still further that the growth of the vegetative part ΔX in the time DT is given by

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_19,

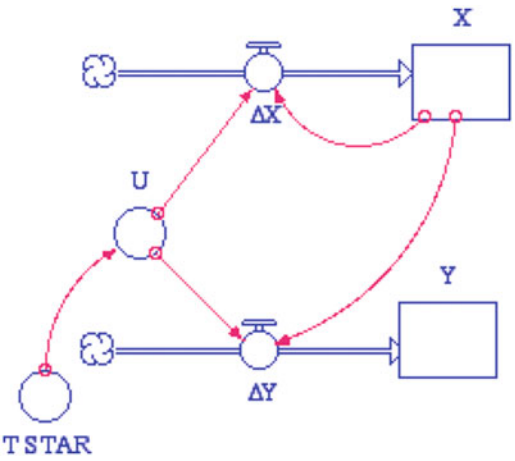

Fig. 19.1

$$\Delta X = \frac{dX}{dt} = U * X \tag{19.1}$$

and the growth of the reproductive part ΔY in the time DT is

$$\Delta X = \frac{dY}{dt} = (1 - U) * X \tag{19.2}$$

U is the control variable that we use to simulate a plant's shift of its resources from vegetative growth to reproductive growth. The control, U, is necessarily $0 \leq U \leq 1$. The optimality problem is one of maximizing Y(T).

For this problem, assume that either the vegetative *or* reproductive portion is growing but not both at the same time. In such a case we say that the control is "bang-bang"—it is either 1 or 0. Under this assumption, the control becomes:

$$\begin{aligned} &\text{if } t < T \text{ STAR then } U = 1 \\ &\text{if } t > T \text{ STAR then } U = 0, \end{aligned} \tag{19.3}$$

where T STAR is the shift time, the time when the plant's production shifts from vegetative to reproductive.

We can build a model (Fig. 19.1) of this process and change T STAR for successive runs until we find the maximum Y at $t = T = 5$. This value is 5.35 if we use a small enough DT, and we experimentally find that the optimal switch time, T STAR, is equal to 4.00 (Fig. 19.2)

This model is a good example of how the correct DT must be found. $DT = 1$ is too large. Choices of $DT = 0.01$ and smaller are appropriate because they give the same answer of $Y = 5.45$ when $T\ STAR = 4$.

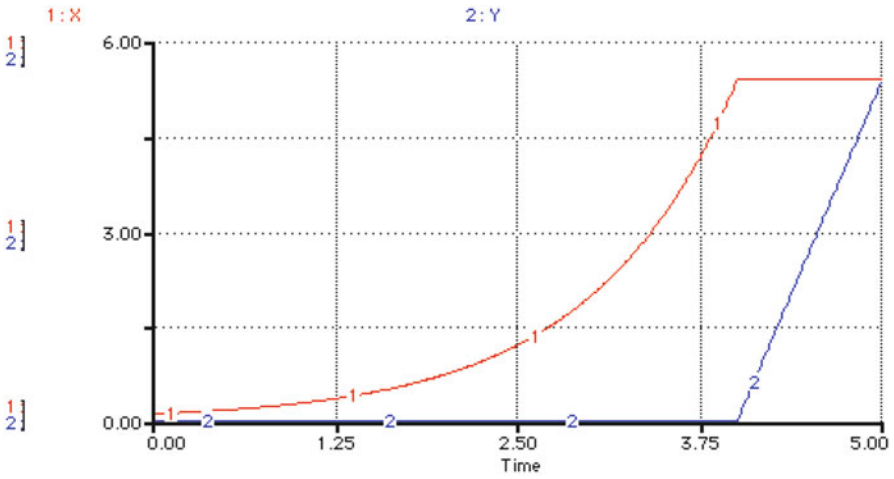

Fig. 19.2

Our result can be verified by theory using the Pontryagin optimization theory [3]. For the situation outlined above, the Hamiltonian, H, is:

$$H = A * U * X + B * (1 - U) * X \tag{19.4}$$

The variables A and B are called the costate or adjoint variables. These costate variables must conform to the following conditions according to the Pontryagin theory:

$$dA/dt = -\partial H/\partial X \tag{19.5}$$

$$dB/dt = -\partial H/\partial Y. \tag{19.6}$$

Thus,

$$dA/dt = -A * U - B * (1 - U) \tag{19.7}$$

and

$$dB/dt = 0. \tag{19.8}$$

Maximizing Y(T) means that the terminal conditions on the costate variables must be

$$A(T) = 0 \tag{19.9}$$

and

$$B(T) = 1. \tag{19.10}$$

From Eqs. (19.8) and (19.10), we have

$$B(t) = 1. \tag{19.11}$$

In the region $t^* \leq t \leq T$, $U = 0$, so with Eqs. (19.7) and (19.11) we have

$$A(t) = T - t. \tag{19.12}$$

A third and final Pontryagin condition is

$$\partial H/\partial U = 0 \tag{19.13}$$

at the optimal switch time only. So at $t = t^*$, $a = b$; $t^* = T - 1$, or in our case, $t^* = 4.0$, which is what we found experimentally on the computer.

Does the hypothesis of optimal plant behavior yield the right answer? There is only one way to find out. Compare it to experimental results [4]. Even if you successfully compare, the hypothesis may not be sufficiently general to cover the behavior of many different types of plants under different environmental conditions. Even if your model did predict correctly for several different kinds of plants, it is only a good suspect in the search for whether or not living organisms seem to follow any kind of optimal plan.

These optimal control problems in plants can be very difficult. Imagine that the growth equations are logistic rather than the simple ones given above. Further, imagine that the growth periods overlap and finally think of the perennial plant, which regrows from root extensions and from seeds. Then the determination of the actual optimal path of the control and of X and Y may be accomplished only by numerical analysis. The control may not be bang-bang but graded, allowing both types of biomass to grow simultaneously for some part of the growing season. The best procedure to follow in most cases is to first do as much analytical work as possible to simplify the ensuing numerical analysis. Usually, one of the costate variables can be found in terms of the two biomasses and perhaps the control. However, the actual solution frequently must be obtained numerically even with a significant quantity of numerical analysis.

19.2 Optimal Plant Model Equations

$X(t) = X(t - dt) + (\Delta X) * dt$
INIT X = .1 {Kg}

INFLOWS:
$\Delta X = U*X$ {Kg per Time Period}
$Y(t) = Y(t - dt) + (\Delta Y) * dt$
INIT $Y = 0$ {Kg}

INFLOWS:
$\Delta Y = (1 - U)*X$ {Kg per Time Period}
$T\_STAR = 4$
$U =$ IF TIME $\leq$ T_STAR THEN 1 ELSE 0 {1/Time Period}

References

1. Cohen D (1971) Maximizing final yield when growth is limited by time or by limited resources. J Theor Biol 33(2):299–307
2. Roughgarden J (1986) Models of population processes in plants, vol 18, Lectures of mathematics in the life science. American Mathematical Society, Providence, pp 235–267
3. Kamien MI, Schwartz N (1983) Dynamic optimization: the calculus of variations and optimal control in economics and management. North Holland, Dover Publications, pp 186–192
4. Hannon B (1993) The optimal growth of helianthus annuus. J Theor Biol 165(4):523–531

Chapter 20
Soybean Plant Growth

In life, a person can take one of two attitudes: to build or to plant. The builders might take years over their tasks, but one day, they finish what they're doing. Then they find that they're hemmed in by their own walls. Life loses its meaning when the building stops.
Then there are those who plant. They endure storms and all the vicissitudes of the seasons, and they rarely rest. But unlike a building, a garden never stops growing. And while it requires the gardener's constant attention, it also allows life for the gardener to be a great adventure.
Gardeners always recognize each other, because they know that in the history of each plant lies the growth of the whole World.

(Paulo Coelho)

20.1 Soybean Plant Model

While the previous chapter described the optimal behavior of an idealized plant, here we turn to a model of a soybean plant under a set of current and potential alternative environmental conditions. We ask and answer the following three key questions:

- What is the seasonal biomass accumulation by a soybean plant over an average growing season?
- How does biomass of the plant parts change over the season with different developmental stages?

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_20,

- How do changes in leaf assimilation rate under different atmospheric conditions impact growth and final biomass of the whole plant over the season, and does accelerated senescence in an elevated O_3 atmosphere impact overall soybean growth?

All stocks in our model (Fig. 20.1) are expressed in grams of carbon, and all flows are grams of carbon per hour. Leaf photosynthesis is calculated over the growing season by integrating the product of leaf assimilation rate, photoinhibition, specific leaf area, and a diurnal switch, which indicates whether the plant operates under day or night conditions. The diurnal cycle itself is specified using the built-in function MOD. X MOD Y gives the remainder when X is divided by Y. In our case, TIME MOD 24 is 3 in period 27, 4 in period 28, etc. Thus,

$$\text{Daily cycle} = (\text{TIME MOD } 24) + 1, \tag{20.1}$$

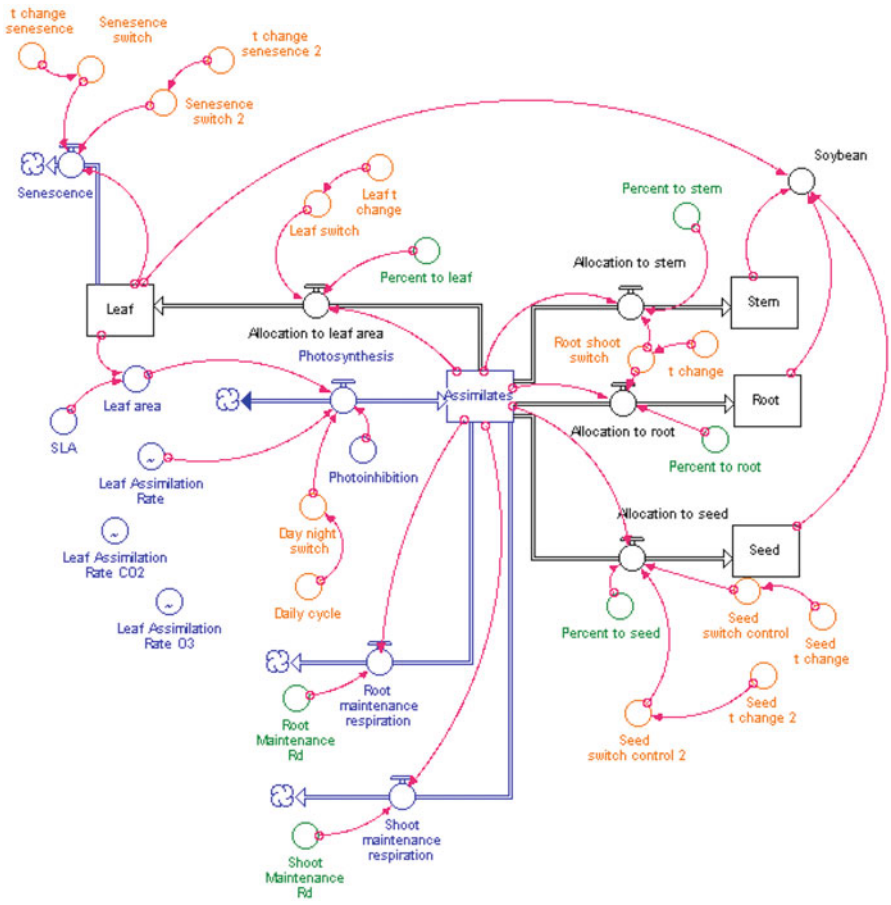

Fig. 20.1

creates a repeating series of counts from 1 to 24 to correspond to the hours in the day, and

$$\text{Day night switch} = \text{IF Daily cycle} <= 6 \text{ OR Daily cycle} >= 19 \text{ THEN 0 ELSE 1.} \tag{20.2}$$

These parameters generate a daily accumulation of assimilates. Assimilates are allocated among the plant parts according to standard percentages found in the literature [1]. Also included is carbon allocation to respiration of the roots and shoot. The rate and magnitude of allocation are controlled by switches, which also incorporate randomness during the growing season. Such randomness accounts for variations in plant development that may arise due to differences in soil water status, growing degree-days, and other environmental variables. A major feedback in the model is the accumulation of leaf area over the season and the subsequent rise in photosynthesis.

Senescence has two components. The initial component is a small loss of leaf mass occurring early in the season due to self-shading and loss of leaf area deep in the canopy. The larger component occurs later in the season as the plant matures. The timing of the senescence components also incorporates randomness. The sum of all the plant components feeds into soybean growth over the season.

Using this model, we are able to grow a single, small plant into a mature soybean under alternative environmental conditions. Under current atmospheric conditions, we assume a specific leaf area (SLA) = 0.0270 (area/mass) and the following initial values:

Assimilates = 0 g
Leaf = .09 g
Root = 0 g
Seed = 0 g
Stem = .1 g

The leaf assimilation rate, which is a key determinant of CO_2 uptake, is shown in Fig. 20.2.

Figure 20.3 demonstrates that there is a daily increase in the assimilated carbon that is subsequently exported at night to the various plant organs (Fig. 20.4). There is no build up of soluble carbon from 1 day to the next, which is consistent with actual total soluble carbohydrate measurements.

Let us now modify the model by incorporating elevated CO_2 concentrations. Growth in elevated CO_2 results in higher leaf assimilation rates and decreased SLA. Decreases in SLA are common in plants exposed to elevated CO_2 and reflect altered carbon contents. Photoinhibition also decreases in elevated CO_2. The greater final biomass of soybeans grown in elevated CO_2 reflects the beneficial decrease in photoinhibition.

The results are shown in Fig. 20.5, where we assume the same parameters as under current ambient conditions except that now Photoinhibition = 0.03, SLA = 0.023, and the Leaf Assimilation Rate CO_2 is as specified in Fig. 20.6 and

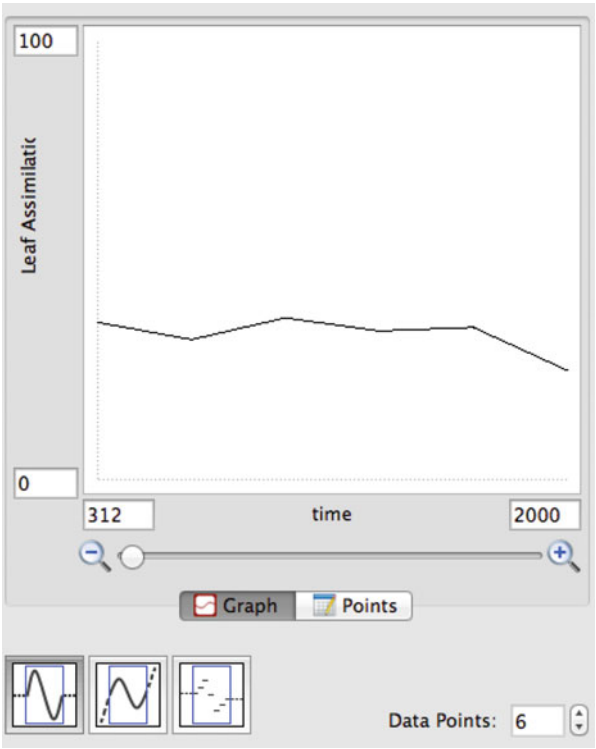

Fig. 20.2

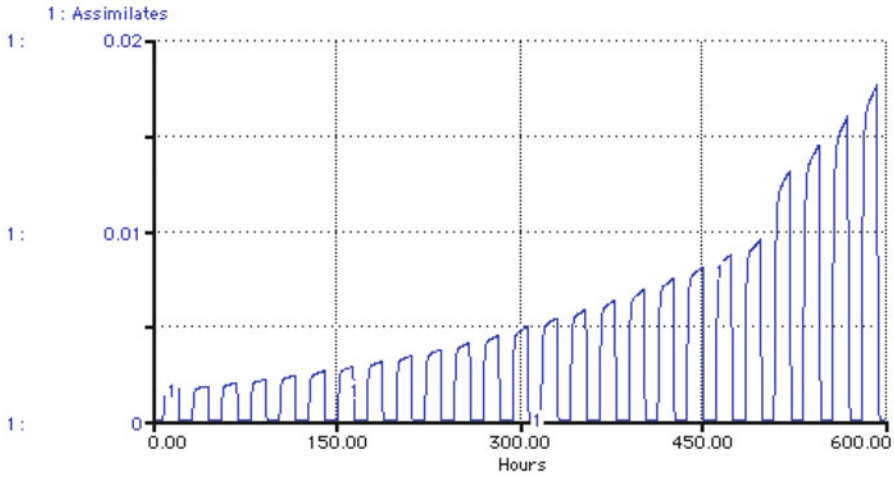

Fig. 20.3

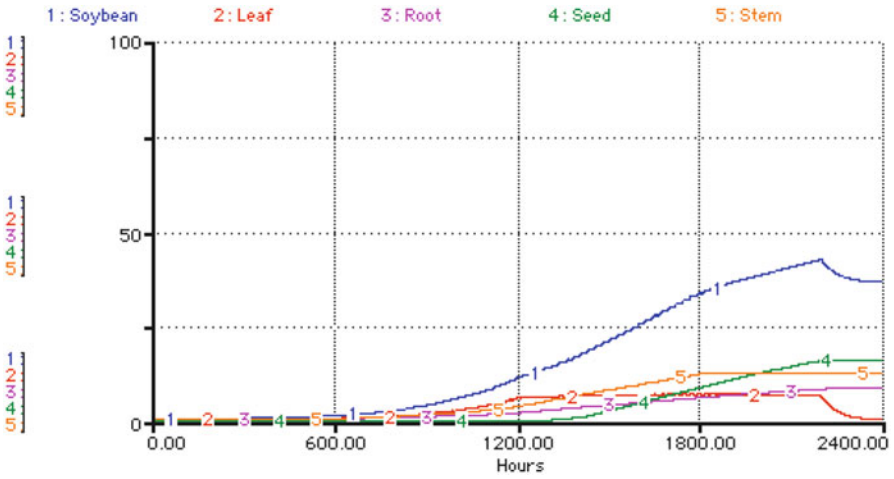

Fig. 20.4

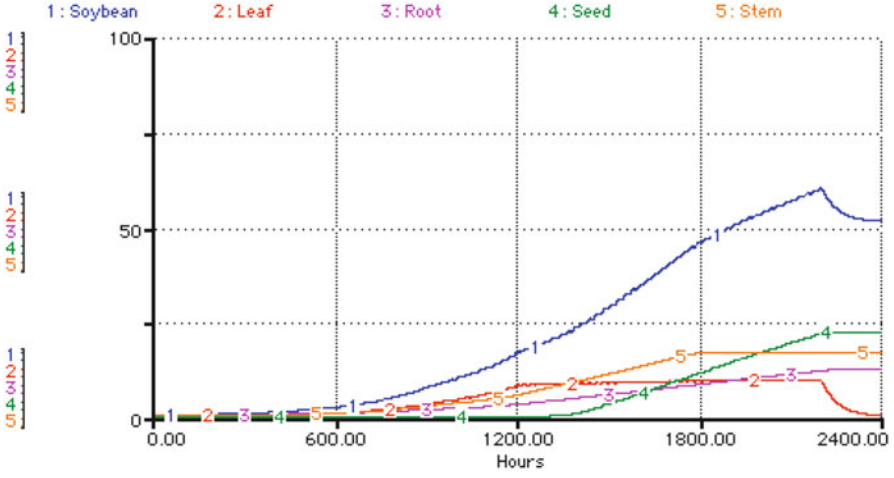

Fig. 20.5

used instead of the specification in Fig. 20.2. Can you explain the sudden downward sweep of curves 1 and 2 in this graph?

For elevated atmospheric O_3 we again assume the same conditions as in our first model run of Figs. 20.2 and 20.3 but adjust the parameters for seed maturity to be

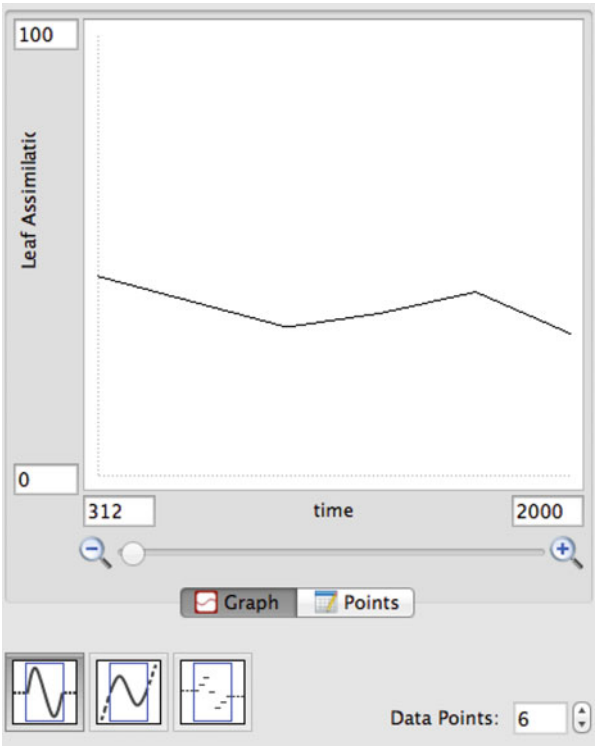

Fig. 20.6

reached at 2,000 h, leaf senescence due to plant maturity to begin randomly sometime between 1,800 and 2,250 h. We also adjust the leaf assimilation rate as shown in Fig. 20.7.

Ozone is a pollutant and damages plant tissue. Over time the accumulated damage results in faster senescence of the whole plant despite higher leaf assimilation than the plant grown in ambient air. As a result, of elevated O_3 conditions the soybean plant does not display a greater final biomass (Fig. 20.8). The faster decline in leaf area is clearly evident in these results.

Elevated atmospheric CO_2 and O_3 concentrations alter the development of plants. How sensitive is the model to alterations in the timing of developmental cues?

The leaf assimilation rate used in all the model runs above represents a maximum for light saturated leaves at the top of the plant. Up to 60 % of photosynthesis in a canopy occurs in light limited conditions. How would incorporating the dynamics of light changes around a plant in the canopy impact individual plant growth?

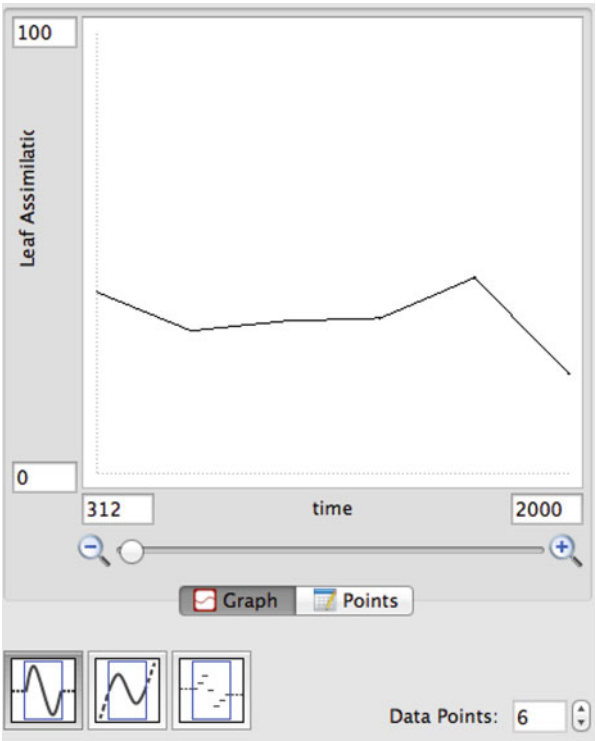

Fig. 20.7

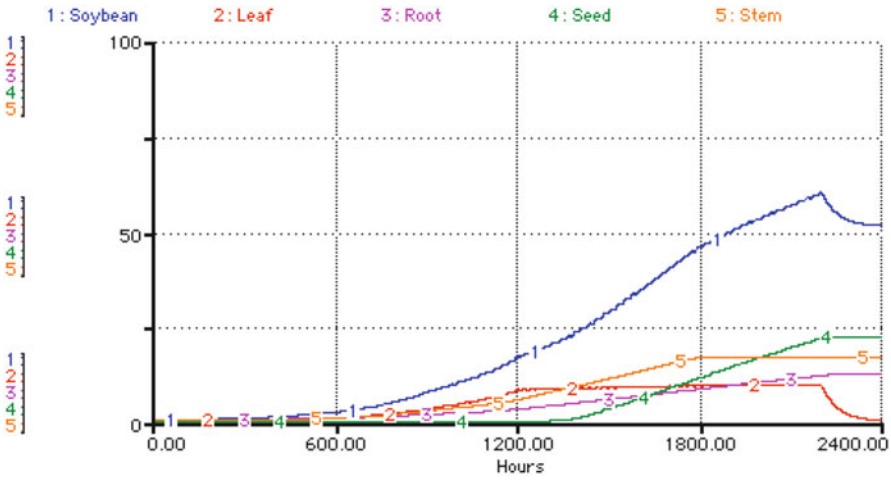

Fig. 20.8

20.2 Soybean Plant Growth Model Equations

Assimilates(t) = Assimilates(t − dt) + (photosynthesis − Allocation_to_stem − Allocation_to_root − Allocation_to_seed − Allocation_to_Leaf_Area − Shoot_Maintenance_Respiration - Root__Maintenance_Respiration) ∗ dt
INIT Assimilates = 0
INFLOWS:
photosynthesis = leaf_area∗((Leaf_Assimilation_Rate∗3600)∗0.000012) ∗day_night__switch- (photoinhibition∗3600)∗0.000012
OUTFLOWS:
Allocation_to_stem = if time <= 1800 then Assimilates∗percent_to_stem∗root_shoot__switch else 0
Allocation_to_root = if time <= 500 then Assimilates∗percent_to_root∗root_shoot__switch else Assimilates∗0.08∗root_shoot__switch
Allocation_to_seed = (1-seed__Switch_control)∗(1-seed__Switch_control_2) ∗Assimilates∗percent_to_seed
Allocation_to_Leaf_Area = if time <= 2200 then Assimilates∗percent_to_leaf ∗leaf_switch else 0
Shoot_Maintenance_Respiration = (Assimilates∗Shoot_Maintenance_Rd)
Root__Maintenance_Respiration = (Assimilates∗Root_Maintenance_Rd)
Leaf(t) = Leaf(t − dt) + (Allocation_to_Leaf_Area - Senescence) ∗ dt
INIT Leaf = 0.09
INFLOWS:
Allocation_to_Leaf_Area = if time <= 2200 then Assimilates∗percent_to_leaf ∗leaf_switch else 0
OUTFLOWS:
Senescence = if time <=2200 then leaf∗senesence_switch∗0.002 else leaf ∗senesence_switch_2∗.02
Root(t) = Root(t − dt) + (Allocation_to_root) ∗ dt
INIT Root = 0
INFLOWS:
Allocation_to_root = if time <= 500 then Assimilates∗percent_to_root∗root_shoot__switch else Assimilates∗0.08∗root_shoot__switch
Seed(t) = Seed(t − dt) + (Allocation_to_seed) ∗ dt
INIT Seed = 0
INFLOWS:
Allocation_to_seed = (1−seed__Switch_control)∗(1−seed__Switch_control_2) ∗Assimilates∗percent_to_seed
Stem(t) = Stem(t − dt) + (Allocation_to_stem) ∗ dt
INIT Stem = 0.1
INFLOWS:
Allocation_to_stem = if time <= 1800 then Assimilates∗percent_to_stem∗root_shoot__switch else 0
Daily_cycle = (TIME MOD 24)+1

Day_night__switch = IF Daily_cycle <= 6 OR Daily_cycle >= 19 THEN 0 ELSE 1
Leaf_area = Leaf*SLA
Leaf_Assimilation_Rate = GRAPH(time)
(312, 35.5), (650, 31.6), (987, 36.5), (1325, 33.5), (1662, 34.3), (2000, 24.6)
Leaf_Assimilation_Rate_CO2 = GRAPH(time)
(312, 44.8), (650, 39.1), (987, 33.3), (1325, 36.5), (1662, 41.3), (2000, 31.9)
Leaf_Assimilation_Rate_O3 = GRAPH(time)
(312, 40.9), (650, 32.1), (987, 34.3), (1325, 35.0), (1662, 44.1), (2000, 22.3)
Leaf_switch = if time <= leaf_t_change then 1 else 0.35
Leaf_t_change = 1200
Percent_to_leaf = 0.2
Percent_to_root = 0.3
Percent_to_seed = 0.4
Percent_to_stem = 0.17
Photoinhibition = 0.05
Root_Maintenance_Rd = 0.22
Root_shoot__switch = if time <= t_change then 1 else 0.4
Seed__switch_control = if time <= seed__t_change then 1 else 0
Seed__switch_control_2 = if time <= seed__t_change_2 then 0.75 else 1
Seed__t_change = RANDOM(1300, 1450)
Seed__t_change_2 = 2200
Senesence_switch = if time <= t_change_senesence then 0 else 1
Senesence_switch_2 = if time <= t_change_senesence_2 then 0 else 1
Shoot_Maintenance_Rd = 0.18
SLA = 0.027
Soybean = root+Leaf+Seed+stem
t_change = RANDOM(850, 975)
t_change_senesence = RANDOM(850, 975)
t_change_senesence_2 = RANDOM(1900, 2250)

Reference

1. Lambers H, Chapin III FS, Pons TL (2008) Plant physiological ecology, 2nd edn., Springer, New York, p 605

Chapter 21
Infectious Diseases

The endemic and epidemic diseases in Scotland fall chiefly, as is usual, on the poor.

(Thomas Malthus, 1798)

21.1 Basic Epidemic Model

In this model we consider the spread of an infectious disease within a population. We assume that there is some initial number of individuals already infected with the disease. These individuals can pass on the disease to a group of susceptibles S. We do not model explicitly the agents that cause the disease, such as viruses or bacteria. Doing that would be rather impractical if we would want to apply our model to real world diseases. Tracing the billions of agents that can cause the outbreak with a particular disease is virtually impossible. Therefore, we do not explicitly model the dynamics of individuals in a population of disease-causing agents but deal with their effects in an aggregate way.

The law of mass action discussed in Part II of this book has proven to be a powerful analogous way of capturing the spread of a disease in a population. The two "reactants" in our case are the susceptible individuals S and the infective ones I. We define a contact rate BETA at which these two groups of individuals make contact and propagate the disease. This contact rate BETA is analogous to the reaction rates in chemical reactions.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_21,

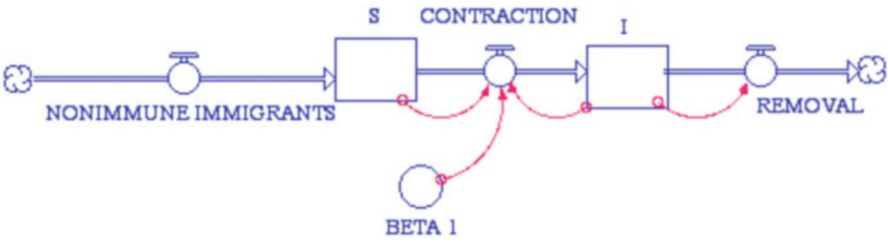

Fig. 21.1

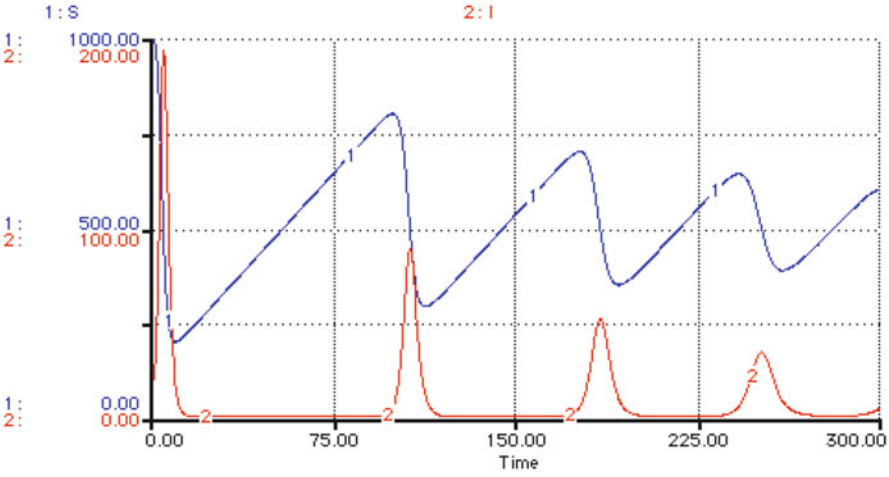

Fig. 21.2

We also model an influx of susceptibles into the stock S. Additionally, we assume that once an individual had the disease that individual ultimately becomes immune to the disease. We therefore remove those infectives from the stock I and let them disappear in a "cloud"—they will not further affect the spread of the disease and, therefore, we need not keep track of them (Fig. 21.1).

This is a very simple epidemic model—but can you anticipate the resulting dynamics? Make an educated guess before you run the model. How do the dynamics change with a change in the rate of NONIMMUNE IMMIGRANTS. This rate is here set to 7 per time period, and the initial stocks for S and I are 1,000 and 20, respectively (Fig. 21.2).

What are the effects of a vaccine on the spread of the disease? Assume that only 20 % of the population receives the vaccine and that it is only 90 % effective among the susceptibles. Infectives are not immunized. Change the model to include an incubation period and reassess the ability of your vaccine to limit the spread of the disease.

21.2 Basic Epidemic Model Equations

I(t) = I(t − dt) + (CONTRACTION − REMOVAL) ∗ dt
INIT I = 20
INFLOWS:
CONTRACTION = BETA∗S∗I
OUTFLOWS:
REMOVAL = I

S(t) = S(t − dt) + (NONIMMUNE_IMMIGRANTS − CONTRACTION) ∗ dt
INIT S = 1000
INFLOWS:
NONIMMUNE_IMMIGRANTS = 7
OUTFLOWS:
CONTRACTION = BETA∗S∗I

BETA = .002

21.3 Two Infective Populations

Let us expand on the model of the previous section and assume that an individual breaks out with the disease upon contact with a virus that can either be carried by members of the same population or by organisms of another species. Prominent examples are the Marburg and Ebola viruses that can spread from monkeys to humans. For a powerful description of the dynamics of these viruses see, for example, Preston [1].

We begin our model with the set up of the previous section and duplicate it to capture the spread of the disease in the second population and from that population to the other one. To duplicate the STELLA model of the previous section, select the entire model by choosing Select All from the Edit menu and then copy it. Then make the necessary changes in the names of the variables and the connections among the two model parts. Notice that we only captured here the one-way movement of the virus from the infective stock I2 to S1 (Fig. 21.3). You can easily explore the case of the virus spreading from any of the two populations to the other.

For the first model run we set the parameters as follows:

Variable	Value	Explanation
S1(t = 0)	1,000	Initial stock of susceptibles in population 1
I1(t = 0)	20	Initial number of infective individuals in population 1
BETA 1	0.008	Contact rate of S1 with I1
SURVIVAL RATE 1	0.065	Rate of survival upon contact of S1 with disease
S2(t = 0)	1,000	Initial stock of susceptibles in population 1
I2(t = 0)	20	Initial number of infective individuals in population 1
BETA 2	0.003	Contact rate of S2 with I2
BETA 2 1	0.00015	Contact rate of S1 with I2
SURVIVAL RATE 2	0.2	Rate of survival upon contact of S1 with disease

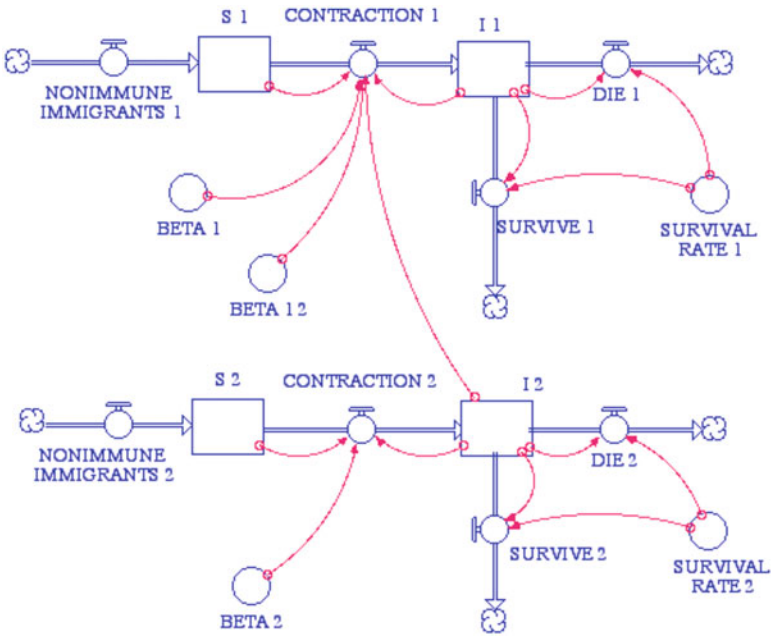

Fig. 21.3

We set the number of nonimmune immigrants for the two populations to 7 and 10, respectively. All of these numbers are hypothetical.

In Fig. 21.3, we used the following specifications to calculate the number of individuals that contract the virus:

$$\text{CONTRACTION 1} = \text{BETA 1} * \text{S 1} * \text{I 1} + \text{BETA 1 2} * \text{S 1} * \text{I 2} \tag{21.1}$$

$$\text{CONTRACTION 2} = \text{RATE OF CONTACT 2} * \text{I 2} * \text{S 2} \tag{21.2}$$

Can you anticipate the dynamics of the spread of the disease? The initial outbreak will be larger than that of the previous model because the number of susceptibles and infective individuals is larger, although the contact rates are quite a bit smaller. What matters, though, is the product of contact rates and sizes of the individual stocks.

Figure 21.4 shows the results for the parameter settings listed above and a choice of DT = 1. Following an initially severe outbreak, new episodes of the disease occur at relatively constant intervals and slightly increase in their amplitude. After only five episodes of outbreaks of the disease among S1, the disease "burns out" and disappears from the population. Can you explain why in the very long run the disease disappears? Plot S2 and I2 in a separate graph to help you find an answer.

Let us assume that the parameters listed above are representative of one of two strains of the virus. The first strain—modeled above—does not move easily from

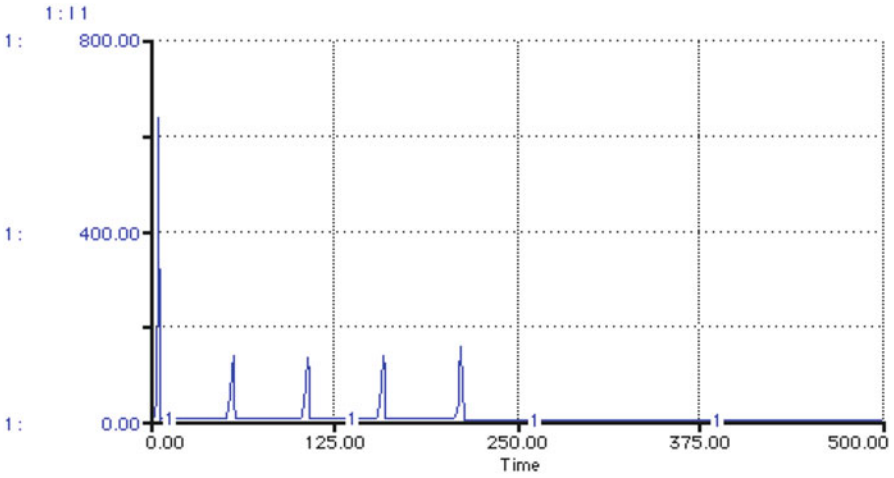

Fig. 21.4

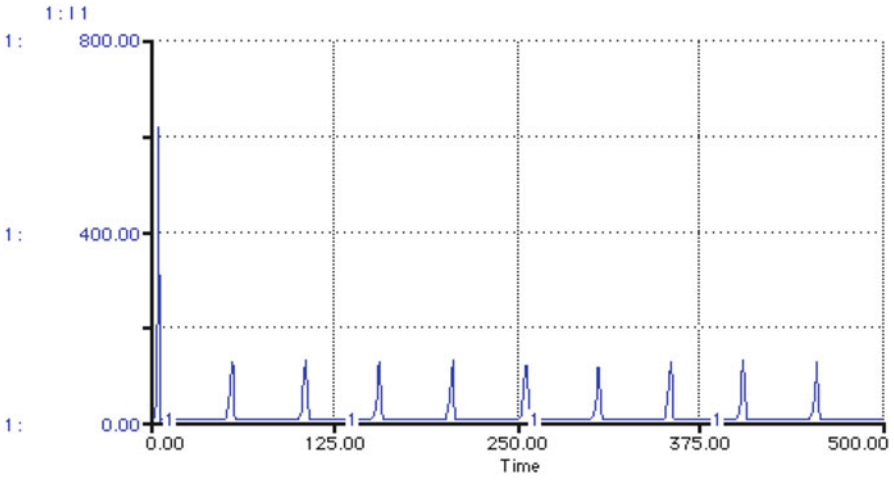

Fig. 21.5

one population to the other, but when it does, it results in only very small rates of survival. You may think of this strain as one that can only be passed on through direct contact with bodily fluids such as saliva or blood. In contrast, the other strain of the virus can travel through air. It is therefore more easy to contract the disease. But this strain is also less lethal. Assume BETA 1 2 = 0.00025, SURVIVAL RATE 1 = 0.155, and all the other parameters as listed above. The first outbreak with the disease is virtually the same for both strains but the subsequent dynamics are very different. The virus of the second strain can stay in the population for long times (Fig. 21.5). Can you explain why? Perform a sensitivity analysis of the contact rates

BETA 1, BETA 2 1, and BETA 2. How are the results affected by the choice of survival rates? How does the outbreak pattern of the disease change for smaller DT and for different integration methods?

21.4 Two Infective Populations Model Equations

I_1(t) = I_1(t − dt) + (CONTRACTION_1 − SURVIVE_1 − DIE_1) ∗ dt
INIT I_1 = 5

INFLOWS:
CONTRACTION_1 = BETA_1∗S_1∗I_1 + BETA_1_2∗S_1∗I_2
OUTFLOWS:
SURVIVE_1 = SURVIVAL_RATE_1∗I_1
DIE_1 = (1−SURVIVAL_RATE_1)∗I_1
I_2(t) = I_2(t − dt) + (CONTRACTION_2 − DIE_2 − SURVIVE_2) ∗ dt
INIT I_2 = 20

INFLOWS:
CONTRACTION_2 = RATE_OF_CONTACT_2∗I_2∗S_2
OUTFLOWS:
DIE_2 = (1-SURVIVAL_RATE_2)∗I_2
SURVIVE_2 = SURVIVAL_RATE_2∗I_2
S_1(t) = S_1(t − dt) + (NONIMMUNE_IMMIGRANTS_1 − CONTRACTION_1) ∗ dt
INIT S_1 = 1000

INFLOWS:
NONIMMUNE_IMMIGRANTS_1 = 7
OUTFLOWS:
CONTRACTION_1 = BETA_1∗S_1∗I_1 + BETA_1_2∗S_1∗I_2
S_2(t) = S_2(t − dt) + (NONIMMUNE_IMMIGRANTS_2 − CONTRACTION_2) ∗ dt
INIT S_2 = 1000

INFLOWS:
NONIMMUNE_IMMIGRANTS_2 = 10
OUTFLOWS:
CONTRACTION_2 = BETA_2∗I_2∗S_2
BETA_1 = .008
BETA_1_2 = 0.00015
BETA_2 = .003
SURVIVAL_RATE_1 = .065
SURVIVAL_RATE_2 = .2

21.5 Temporary Immunity

In the previous sections of this chapter we have modeled a disease that spreads in a population by turning susceptible, nonimmune individuals into infective ones. Following infection, the individuals were assumed to either die or remain immune for the rest of their lives. In both cases, the outflows from the stock of infective individuals disappeared into clouds—we did not keep track of them. We need to change this setup if we wish to model the case of a disease that only leads to temporary immunity. In that case, the flow of individuals who survive must end in a reservoir, rather than a cloud. What are the effects of temporary immunity on the population modeled in Sect. 20.1?

Assume that once an individual becomes infected with the virus, that individual will either die from the disease or survive. The survival rate is 90 % per time and there are no other causes of death. Those individuals that survive become temporarily immune. A fraction of the stock of temporarily immune individuals, T, will become sick again with the disease at a given RECURRENCE RATE, and again pass on the virus to the susceptible population (Fig. 21.6).

The effect of temporary immunity is an overall larger stock of infective individuals. The following results are plotted for a RECURRENCE RATE ranging from 0.02 to 0.1. The level of the initial outbreak of the disease is different under each of these rates, but the remaining dynamics show convergence towards the same steady-state level (Fig. 21.7).

Change the duration of temporary immunity and observe the results. Introduce an incubation period for the disease and vaccination program as you did in first section of this chapter.

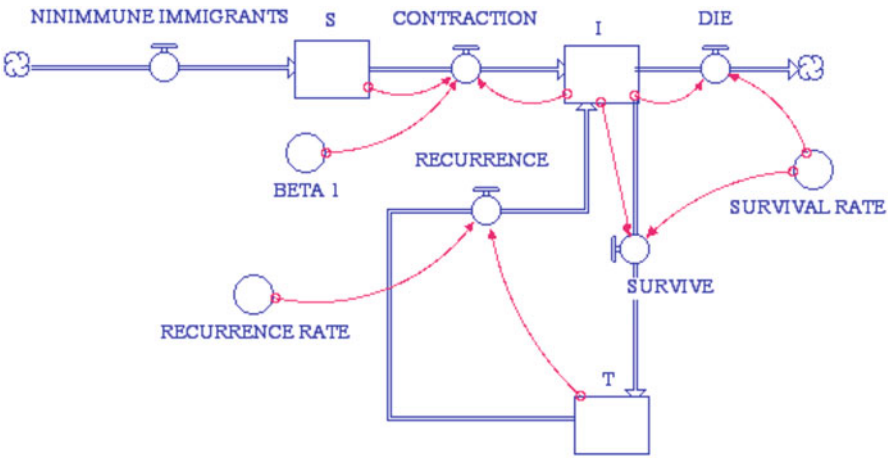

Fig. 21.6

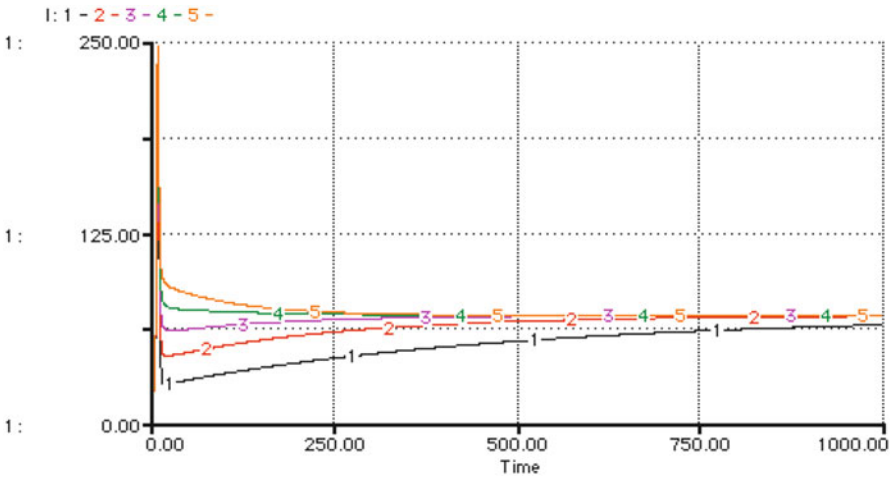

Fig. 21.7

Disaggregate your model to deal with the case of a virus that affects different age cohort in the population differently. An example for such a disease is chicken pox. Chicken pox is a highly infectious childhood disease. It is caused by the varicella zoster virus. This virus can be spread either through direct contact with infected individuals or through the air. After exposure to the virus, the incubation period before an individual becomes contagious is approximately seven days. Individuals are contagious for about seven days during which symptoms including fever and blisters appear, and then remain sick for an additional fourteen days. Once an individual recovers he or she develops a natural immunity and is unlikely to get the disease again.

Later in life, the varicella zoster virus manifests itself in the form of shingles in about 15 % of the population that contracted chicken pox. Shingles has symptoms that are similar to chicken pox but strikes mostly individuals over the age of fifty that are fatigued or under stress. It takes approximately ten days to recover from shingles and during this time susceptible individuals can contract chicken pox from those suffering from shingles.

Recently, a vaccine has been approved to immunize people against chicken pox. The target population for immunization are children, as they comprise the highest infectious class. What are the effects of chicken pox on a given population and how do those effects change as immunization takes place? What are the effects of immunization during childhood on the occurrence of shingles? How does an immunization policy effect the average age at which a person contracts the disease? Set up a model to provide answers to these questions. To achieve some realism with your model, consult the literature for parameter values, such as Edelstein-Keshet [2], Finger et al. [3], May [4], and Hethcote [5].

21.6 Temporary Immunity Model Equations

I(t) = I(t − dt) + (CONTRACTION + RECURRENCE − DIE − SURVIVE) ∗ dt
INIT I = 20

INFLOWS:
CONTRACTION = BETA∗S∗I
RECURRENCE = RECURRENCE_RATE∗T
OUTFLOWS:
DIE = (1−SURVIVAL_RATE)∗I
SURVIVE = SURVIVAL_RATE∗I
S(t) = S(t − dt) + (NINIMMUNE_IMMIGRANTS − CONTRACTION) ∗ dt
INIT S = 1000

INFLOWS:
NINIMMUNE_IMMIGRANTS = 7
OUTFLOWS:
CONTRACTION = BETA∗S∗I
T(t) = T(t − dt) + (SURVIVE − RECURRENCE) ∗ dt
INIT T = 10

INFLOWS:
SURVIVE = SURVIVAL_RATE∗I
OUTFLOWS:
RECURRENCE = RECURRENCE_RATE∗T
BETA = .002
RECURRENCE_RATE = .1
SURVIVAL_RATE = .9

21.7 Epidemic with Vaccination

Let us further expand on the models of the previous sections and introduce a number of features that make those models more meaningful. Among these features are

- The explicit inclusion of birth rates;
- Death rates that are not only influenced by the disease but that result also from natural mortality;
- A vaccination program which allows the population to become immune to the disease without having to first be sick;
- Mutations in the disease that result in immune people not staying immune forever; and
- Ignorance of a fixed portion of the contagious population. These people are assumed not to know that they carry the disease. Consequently, we assume that

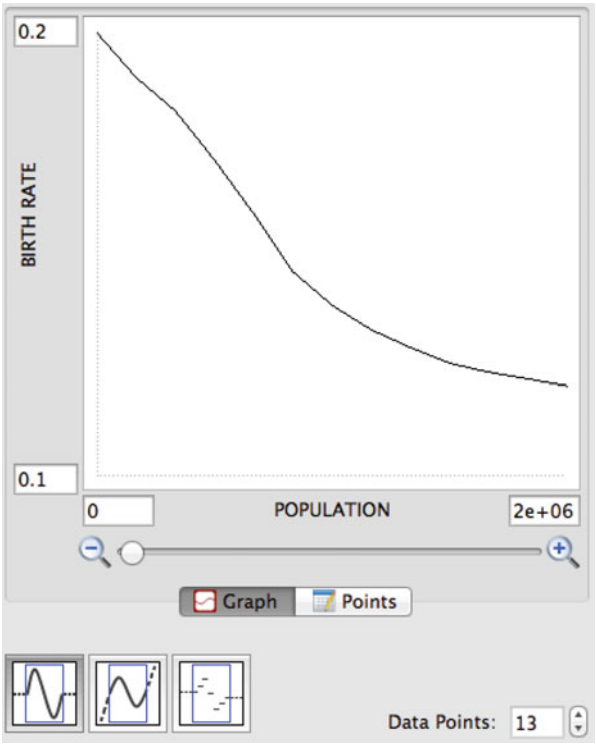

Fig. 21.8

ignorance would increase the rate at which the disease gets passed on from the infective to the susceptible population.

The birth and natural death rates are specified graphically in this model as shown in Figs. 21.8 and 21.9 respectively.

CONTRACTING is a function of susceptible, healthy people coming into contact with people who are aware that they are contagious mixed in with people who are immune:

$$\text{CONTRACTING} = \text{IF IMMUNE} > 0 \text{ THEN CC} * \text{NON\_IMM} * \text{CONTAGIOUS ELSE } 0 \tag{21.3}$$

The VACCINATING biflow is specified as

$$\text{VACCINATING} = \text{VAC\_FRACTION} * \text{NON\_IMM} \tag{21.4}$$

and contains the rate of vaccination. A flow from IMMUNE to NONIMMUNE captures 1.5 % of immune people that lose their immunity (Fig. 21.10).

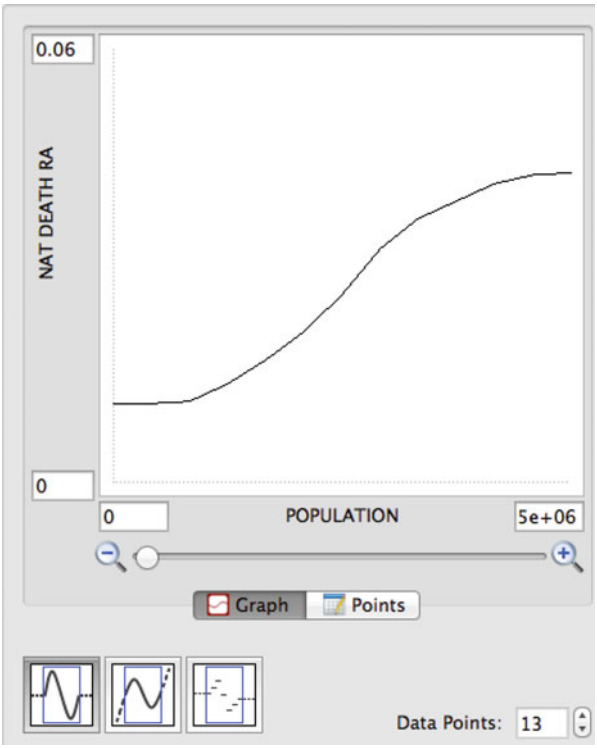

Fig. 21.9

Consistent with the model of the previous section, there are a series of epidemic outbreaks (Fig. 21.11). Due to the additional features of this model, however, the numbers of immune and nonimmune people tends to rise as does the number of sick. This rise is due to the differences between the birth and death rates.

Introduce a spatial component into the model by considering two regions with different contact rates and different vaccination programs. Investigate the implications of travel restrictions imposed by one of the regions on people originating in the other region.

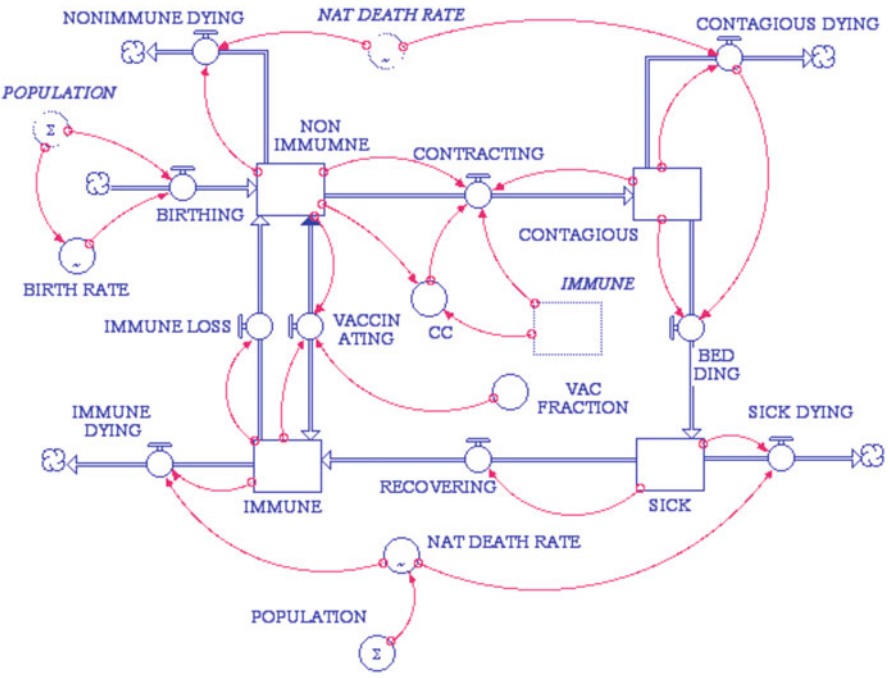

Fig. 21.10

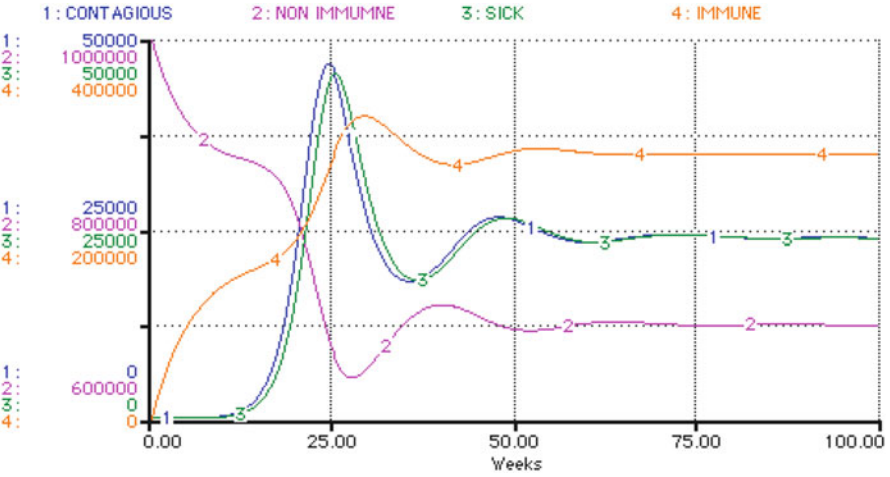

Fig. 21.11

21.8 Epidemic with Vaccine Model Equations

CONTAGIOUS(t) = CONTAGIOUS(t − dt) + (CONTRACTING − CONTAGIOUS_DYING − BED_DING) ∗ dt
INIT CONTAGIOUS = 1 {People}
INFLOWS:
CONTRACTING = IF IMMUNE > 0 THEN CC∗NON_IMMUMNE∗CONTAGIOUS ELSE 0
OUTFLOWS:
CONTAGIOUS_DYING = CONTAGIOUS∗NAT_DEATH_RATE/52 {People per Week}
BED_DING = CONTAGIOUS-CONTAGIOUS_DYING {People per Week}
IMMUNE(t) = IMMUNE(t − dt) + (RECOVERING + VACCIN_ATING − IMMUNE__DYING - IMMUNE_LOSS) ∗ dt
INIT IMMUNE = 0 {People}
INFLOWS:
RECOVERING = 0.9∗SICK {People per Week}
VACCIN_ATING = VAC_FRACTION∗NON_IMMUMNE - 0.15∗IMMUNE
OUTFLOWS:
IMMUNE__DYING = IMMUNE∗NAT_DEATH_RATE/52 {People per Week}
IMMUNE_LOSS = .002∗IMMUNE {People per Week}
NON_IMMUMNE(t) = NON_IMMUMNE(t − dt) + (BIRTHING + IMMUNE_LOSS − CONTRACTING − NONIMMUNE_DYING − VACCIN_ATING) ∗ dt
INIT NON_IMMUMNE = 1000000 {People}
INFLOWS:
BIRTHING = POPULATION∗BIRTH_RATE/52 {People per Week}
IMMUNE_LOSS = .002∗IMMUNE {People per Week}
OUTFLOWS:
CONTRACTING = IF IMMUNE > 0 THEN CC∗NON_IMMUMNE∗CONTAGIOUS ELSE 0
NONIMMUNE_DYING = NAT_DEATH_RATE/52∗NON_IMMUMNE {People per Week}
VACCIN_ATING = VAC_FRACTION∗NON_IMMUMNE − 0.15∗IMMUNE
SICK(t) = SICK(t - dt) + (BED_DING − RECOVERING − SICK_DYING) ∗ dt
INIT SICK = 0 {People}
INFLOWS:
BED_DING = CONTAGIOUS-CONTAGIOUS_DYING {People per Week}
OUTFLOWS:
RECOVERING = 0.9∗SICK {People per Week}
SICK_DYING = (.1∗SICK)+(NAT_DEATH_RATE/52∗SICK) {People per Week}
BIRTH_RATE = GRAPH(POPULATION)
(0.00, 0.199), (166667, 0.189), (333333, 0.181), (500000, 0.171), (666667, 0.159), (833333, 0.146), (1e+06, 0.138), (1.2e+06, 0.133), (1.3e+06, 0.129), (1.5e+06, 0.125), (1.7e+06, 0.123), (1.8e+06, 0.122), (2e+06, 0.12)

CC = 0.000002*(1 − 1*IMMUNE/(IMMUNE+NON_IMMUMNE)) {+SINWAVE (0.0000005,52) }
NAT_DEATH_RATE = GRAPH(POPULATION)
(0.00, 0.0108), (208333, 0.0108), (416667, 0.0111), (625000, 0.0135), (833333, 0.0168), (1e+06, 0.0207), (1.2e+06, 0.0258), (1.5e+06, 0.0321), (1.7e+06, 0.0363), (1.9e+06, 0.0387), (2.1e+06, 0.0411), (2.3e+06, 0.0423), (2.5e+06, 0.0426)
VAC_FRACTION = 0.03
POPULATION = CONTAGIOUS + IMMUNE + NON_IMMUMNE + SICK

References

1. Preston R (1995) The hot zone. Anchor Books, New York
2. Edelstein-Keshet L (1988) Mathematical models in biology. Random House, New York
3. Finger R, Hughes J, Meade BJ, Pelletier AR, Palmer CT (1994) Age-specific incidence of chickenpox. Public Health Rep 109(6):750–755
4. May RM (1983) Parasitic infections as regulator of animal populations. Am Sci 71(1):36–45
5. Hethcote HW (1976) Qualitative analyses of communicable disease models. Math Biosci 28 (3–4):335–356

Part V
Single Population Models

Chapter 22
Adaptive Population Control

To make increased population the cause of improved agriculture, is to commit the absurd blunder of confounding cause and effect.

(Rogers, Political Economy, 1868)

22.1 Adaptive Population Control Model

In this chapter we model the action of a population that is collectively trying to control their population size by imperfectly recalling much of what they have done in terms of birth rate control over the recent past. They assess the gap between their current population size and that size dictated by their physical environment. It takes time to gain the information about these two population sizes. Once the population knows these levels, we assume they react by changing their birth rate. The new birth rate is an average of the ones remembered over the recent past. The death rate for this population and the level of the desired population size are dependent on the population density.

Figure 22.1 shows that part of the model that captures the relationships among area, population density, death rates, and desired population level. Here we also introduce a multiplier A to scale the variable area for alternative runs. By defining population density as

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_22,

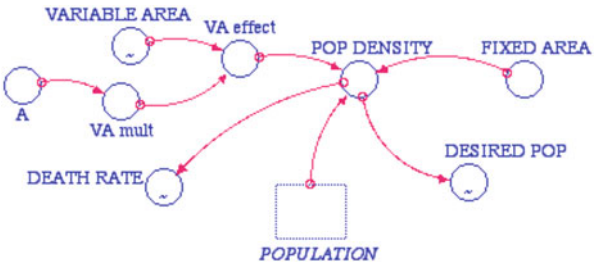

Fig. 22.1

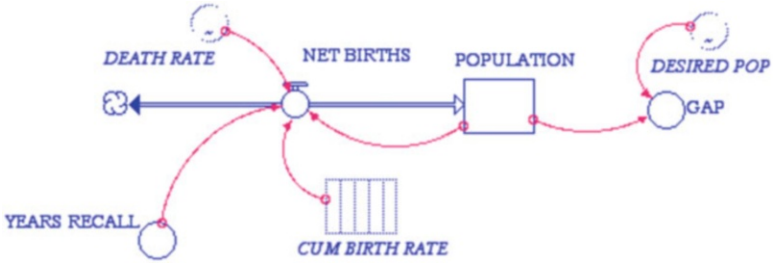

Fig. 22.2

$$\text{POP DENSITY} = \text{POPULATION}/(\text{VA effect} * 1 + .0001 + \text{FIXED AREA} * 0) \tag{22.1}$$

We can make use of either a fixed area or a variable area in our calculation of the population density simply by multiplying one of them by one and the other by zero. With this setup, the model can easily be run under alternative assumptions without having to re-do parts of the diagram.

Figure 22.2 shows that part of the model that traces the birth rate history and the extent to which that history is remembered. The resulting population and the desired population then define a gap that is used to set the new birth rate, given some sensitivity with which the population is able to react to the size of this gap (Fig. 22.3). The gap itself is defined as the difference between the desired population size and the size of the population two time periods earlier, using the DELAY built in function:

$$\text{GAP} = \text{DESIRED POP} - \text{DELAY}(\text{POPULATION}, 2) \tag{22.2}$$

In essence, we assume here that it takes two periods of time to obtain a population census.

The GAP is then normalized (Fig. 22.3) by the current population level and multiplied by a reaction sensitivity to get the fractional birth rate (FRACTIONAL BR).

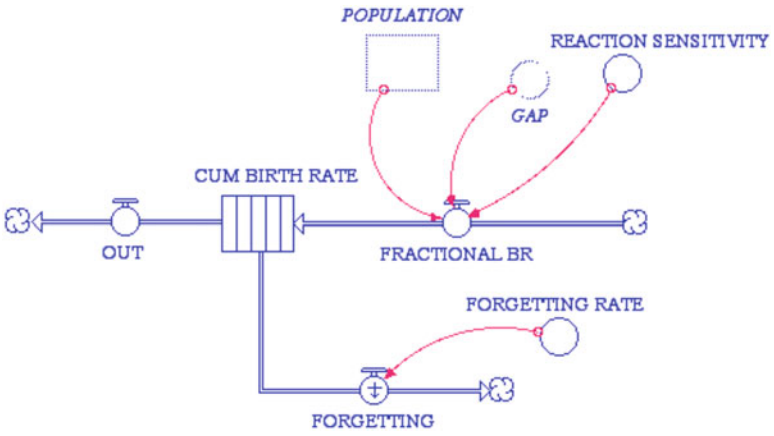

Fig. 22.3

We now wish to keep track of these FRACTIONAL BRs for a length of time called the YEARS RECALL. This part of the model captures the "adaptive control"—the influence of the remembered composite history of the system on its current and future performance.

To keep track of the FRACTIONAL BRs, we use a stock that accumulates the birth rates. We call this stock CUM BIRTH RATE. However, we do not specify this stock as a reservoir but as a conveyor to retain information on the birth rates and then dump the oldest values on the conveyor, those that have been on the conveyor for YEARS RECALL. We also draw a second outflow from the conveyor and call it FORGET. The leakage—or exponential forgetting of the collected birth rate signals stored in the conveyor CUM BIRTH RATE—is controlled by a FORGETTING RATE. We set it here to 0.05, i.e., 5 % of the information is lost from all slats on the conveyor. Had we wanted to reserve this forgetting process to some portion of the conveying process, we could set the "No Leak Zone" of the conveyor at something less than the full length in the FORGET variable.

The CUM BIRTH RATE, the sum of all the birth rates on the various conveyor slats, is sent to NET BIRTHS where it is divided by the YEARS RECALL to get an average rate, and then compared to the death rate to calculate a net birth rate. This net birth rate is then multiplied by the population level to find the addition or subtraction to the total POPULATION.

The graphical functions of Figs. 22.4 and 22.5 are used to express the relationships between population density and death rate and between density and desired population. Figure 22.6 shows how land area varies over time.

Figure 22.7 shows the case of a variable area while Fig. 22.8 depicts the population dynamics under the assumption of a constant area over time, varying that amount of land between 10 and 80 for five successive runs. For all but one of the cases, the result is a population that ultimately is damped to reach a steady state. For a fixed land area of 10, the population keeps oscillating.

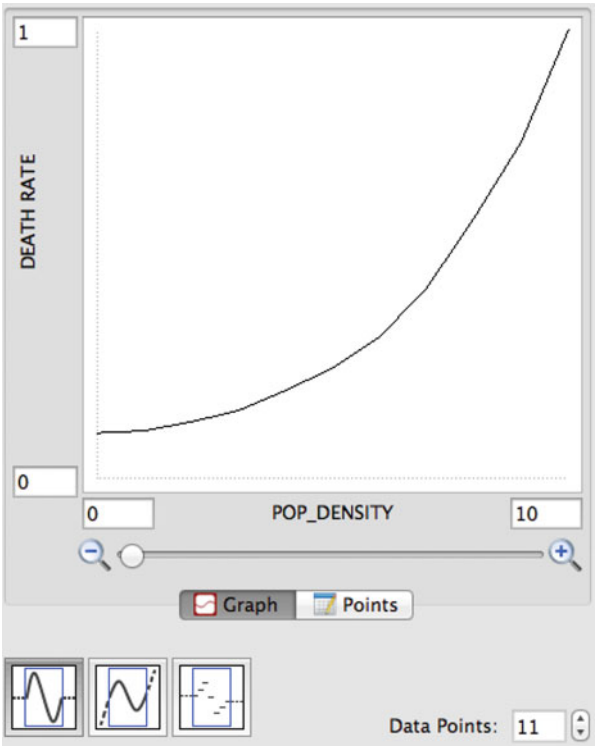

Fig. 22.4

Of course part of the initial fluctuations in population sizes is due to the fact that there is no initial history of the birth rates in this population. To this extent the model might be thought unrealistic. Yet suppose that we had a population that was initially stabilized at for many years, and then due to a technological change or a conquest of new lands, was suddenly exposed to the given amounts of land. In this case the model is more realistic. Such a population would have no remembrance of any adaptation process to sudden new resource availability. But we show that as long as they are trying to adapt to the available resources (close the GAP), they will eventually reach a new steady state. In essence, this population is remembering a process of adaptation, but not the details. A more sophisticated model would allow them to remember how much social trauma the cycling produced and the next period of sudden resource increase would be met with less extreme peaks and lows.

In general we are trying to capture here the idea that a population reacts to an average remembered history of efforts to achieve some goal, in this case, closing the gap between actual and desired levels of the population. Our approach to the final steady state might be framed as an optimality problem: What controls (birth rates) do we use to proceed to the steady state within a certain time, perhaps with certain constraints such as avoiding population decline while rising to the ultimate

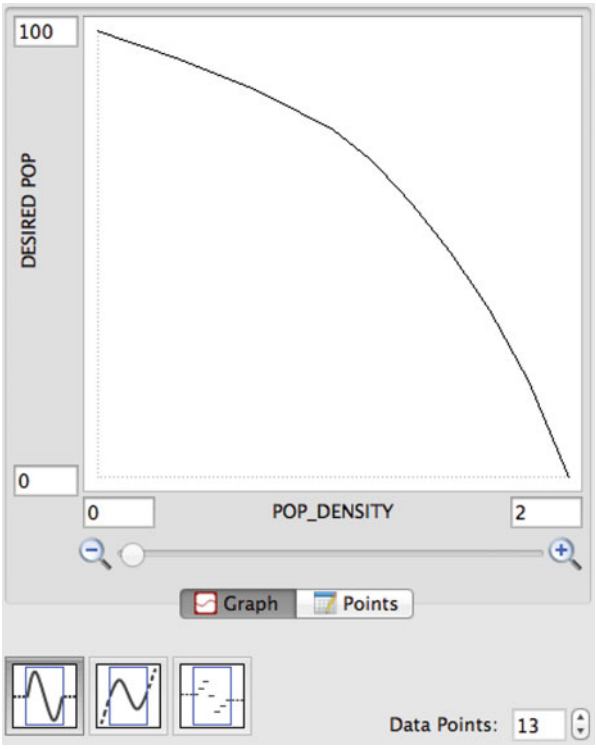

Fig. 22.5

steady-state level? This constraint may be appropriate given the trauma of reducing populations. The problem with the optimality proposal above though is that we don't know a priori, what the final steady state is going to be, given all these parameters. The final steady-state population is a function of the very parameters that we wish to change in order to control the ascent to the steady state. We could use as an optimality goal, the rise (only) in a prescribed time to *a* steady state and not be partial to the actual value of that steady state. In that case we would want to know which of the controllable parameters are accessible—in our case here, only the forgetting rate, the number of years of recall, and perhaps the reaction sensitivity. Then we wish to know which of these causes a damping in the system. We would next set those that cause damping such that there is no oscillation in the population. This slope constraint is probably the best goal to use: when that value gets sufficiently low, we have reached the goal—we are sufficiently close to the steady state. Try out this idea and ones of your own to achieve the "no oscillation" goal.

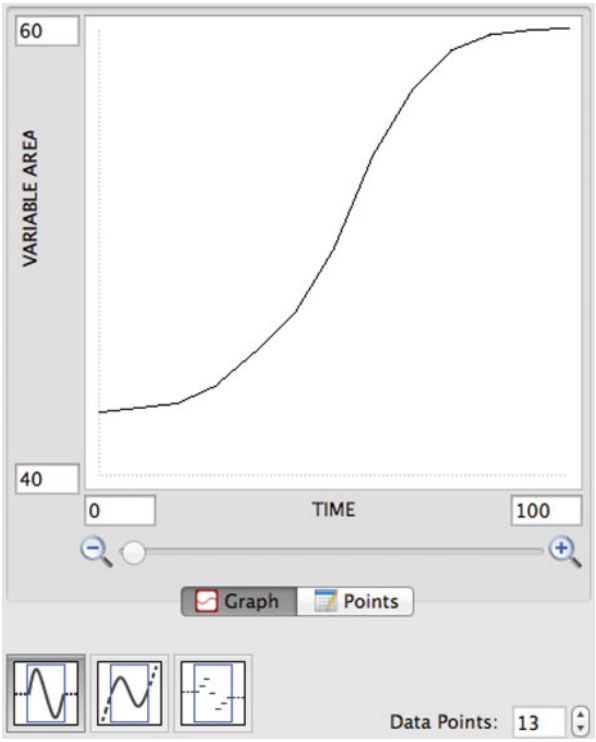

Fig. 22.6

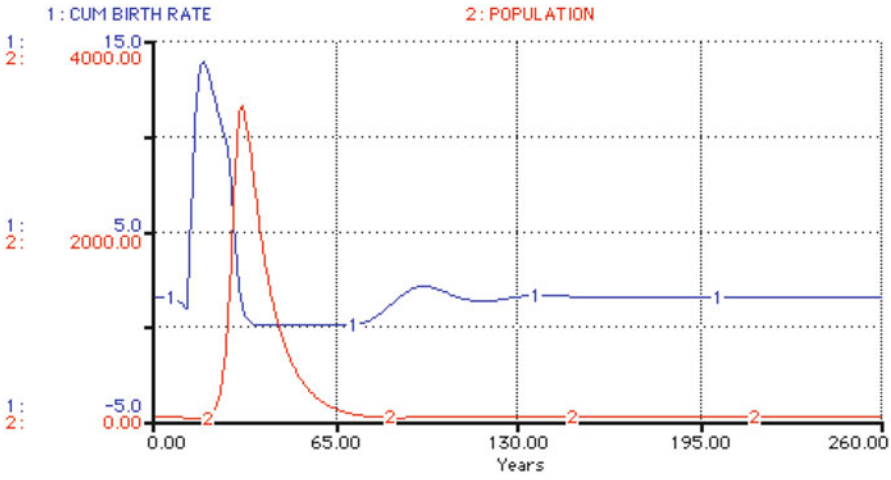

Fig. 22.7

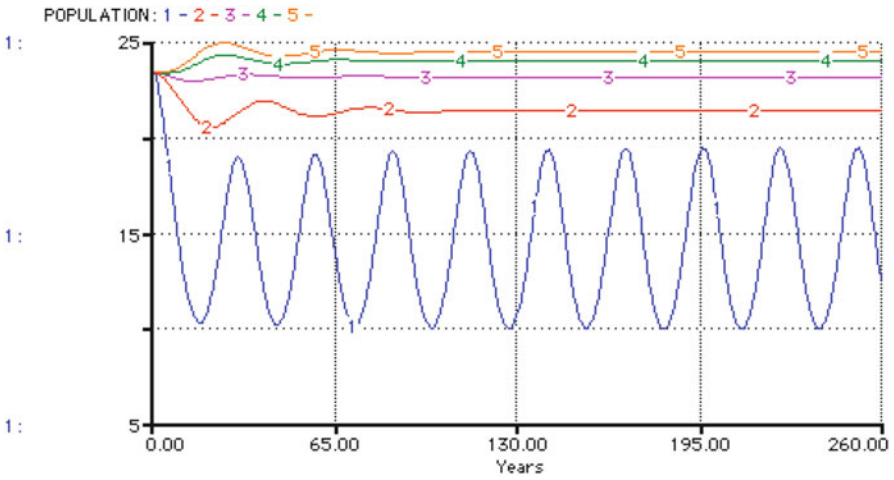

Fig. 22.8

22.2 Adaptive Population Control Model Equations

POPULATION(t) = POPULATION(t − dt) + (NET_BIRTHS) ∗ dt
INIT POPULATION = 23.49
INFLOWS:
NET_BIRTHS = (CUM_BIRTH_RATE/YEARS_RECALL-DEATH_RATE) ∗POPULATION
CUM_BIRTH_RATE(t) = CUM_BIRTH_RATE(t − dt) + (FRACTIONAL_BR − OUT − FORGETTING) ∗ dt
INIT CUM_BIRTH_RATE = 1.5
TRANSIT TIME = YEARS_RECALL
CAPACITY = INF
INFLOW LIMIT = INF
INFLOWS:
FRACTIONAL_BR = REACTION_SENSITIVITY*GAP/POPULATION
OUTFLOWS:
OUT = CONVEYOR OUTFLOW
FORGETTING = LEAKAGE OUTFLOW
LEAKAGE FRACTION = FORGETTING_RATE
LEAK ZONE = 0% to 100%
A = 10
DEATH_RATE = GRAPH(POP_DENSITY)
(0.00, 0.1), (1.00, 0.105), (2.00, 0.125), (3.00, 0.15), (4.00, 0.195), (5.00, 0.245), (6.00, 0.315), (7.00, 0.425), (8.00, 0.58), (9.00, 0.75), (10.0, 1.00)
DESIRED_POP = GRAPH(POP_DENSITY)

(0.00, 99.5), (0.167, 96.5), (0.333, 93.5), (0.5, 90.0), (0.667, 86.5), (0.833, 82.0), (1, 77.5), (1.17, 70.5), (1.33, 61.0), (1.50, 50.0), (1.67, 37.0), (1.83, 21.0), (2.00, 0.00)
FIXED_AREA = 50
FORGETTING_RATE = 0.05
DOCUMENT: Acts like a damping coefficient.
GAP = DESIRED_POP-DELAY(POPULATION,2)
POP_DENSITY = POPULATION/(VA_effect*0+.0001+FIXED_AREA)
REACTION_SENSITIVITY = 0.05
DOCUMENT: The inverse of this number is like a damping coefficient.
VARIABLE_AREA = GRAPH(TIME)
(0.00, 42.8), (8.33, 43.0), (16.7, 43.2), (25.0, 44.0), (33.3, 45.5), (41.7, 47.2), (50.0, 50.1), (58.3, 54.3), (66.7, 57.2), (75.0, 59.0), (83.3, 59.7), (91.7, 59.9), (100, 60.0)
VA_effect = VA_mult*VARIABLE_AREA
VA_mult = 1−time/A
YEARS_RECALL = 15

Chapter 23
Roan Herds

We descended into a valley, bent upon the destruction of a roan antelope.

(W. C. Harris, 1839)

23.1 Roan Herd Model

In the previous chapter we have captured environmental influences on population dynamics through the concept of a carrying capacity, captured through the influence on death rates from changes in available land and associated population density. As parameters of the physical environment change, so does the carrying capacity of the ecosystem. Let us model environmental influences in more detail and focus on seasonal fluctuations and spatial differences in environmental parameters. We develop the model of this chapter for roans.

The roan is an antelope-like animal in Africa. Our problem is to model its population cycles given a weather-grass availability pattern (called KP and KM) and two different habitats called "prime" and "marginal" ground. The converter KP controls the birth and death rate of any roan that live on the prime ground; the converter KM controls the same for those roans living on the marginal ground. These converters are shown in Figs. 23.1 and 23.2.

For these converters, the variable YEAR is defined by

$$\text{YEAR} = \text{INT}(\text{MOD}(\text{TIME}, 12)) + 1 \tag{23.1}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_23,

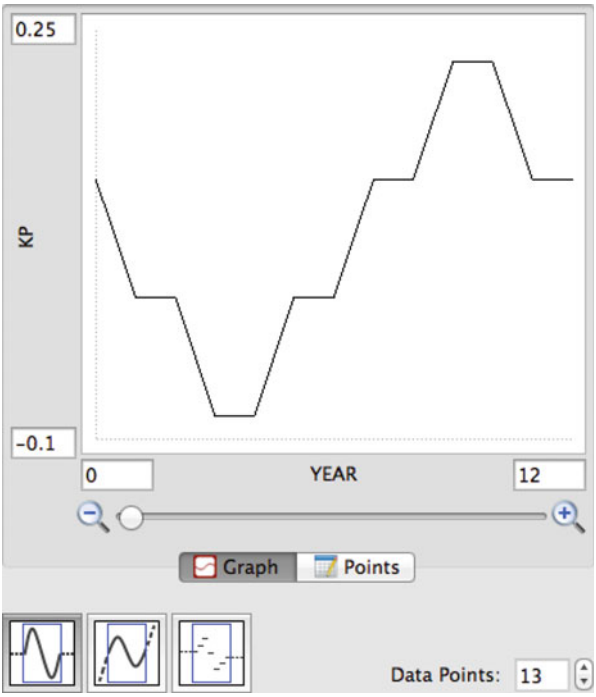

Fig. 23.1

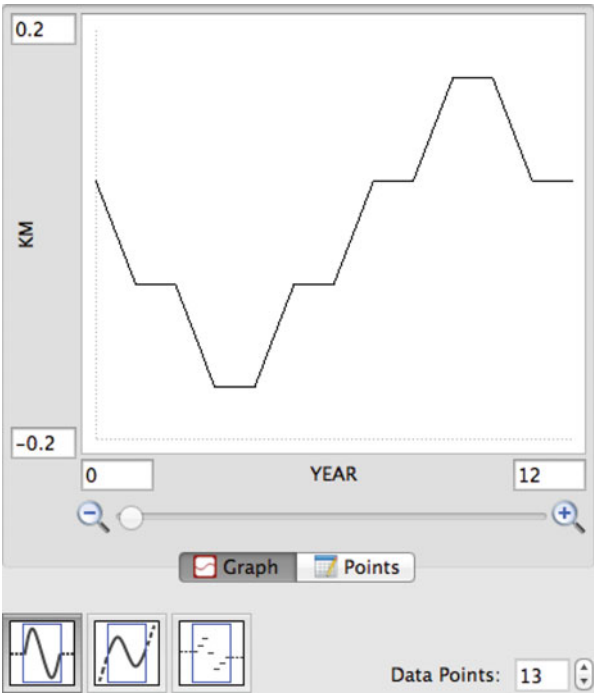

Fig. 23.2

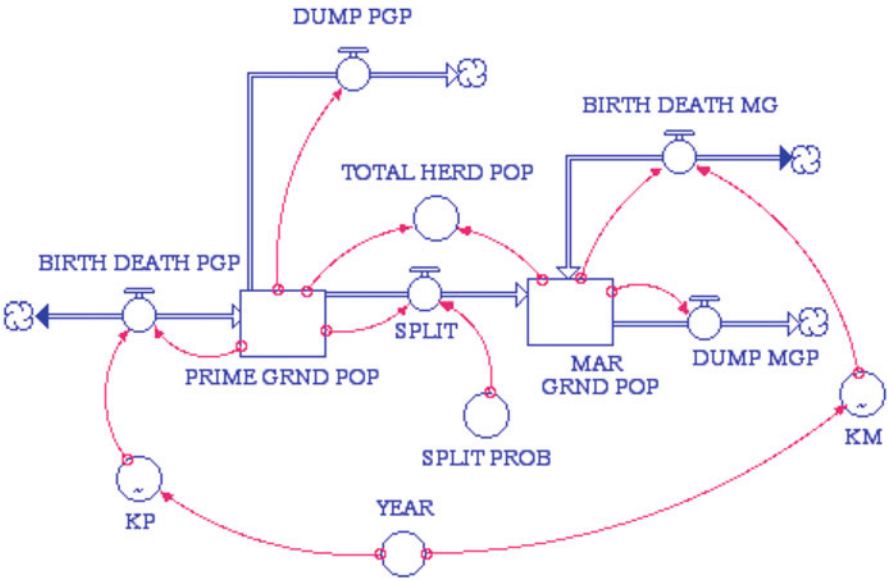

Fig. 23.3

which utilizes the built-in function INT which returns the largest integer less than or equal to its argument. Thus, YEAR counts the numbers 1, 2, . . ., 12 corresponding to the year of the 12-year rainfall cycle. KP and KM reflect that rainfall pattern on births.

Ideally, one should imagine a spatial pattern of prime and marginal grounds and provide an initial population to each. This is the subject of further model building. For now let us say that there are only two places in the model, prime ground and marginal ground. If the population in either place drops below 3, the herd is lost from that area, i.e., the outflows DUMP PGP and DUMP MGP in Fig. 23.3 empty the population stocks on the respective ground type.

The herd on the prime ground (PRIME GRND POP) has a tendency to split to the marginal ground on a random basis. The probability that a split occurs is

$$\text{SPLIT PROB} = \text{INT}(2 * \text{RANDOM}(0, 1)), \tag{23.2}$$

making the SPLIT flow either zero or one according to the following rule:

$$\begin{aligned} \text{SPLIT} = {} & \text{IF PRIME GRND POP} > 7 \text{ THEN} \\ & \text{INT}(\text{PRIME GRND POP}/3) * \text{SPLIT PROB ELSE } 0 \end{aligned} \tag{23.3}$$

The nature of the split function is taken from Starfield and Blelock [1]. These authors define the problem in much more detail than we have done here. In fact, they define a simplified approach and our model is even simpler. Yet, the results of

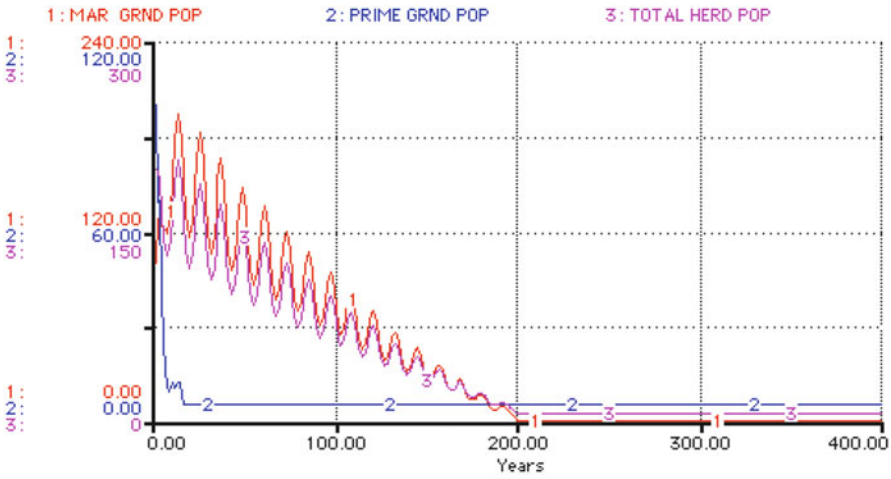

Fig. 23.4

this simplified approach seem to agree with those in the more elaborate model of Starfield and Blelock, reconfirming our earlier statements that you should start with simple models. These are often the most powerful ones.

As Fig. 23.4 shows, the population cycles in the first few decades and then proceeds to approach to a steady-state population of 5 on the prime ground and zero on the marginal ground. However, we should not expect to accurately model anything more than a couple of weather cycles at most. In the first 25 years, the cycle seems normal enough even though the long-term effect is quite different.

Population cycles that are more pronounced than the ones found in this model are typical for animals that produce more rapidly than roans do. Voles and lemming are two prominent examples, and we will model their population dynamics in the following two chapters.

Try adding another marginal land unit. Double the initial prime ground herd size. The newer marginal unit, call it MARGINAL 2, receives "splits" from the prime ground with half of the split probability of the first unit. The marginal units are connected and can transfer roan back and forth with the following rule: If one of the marginal units is larger by 3 or more roan than the other for more than 1 year, then that unit transfers roan to the less populated unit in groups of 3. Run your model for 24 years and interpret your results.

23.2 Roan Herd Model Equations

MAR__GRND_POP(t) = MAR__GRND_POP(t − dt) + (SPLIT + BIRTH_ DEATH_MG − DUMP_MGP) * dt
INIT MAR__GRND_POP = 100 {Individuals}

INFLOWS:
SPLIT = IF PRIME_GRND_POP > 7 THEN INT(PRIME_GRND_POP/3*SPLIT_PROB) ELSE 0 {Individuals per Year}
BIRTH_DEATH_MG = INT(KM*MAR__GRND_POP) {Individuals per Year}
OUTFLOWS:
DUMP_MGP = IF MAR__GRND_POP < 3 THEN INT(MAR__GRND_POP) ELSE 0 {Individuals per Year}
PRIME_GRND_POP(t) = PRIME_GRND_POP(t − dt) + (BIRTH_DEATH_PGP − SPLIT − DUMP_PGP) * dt
INIT PRIME_GRND_POP = 100 {Individuals}
INFLOWS:
BIRTH_DEATH_PGP = IF PRIME_GRND_POP >= 7 THEN INT(KP*PRIME_GRND_POP) ELSE 0 {Individuals per Year}
OUTFLOWS:
SPLIT = IF PRIME_GRND_POP > 7 THEN INT(PRIME_GRND_POP/3*SPLIT_PROB) ELSE 0 {Individuals per Year}
DUMP_PGP = IF PRIME_GRND_POP < 3 THEN INT(PRIME_GRND_POP) ELSE 0 {Individuals per Year}
KM = GRAPH(YEAR)
(0.00, 0.05), (1.00, −0.05), (2.00, −0.05), (3.00, −0.15), (4.00, −0.15), (5.00, −0.05), (6.00, −0.05), (7.00, 0.05), (8.00, 0.05), (9.00, 0.15), (10.0, 0.15), (11.0, 0.05), (12.0, 0.05).
KP = GRAPH(YEAR)
(0.00, 0.12), (1.00, 0.02), (2.00, 0.02), (3.00, −0.08), (4.00, −0.08), (5.00, 0.02), (6.00, 0.02), (7.00, 0.12), (8.00, 0.12), (9.00, 0.22), (10.0, 0.22), (11.0, 0.12), (12.0, 0.12).
SPLIT_PROB = INT(2*RANDOM(0,1))
TOTAL_HERD_POP = PRIME_GRND_POP+MAR__GRND_POP {Individuals}
YEAR = INT((time MOD 12)) + 1

Reference

1. Starfield AM, Bleloch AL (1986) Building models for conservation and wildlife. MacMillan, New York (Chapter 2)

Chapter 24
Population Dynamics of Voles

The true Voles. . .number about fifty known species.

(Cassell's Natural History, 1880)

24.1 Basic Model

Microtine rodent populations, such as voles and lemmings, have been the subject of intense interest in population biology for over 50 years. Much of this interest stems from the dramatic fluctuations in density observed in many populations. These fluctuations are often cyclic in nature, with large-scale irruptions occurring every 2–4 years. Voles and other small rodents are also of great economic importance due to their potential as agricultural pests and as vectors of disease. Voles may cause substantial damage to a wide variety of crops and cause severe damage to fruit orchards by girdling trees. Renewed attention is also being given to the population dynamics of small rodents due to their prospective role in outbreaks of Lyme disease and the Hanta virus. Understanding the factors that regulate their population densities is the first step to controlling future outbreaks.

The periodic oscillations these animals experience, often referred to as vole cycles, have generated a tremendous number of experimental studies. Both field and laboratory approaches have been utilized to determine how voles respond to environmental changes. Although no consensus has been reached concerning the causes of vole cycles, it is clear that no single extrinsic factor, such as weather or food availability, can be directly responsible for this phenomenon. Could intrinsic factors be invoked to explain the occurrence of cyclical population changes in voles?

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_24,

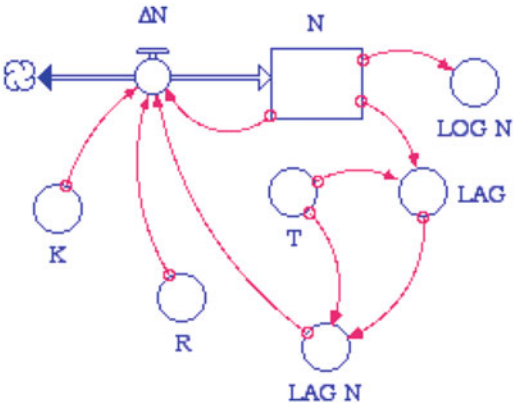

Fig. 24.1

The logistic model of population growth has been previously used as a starting point for simulating vole dynamics, but has failed to generate cycles using parameters that are commonly observed. One attribute that may have been missing from previous models is some form of time lag. It may be that vole populations behave in a logistic fashion, but with a delayed response to changes in density. In this chapter we construct a simple logistic model of population density to determine whether time delayed components can be used to generate multiyear population cycles in voles.

The basic logistic equation for population growth is

$$\Delta N = \frac{dN}{dt} = R * N * \left(1 - \frac{N}{K}\right) \tag{24.1}$$

where R is the intrinsic rate of increase; K is the environmental carrying capacity; and N is the population size or, in the case of a fixed area, the population density. In real populations of animals there may often be a delay between a change in total population size and the animals' response to that change. For example, there may be an increase in birth rates due to increasing food availability, but if reproduction is limited by competition the new animals entering the population may not create any appreciable impact until they reach adult size several months later. This may mean that current density is dependent upon the density at some time period in the past, a phenomenon known as delayed density dependence. Delayed density dependence can be incorporated into our logistic equation as

$$\Delta N = \frac{dN}{dt} = R * N * \left(1 - \frac{N_T}{K}\right) \tag{24.2}$$

where N_T represents the population size at some earlier time period T.

Our basic vole population model is shown in Fig. 24.1.

Using the standard logistic growth equation, the incremental change in population size, ΔN would decrease as the population approaches the environmental carrying capacity, K. You can verify this by setting the time lag, T, equal to zero.

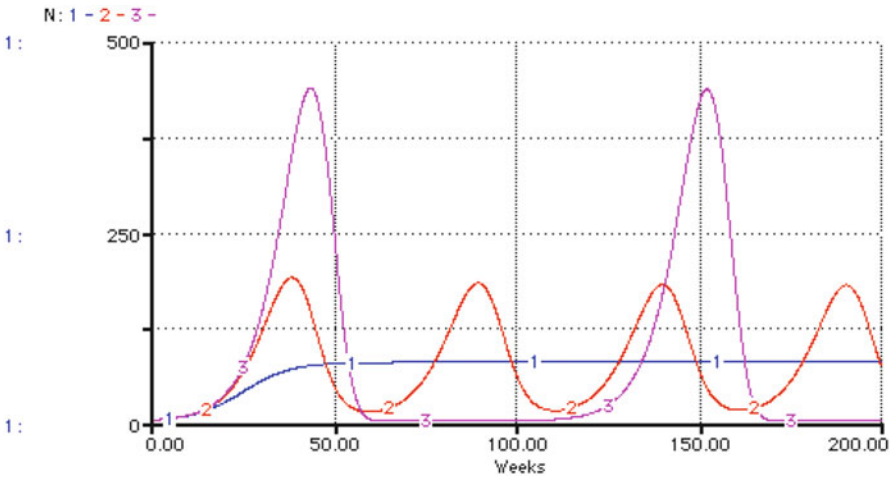

Fig. 24.2

When a nonzero time lag is added however, the population may no longer reach a steady state. Run the model using $K = 78$, $R = .15$, and an initial population of 2. Rerun your model with $T = 12$ and $T = 18$. The values for R and K were measured from a cycling population of meadow voles (*Microtus pennsylvanicus*) in Massachusetts [1]. $N = 2$ represents a founding pair of voles in a new population, and $T = 12$, $T = 18$ represent time lags of approximately one and one and a half generation, respectively. You should find that at $T = 12$ the population oscillates around the carrying capacity, with a period of approximately 1 year. Increases in the time lag lead to oscillations of higher amplitude and lower periodicity. Figure 24.2 summarizes the results for $T = 0$, 12, and 18, respectively.

Now experiment with each of the parameters to determine their potential effects on the population. Predict the qualitative changes you would expect before running the model with modified parameters. The carrying capacity K only raises or lowers the amplitude of the cycle without altering the periodicity. Run the simulation with different initial population sizes, N. Any $N \leq K$ alters the "starting point" of the population without affecting the shape of the oscillation itself. For N much larger than K extreme density fluctuations result that eventually settle down into the standard 1-year oscillation. An illustration for $N = 80$ and $K = 28$ is shown in Fig. 24.3.

The model is quite sensitive to modest changes in the population's intrinsic rate of increase, R. A small increase in this value produces tremendous changes in the amplitude of the oscillation and also increases the period of the oscillation. As R increases, the population will "overshoot" the carrying capacity to a larger extent before the damping effects of the carrying capacity pull the population back into a decline. We will retain our $R = 0.15$ as it has been empirically determined from a natural population. K and N have no affect on periodicity, thus if we are to obtain multiyear cycles from our model we must turn our attention to the effects of the time lag.

Experiment with a large range of values to determine the sensitivity of the model to changes in T. Values of $T > 30$ weeks generate very unstable populations that

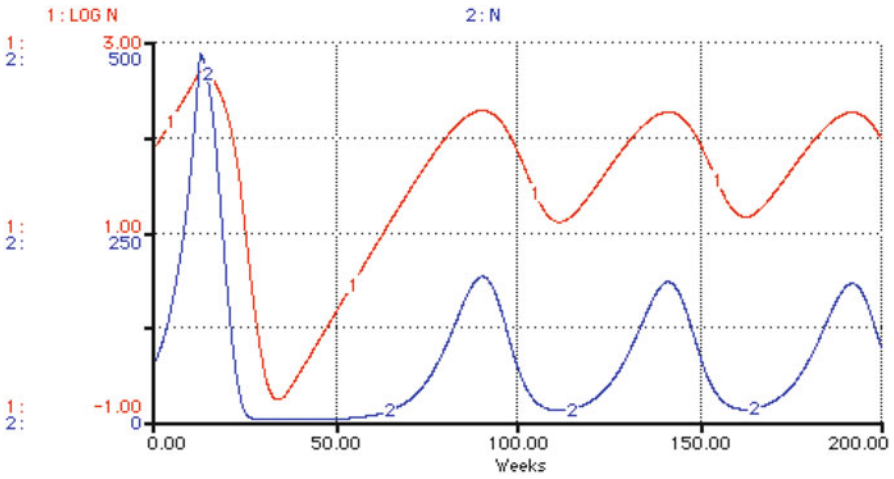

Fig. 24.3

eventually crash to extinction. Very small time lags reduce the period and amplitude of the oscillation; T < 4 weeks produces only temporary cycles that ultimately dampen out on the environmental carrying capacity, K. Conversely, increasing T above 12 increases both the amplitude and period of our cycle.

A time lag of 18 weeks yields a "correct" population cycle with a period of approximately 2 years. At first glance it appears that we have now achieved our task, simulating a vole cycle from field parameters. A closer look at the oscillation, however, reveals a serious flaw in the model. If we plot the simulation using the logarithm of population density, as we have done above, we see that the density undergoes changes of over four orders of magnitude during the course of a cycle. Such changes are not observed in North American vole populations and are not biologically meaningful. The densities of one animal per 100 ha implied by the graph would certainly lead to extinction! The model must be re-examined to determine whether multiyear cycles can be generated without experiencing density changes of over three orders of magnitude. This is done in the following section.

24.2 Basic Vole Model Equations

N(t) = N(t − dt) + (ΔN) ∗ dt
INIT N = 2 {Individuals}
INFLOWS:
ΔN = R∗N∗(1−(LAG_N/K)) {Individuals per Week}
K = 78 {Individuals}
LAG = DELAY(N, T) {Individuals}
LAG_N = IF (TIME>T) THEN LAG ELSE 0 {Individuals}

LOG_N = LOG10(N) {individuals}
R = .15 {Individuals per Individuals per Week}
T = 12 {weeks}

24.3 Vole Population Dynamics with Seasonality

One shortcoming of our previous model may be its failure to account for seasonal shifts in behavior. Seasonal changes in the demographics of voles have been well documented. Let us add seasonality to our delayed-logistic model by varying R, the intrinsic rate of increase, as a function of time. This modification has a solid foundation in field data as these rodents are seasonally reproductive; breeding is sharply curtailed during the winter months. If we assume that limited winter reproduction is roughly equal to winter mortality, we can allow R to alternate between zero and its maximum value during the breeding season.

We will now modify our model to set R = 0 for 16 weeks of each 52-week period and set R = 0.15 for the remainder of the year. This is done in the converter VAR R of our model (Fig. 24.4), where WEEK is a cyclical clock and defined as

$$\text{WEEK} = (\text{TIME MOD } 52) + 1 \tag{24.3}$$

and

$$\text{VAR R} = \text{IF } (\text{week} >= (52 - \text{WINTER}) \text{ and week} <= 52) \text{ THEN 0 ELSE R.} \tag{24.4}$$

Since WEEK resets itself to 1 after an entire year has elapsed, the parameter VAR R is set to zero during the 16 weeks of winter, then reverts back to its original

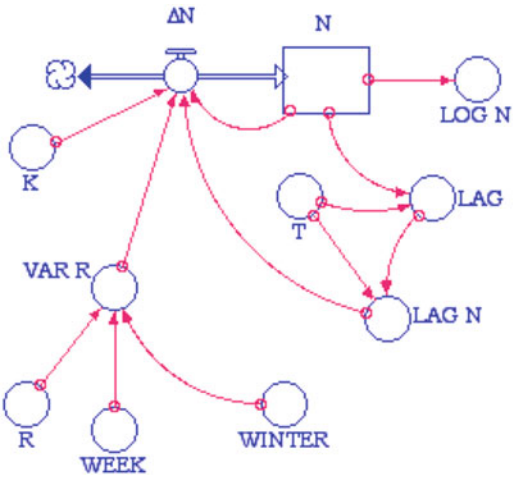

Fig. 24.4

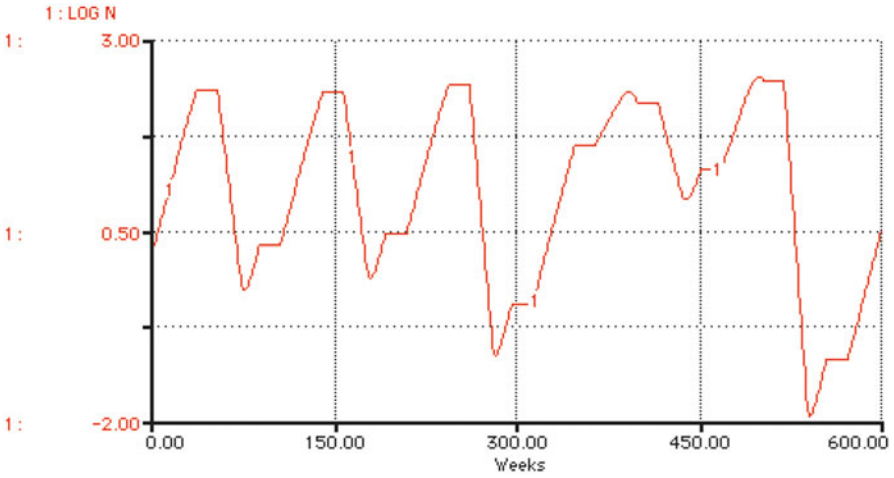

Fig. 24.5

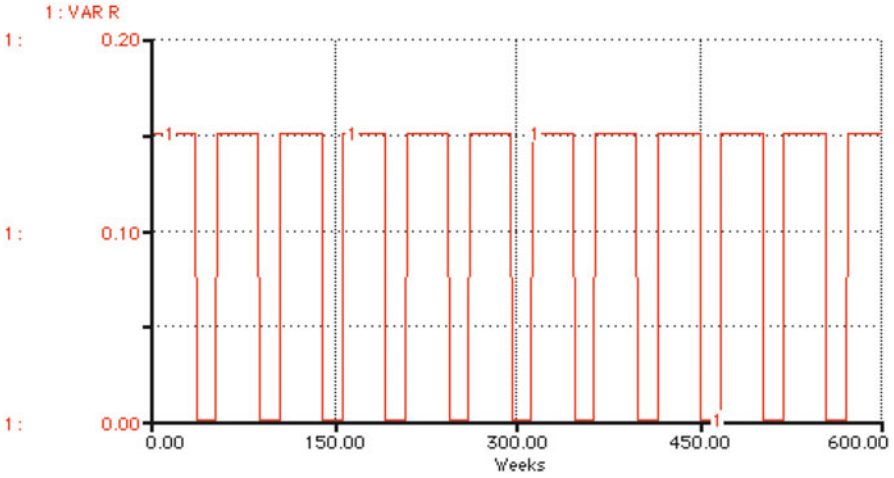

Fig. 24.6

value for the rest of the year. Run the model and observe the results on a logarithmic density plot, using a value of T = 18 weeks. The results are shown in Fig. 24.5.

Initially the population cycles over a period of approximately 2 years, within a fairly reasonable amplitude. After several years elapse however, the amplitude fluctuations become more irregular and we encounter a familiar problem: the range of fluctuations in population density in our model is unreasonably large.

Another shortcoming of this model may be its boolean-style approach to seasonality. A graph of VAR R reveals an "on-off" or boolean mode in the intrinsic rate of increase: the value is reset immediately from one value to another without any transition (Fig. 24.6). It would be more realistic to alter the seasonal component of R in a more continuous fashion. This type of change is modeled in the following section.

24.4 Vole Model with Seasonality Equations

N(t) = N(t − dt) + (ΔN) ∗ dt
INIT N = 2 {Individuals}
INFLOWS:
ΔN = VAR_R∗N∗(1−(LAG_N/K)) {Individuals per Week}
K = 78 {Individuals}
LAG = DELAY(N,T) {Individuals}
LAG_N = IF (TIME>T) THEN LAG ELSE 0 {Individuals}
LOG_N = LOG10(N)
R = 0.15 {Individuals per Individuals per Week}
T = 18 {Weeks}
VAR_R = IF (week >= 52 − WINTER) THEN 0 ELSE R {Individuals per Individual per Week}
WEEK = (TIME MOD 52)+1
WINTER = 16 {weeks}

24.5 Sinusoidal Seasonal Change

Allowing the seasonal component of R to change in a continuous fashion is a better approximation of reality as the population is not perfectly synchronous in its behavioral response to seasonal changes. This form of seasonality may be achieved in our model by allowing the intrinsic rate of growth to oscillate between zero and R in a sinusoidal fashion. In our sinusoidal model, a new variable, SIN R, is simply a sine wave function with a 52-week period oscillation and with an amplitude of −0.75 to +0.75. By setting

$$\text{VAR R} = 0.075 + \text{SINWAVE}(0.075, 52) \tag{24.5}$$

We create an R value varying continuously between 0.15 and 0 over the course of 1 year. Simply replace this new specification of VAR R in the model of the previous section (Fig. 24.4) and you should receive quite satisfying results: multiyear cycles within biologically reasonable amplitudes of density (Fig. 24.7).

What implications do these simulations have for our understanding of vole biology? First, they demonstrate that a simple delayed-logistic equation, with a single seasonal component, is capable of generating multiyear fluctuations in population density. This was accomplished without resorting to external factors such as predation or climate change. These simulations were completed using actual field data for the intrinsic rate of increase (R) and the environmental carrying capacity (K). Second, the lag component was a mere 18 weeks, as opposed to the considerably longer lags used in previous models (e.g. [2]). The 18-week delay used in this simulation roughly approximates the generation time observed in many species of vole. This implies that roughly one generation elapses between any

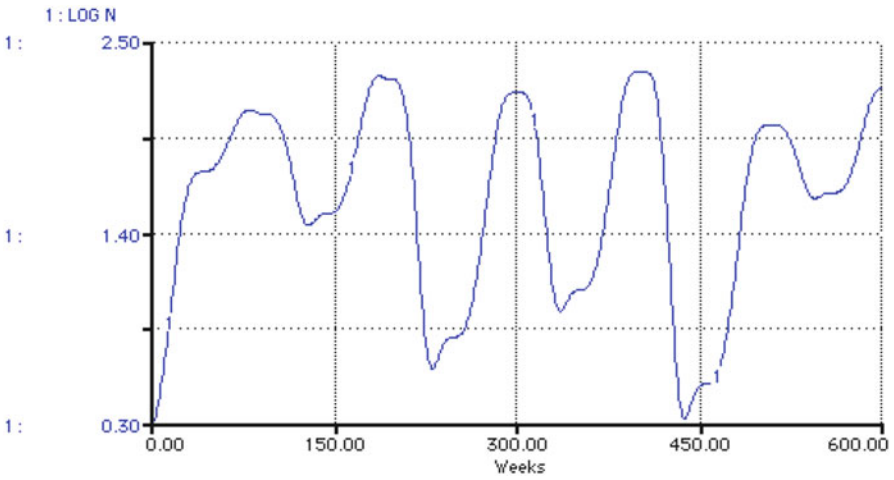

Fig. 24.7

change in density and a corresponding change in population growth rate. In our simple model the time lag was of paramount importance in generating population cycles. Interestingly, regular periodic oscillations in density do not occur in populations where dispersal is prevented, such as on island populations or populations that are enclosed within vole-tight fences. In light of our model results this suggests that blocking dispersal may influence the time-delayed component of vole cycles. A greater understanding of the relationship between dispersal and its potential effects on time-delayed responses may shed new light on the field of microtine population dynamics. Third, modeling of population dynamics, combined with data from field studies, can provide insight into the mechanisms of population change that not only enhance our understanding of the driving forces of the system but further sharpen the focus of subsequent studies of these populations.

24.6 Sinusoidal Vole Model Equations

N(t) = N(t − dt) + (ΔN) ∗ dt
INIT N = 2 {Individuals}
INFLOWS:
ΔN = VAR_R∗N∗(1−(LAG_N/K)) {Individuals per Week}
K = 78 {Individuals}
LAG = DELAY(N, T) {Individuals}
LAG_N = IF (TIME>T) THEN LAG ELSE 0 {Individuals}
LOG_N = LOG10(N)
T = 18 {Weeks}
VAR_R = SINWAVE(.075,52) + .075 {Individuals per Individual per Week}

References

1. Tamarin RH (1977) Demography of the beach vole (*Microtus breweri*) and the meadow vole (*Microtus pennsylvanicus*) in Southern Massachusetts. Ecology 58(6):1310–1321
2. May RM (ed) (1981) Theoretical ecology, 2nd edn. Blackwell Scientific Publishers, Oxford, pp 5–29

Chapter 25
Lemming Population Dynamics

A kind of Mice, (they call Leming. . .) in Norway, which eat up every green thing. They come in such prodigious Numbers, that they fancy them to fall from the Clouds.

(Derham, 1713)

25.1 Lemming Model

Similar to the vole population modeled in the previous chapter, lemming populations can experience significant population fluctuations from year to year. Many legends surround the seemingly erratic population dynamics of lemmings [1].

To model lemming population dynamics it seems advisable [2] to distinguish two types of lemmings: Type 1 reproduces rapidly and migrates in response to overcrowding. Type 2 has a lower reproductive capacity but is less sensitive to high population densities. The change in the densities of each type is given, respectively, by

$$\Delta N1 = \text{IF } N1 > 0 \text{ THEN } N1 * (A1 - (B1 - C1) * N2 - C1 * (N1 + N2)) \text{ ELSE } 0 \tag{25.1}$$

$$\Delta N2 = \text{IF } N2 > 0 \text{ THEN } N2 * (-A2 + B2 * N1) \text{ ELSE } 0 \tag{25.2}$$

The parameters A1, A2, B1, B2, and C1 capture the density-dependence of birth and deaths rates of the two types of lemmings. Figure 25.1 shows the model.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_25,

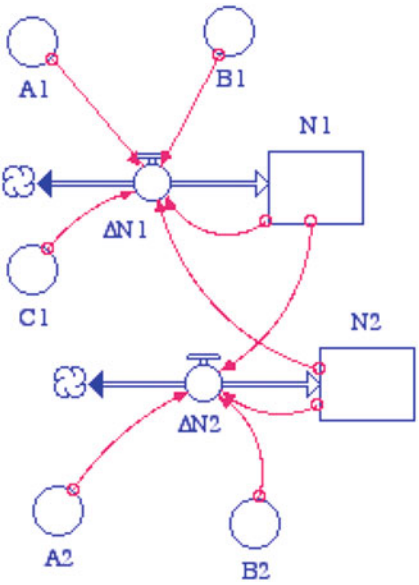

Fig. 25.1

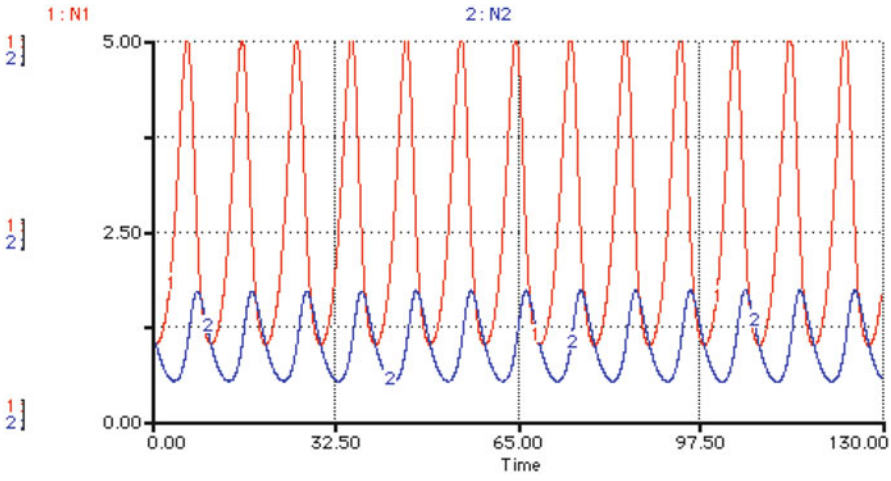

Fig. 25.2

Appropriate choice of these parameters yield oscillatory behavior noted in the lemming populations. For the results in Figs. 25.2 and 25.3, we have set A1 = 0.9 (1/time period), A2 = 0.5 (1/time period), B1 = 0.9 (1/area/time period), B2 = 0.2 (1/individuals/area/time period), and C1 = 0.0043 (1/individuals/area/time period).

As in the previous chapter, introduce seasonality with random variations into the model. Then find the parameters that yield oscillation of the lemming population.

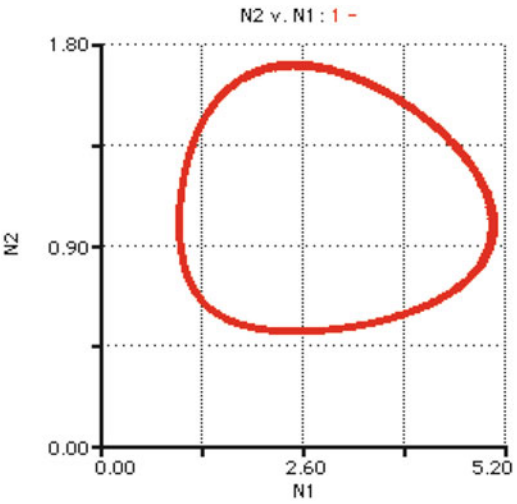

Fig. 25.3

25.2 Lemming Model Equations

N1(t) = N1(t − dt) + (ΔN1) ∗ dt
INIT N1 = 1 {Individuals per Unit Area}
INFLOWS:
ΔN1 = IF N1>0 THEN N1∗(A1-(B1-C1)*N2-C1*(N1+N2)) ELSE 0 {Individuals per Unit Area per Time Period}
N2(t) = N2(t − dt) + (ΔN2) ∗ dt
INIT N2 = 1 {Individuals per Unit Area}
INFLOWS:
ΔN2 = IF N2>0 THEN N2∗(−A2+B2∗N1) ELSE 0 {Individuals per Unit Area per Time Period}
A1 = .9 {1/Time Period}
A2 = .5 {1/Time Period}
B1 = .9 {1/Individuals per Unit Area per Time Period}
B2 = .2 {1/Individuals per Unit Area per Time Period}
C1 = .0043 {1/Individuals per Unit Area per Time Period}

References

1. Decker H (1975) A simple mathematical model of rodent population cycles. J Math Biol 2(1):57–67
2. Myers HJH, Krebs CJ (1971) Genetic, behavioral and reproductive attributes of dispersing field voles *Microtus pennsylvanicus* and *Microtus ochragaster*. Ecol Monogr 41(1):53–78

Chapter 26
Multi-Stage Insect Models

We may define insects to be little animals without red blood, bones or cartilages, furnished with a trunk or else a mouth, opening lengthwise, with eyes which they are incapable of covering, and with lungs which have their openings in the sides.

(Goldsmith, 1774)

26.1 Matching Experiments and Models of Insect Life Cycles

One of the most advanced areas of dynamic modeling at the organism level is found in entomology. Insects have received much attention in part because they are animals of great economic significance: they cause billions of dollars of damage to food supplies around the world every year. We try to control their population levels, having long ago realized they multiply and evolve too fast for elimination.

To better understand the dynamics of insect populations, we model the life cycle of an insect, simplified into two stages, egg and adult. Typically, the data used in understanding insect population dynamics come from laboratory experiments in which one watches each egg and notes when it dies or hatches. Data from such an experiment (at constant temperature) might look like those shown in Table 26.1 of the life history of 100 new insect eggs. Note how the final number of survivors must equal the total number hatched. ESF is the experimental survival fraction and T is the mean maturation time or the mean time to hatch, in days.

Time is measured in days in this case, with data displayed for the beginning of the next day (the result of the previous day). This table yields two important averaged numbers, the experimental survival fraction, ESF (0.699, say 0.7), and

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_26,

Table 26.1 Sample life history table from an experiment

	Time	Died	Survived	Hatched	Time∗Survived	Time∗Hatched
	0	1	99	0	0	0
	1	3	96	0	96	0
	2	3	93	2	186	4
	3	4	89	3	267	9
	4	5	84	21	336	84
	5	5	79	9	395	45
	6	6	73	7	438	42
	7	6	67	5	469	35
	8	7	60	3	480	24
	9	8	52	2	468	18
Col. sum	45			52	3,135	261
					3,135/45/100 = 0.697 = ESF	261/52 = 5.02 = T

the experimental maturation time, T (5.019, say 5 days). How can we use such data to parameterize a model, when the model time step is dramatically different from this experimentally found maturation time?

We must develop a new concept: the model survival fraction, MSF. In ecological experiments, the instantaneous survival rate cannot be measured. But the survival rate can be measured over some real time period, the maturation time, as we see from Table 26.1. A problem arises when we wish to model the system at a shorter time step than the real one. We need to model at these shorter times because the characteristic time of the system may be shorter than the shortest feasible measurement time of some particular part of the system. So we have the experimental time step (the maturation time) and the model time step (DT) and we must devise a conversion from experiment to model.

That conversion is based on the assumption that the survival fraction is a declining exponential, with ESF(T) and T defining its mean point.

$$\begin{aligned} \mathrm{ESF(t)} = \mathrm{N(t + DT)/N(t)} &= \mathrm{EXP(-m * t)} \\ &= \text{the dimensionless experimental survival} \\ &\quad \text{fraction as a function of time.} \end{aligned} \tag{26.1}$$

We are constrained here to the assumption of exponential decline since both the DEATH and HATCH flows in the model are arranged as exponential decays.

N is the population level. Later, when we attempt to verify the experimental data with our model, we will shut off the birth and hatch rate and observe the (necessarily exponential) decline in egg population due to death. The resulting instantaneous survival fraction is determined with the constant m. Using the experimental data, we solve this equation for −m:

$$-\mathrm{m} = \mathrm{LOGN(ESF(T))/T}. \tag{26.2}$$

The *model* survival fraction (based on our choice of the time step DT) is derived:

$$\mathrm{MSF} = \mathrm{ESF(t + DT)/ESF(t)} = \mathrm{EXP(-m * DT)}. \tag{26.3}$$

When the expression for −m is substituted into Eq. (26.3), we get

$$\mathrm{MSF} = \mathrm{EXP(LOGN(ESF)/T * DT)}, \tag{26.4}$$

which is the basic equation for the model survival fraction. We now have the instantaneous survival fraction and those surviving will mature or hatch at the maturation or HATCH rate

$$\mathrm{HATCHING} = \mathrm{EGGS/T * MSF}, \tag{26.5}$$

that is, the survivors hatch at the rate, EGGS/T. Remember, eggs don't have to hatch or die. They may simply wait. When they do die, they are claimed at the model death rate

$$\mathrm{DYING} = \mathrm{EGGS} * (1 - \mathrm{MSF})/\mathrm{DT}, \tag{26.6}$$

the multiplier fraction being the model death rate per egg.

Such life table data can be used to determine the death rate of the adults, a normal demographic application. If we were to watch 100 new adults we could calculate the experimental adult survival fraction, EASF, and adult survival time (mean length of adult life), TA. Let's say that we found these numbers to be 0.8 and 1.0, respectively. These numbers are using in a parallel way to obtain the equivalent of Eq. (26.6) for ADULTS.

The layout for the egg-adult model is shown in Fig. 26.1, with an EGG LAY RATE of 0.5 (eggs per adult per day) and the model results are shown in Fig. 26.2.

Now turn off the BIRTHING and HATCHING flows and put 100 eggs in EGGS. Run the model to verify that it reproduces the experimental mean maturation rate, T, and the experimental survival fraction at T. This must be so since our modeling process is one of exponential decay for both the DYING and HATCHING flows.

Suppose we are uncertain about the exact egg experimental fraction. We may suspect that using literature data is not good enough, and think that this number is within ±10 %. Next we do an experiment to find this number if the total number of adults in 24 days is within ±10 %. Insert a larval stage into this model with a larval survival fraction of 0.8 in 3 days maturation time. Why doesn't the stock of adults in this model grow as is did in the first version?

Using this model, vary T to find the maximum number of eggs to be left alive for next season after 14 days.

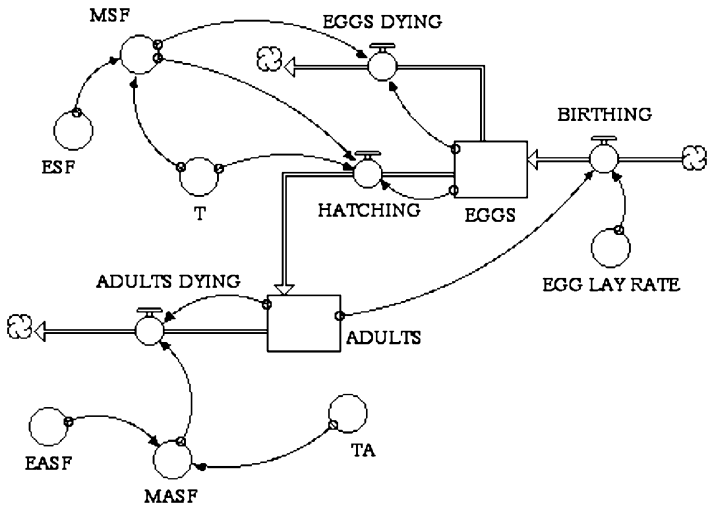

Fig. 26.1

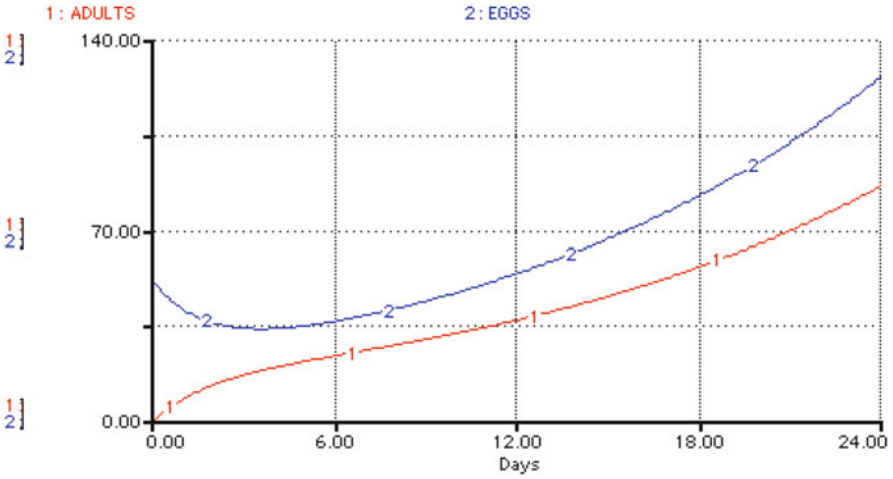

Fig. 26.2

26.2 Two-Stage Insect Model Equations

ADULTS(t) = ADULTS(t − dt) + (HATCHING - ADULTS_DYING) ∗ dt
INIT ADULTS = 0
INFLOWS:
HATCHING = EGGS/T*MSF

OUTFLOWS:
ADULTS_DYING = ADULTS*(1 − MASF)/DT
EGGS(t) = EGGS(t − dt) + (BIRTHING − EGGS_DYING − HATCHING) ∗ dt
INIT EGGS = 50
INFLOWS:
BIRTHING = EGG_LAY_RATE*ADULTS
OUTFLOWS:
EGGS_DYING = EGGS*(1 − MSF)/DT
DOCUMENT: Instantaneous survival fraction + instantaneous mortality fraction = 1.
HATCHING = EGGS/T*MSF
EASF = .8 {Experimental daily adult survival fraction per stage, dimensionless.}
EGG_LAY_RATE = 0.5
DOCUMENT: Experimental laying rate. EGGS PER ADULT PER DAY.
ESF = .7 {Experimental egg survival fraction, dimensionless, per stage. Stage = 1/EXP MATURE RATE, i.e., 70 eggs per 100 eggs survive each 1/EXP MATURE RATE days, as noted in the experiment.}
MASF = EXP(LN(EASF)*DT/TA)
MSF = EXP(LN(ESF)/T*DT)
T = 5
DOCUMENT: Inverse is the Experimental Maturation Rate, 1/DAY.
TA = 1
DOCUMENT: One day = experimental period for which adult mortality is measured.

26.3 Two-Stage Insect Model with a Degree-Day Calculation Controlling the Maturation Rate

Let us take the model of the previous section and specify the maturation rate as a function of the temperature. In this model, time and temperature are now the two independent variables. We have assumed a sine function for the mean daily temperature and assumed ±10 °F + the mean daily to get the high and low temperature for the day:

$$\text{DAILY MEAN TEMP} = 47 + 27 * \text{SIN}\,(2 * \text{PI}/365 * \text{TIME}) \qquad (26.7)$$

If the high temperature is less than the threshold or base temperature, no degree-days for that day are calculated. If the minimum temperature for that day is less than the threshold temperature but the maximum temperature is greater than the threshold, the degree-days, DD 1, are calculated as the maximum temperature minus the threshold temperature, divided by 2:

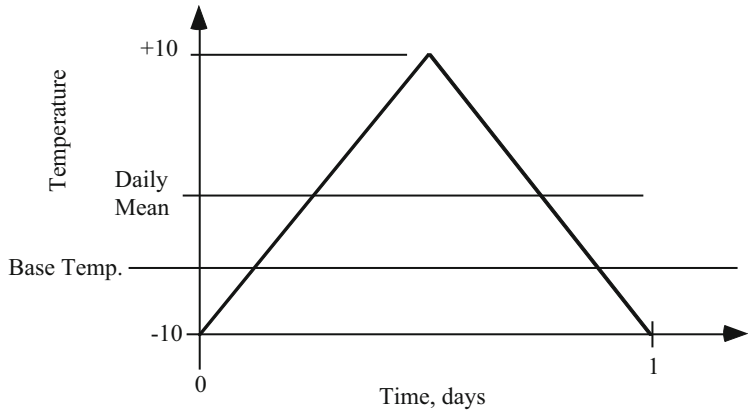

Fig. 26.3

$$\text{DD} = \text{IF DAILY MEAN TEMP} + 10 < \text{BASE TEMP THEN 0 ELSE} \\ (\text{DAILY MEAN TEMP} + 10 - \text{BASE TEMP})/2 \tag{26.8}$$

The calculation is illustrated in Fig. 26.3.

For this model we needed to assume a relation between the degree-days and the maturation time that we assumed to be a declining exponential function. Degree-days (DD) are not accumulated over time but are determined for each time step (1 day). We have also set up a linearly increasing effect of DD on the birth or oviposition rate as shown in Figs. 26.4 and 26.5 in order to prevent the population from growing too large. Figure 26.6 shows the complete model and Fig. 26.7 presents its results.

Together these additions make the population rise and fall. But the peak declines exponentially. Do you know why?

Figure 26.7 shows the results. Note how the adults die off and the egg number levels off when the threshold temperature is reached. Run this model for 2 years. When will it begin to repeat itself and thus be clear of the initial conditions? Try different initial conditions and find the same sort of independence. The results are extremely dependent on the parameter values. Why is this and how could the model be restructured to reduce such dependency (or is it real)? The insects are gradually dying out of this system. What are the logical changes that you could make to stabilize their annual peaks? For example, stabilize the egg numbers by experimenting with the adult maturation time.

This result is extremely sensitive to the Egg Lay Rate. What happens if I change the shape of its graphical relationship with the Degree-days, DD? Try a slightly convex form of the graph.

Assume that this model represents the pattern of a needed predator insect. Try to add in the fewest number of eggs at the best time to keep this species between 30 and 50 adults at the most throughout the indefinite future.

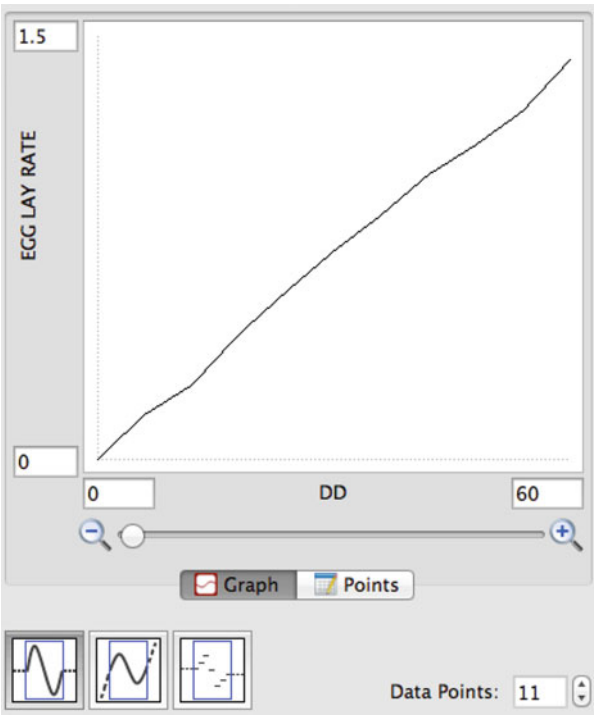

Fig. 26.4

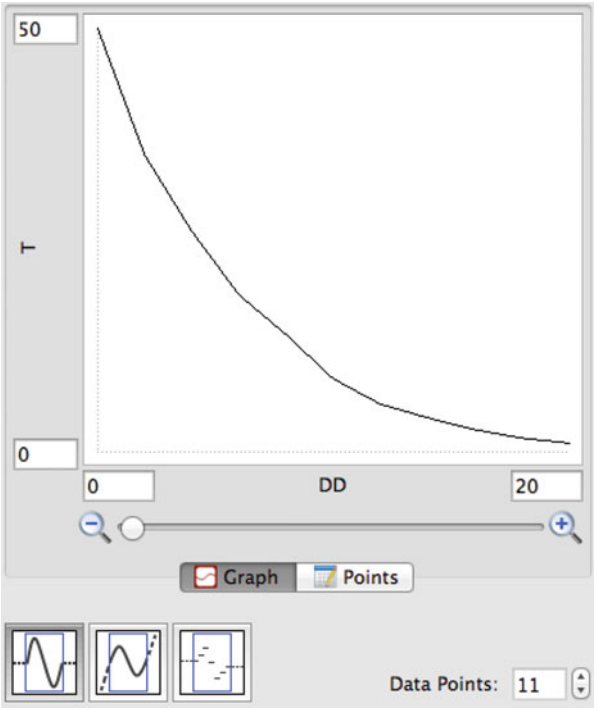

Fig. 26.5

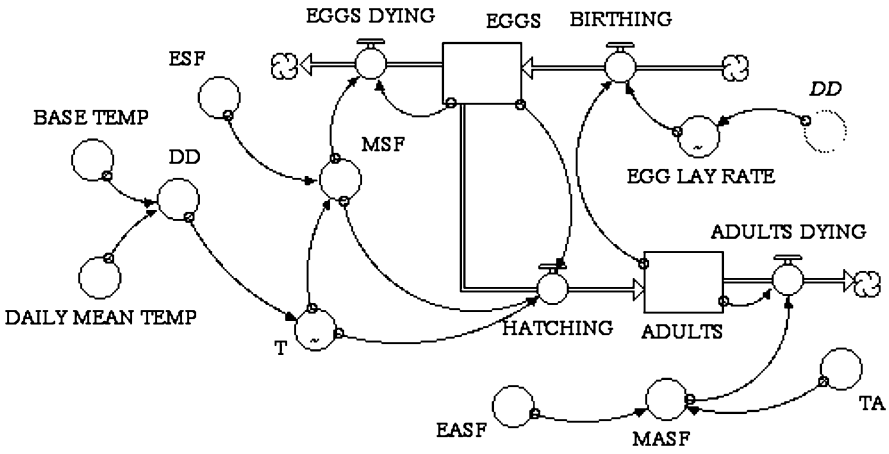

Fig. 26.6

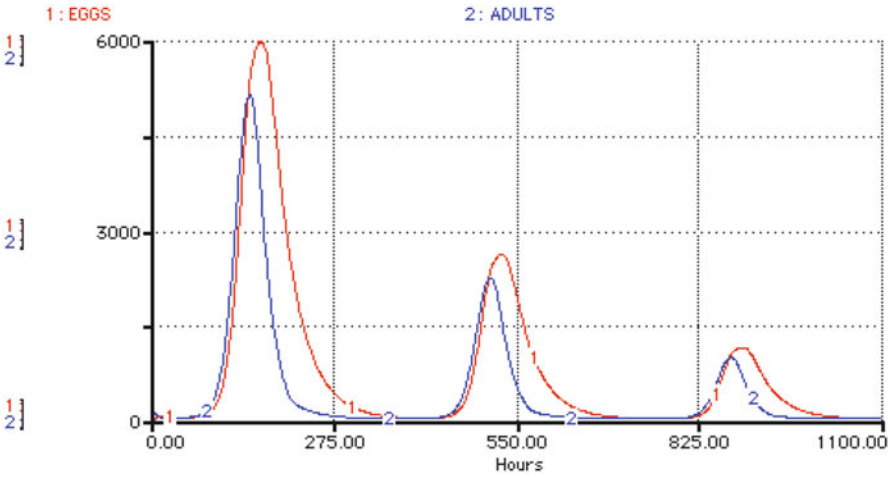

Fig. 26.7

As a modification to the model above, we could assume that the egg maturation rate was a function of time and the *accumulated* degree-days. We would have to represent oviposition by adding in the eggs over a very short period. Can you devise a feasible model of this sort? Use a normal distribution of the daily temperature and assume a standard deviation and then use the base temperature as the basis for calculating the accumulated degree-days. Take the difference between the daily mean as now calculated, add the normal deviation for the day, and subtract the base temperature to determine the number of degree-days for each day. Then have the maturation rate a function of the cumulative DD and time. Find the adult maturation time that stabilizes the egg population. Find the egg maturation time that maximizes the adult population.

26.4 Two-Stage Insect Model with Degree-Day Equations

ADULTS(t) = ADULTS(t − dt) + (HATCHING − ADULTS_DYING) ∗ dt
INIT ADULTS = 100
INFLOWS:
HATCHING = EGGS∗MSF/T
OUTFLOWS:
ADULTS_DYING = ADULTS∗(1 − MASF)/DT
EGGS(t) = EGGS(t − dt) + (BIRTHING - EGGS_DYING − HATCHING) * dt
INIT EGGS = 0
INFLOWS:
BIRTHING = EGG_LAY_RATE∗ADULTS
OUTFLOWS:
EGGS_DYING = EGGS∗(1 − MSF)/DT
HATCHING = EGGS∗MSF/T
BASE_TEMP = 48
DAILY_MEAN_TEMP = 47 + 27∗SIN(2∗PI/365∗TIME)
DD = if DAILY_MEAN_TEMP+10 < BASE_TEMP then 0 else (DAILY_MEAN_TEMP+10-BASE_TEMP)/2
EASF = .8 {The adult survival rate.}
EGG_LAY_RATE = GRAPH(DD)
(0.00, 0.00), (6.00, 0.158), (12.0, 0.263), (18.0, 0.443), (24.0, 0.593), (30.0, 0.735), (36.0, 0.863), (42.0, 1.00), (48.0, 1.11), (54.0, 1.23), (60.0, 1.41)
ESF = .7
MASF = EXP(LN(EASF)/TA∗DT)
MSF = EXP(LN(ESF)/T∗DT)
T = GRAPH(DD)
(0.00, 50.0), (2.00, 35.0), (4.00, 26.0), (6.00, 18.5), (8.00, 13.8), (10.0, 8.60), (12.0, 5.60), (14.0, 4.00), (16.0, 2.60), (18.0, 1.60), (20.0, 1.00)
TA = 1

26.5 Three-Stage Insect Model

In this section we return again to the basic two-stage insect model but introduce now a larval stage. The survival rate for larvae is given by

$$\text{LARV INTANT SURV} = \text{EXP}(\text{LOGN}(S2) * U2 * DT) \tag{26.9}$$

Unlike in the previous models we must now distinguish the experimental egg maturation rate from the experimental larval maturation rate. The resulting model is shown in Fig. 26.8, and its results are depicted in Fig. 26.9.

Perform a sensitivity analysis for the parameters of T and EGG LAY RATE, and interpret your results. Then, extend your model to incorporate the degree-day influence on maturation rates of eggs discussed above as well as a degree-day

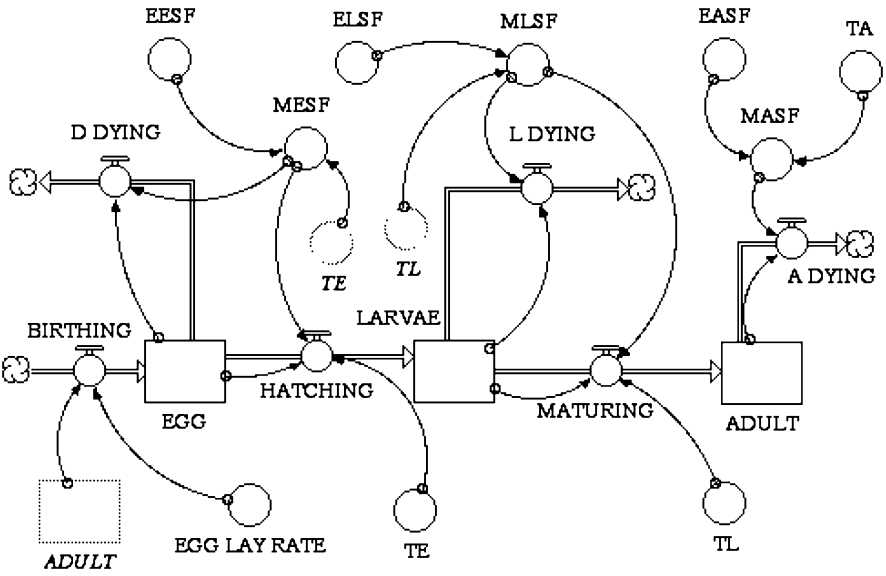

Fig. 26.8

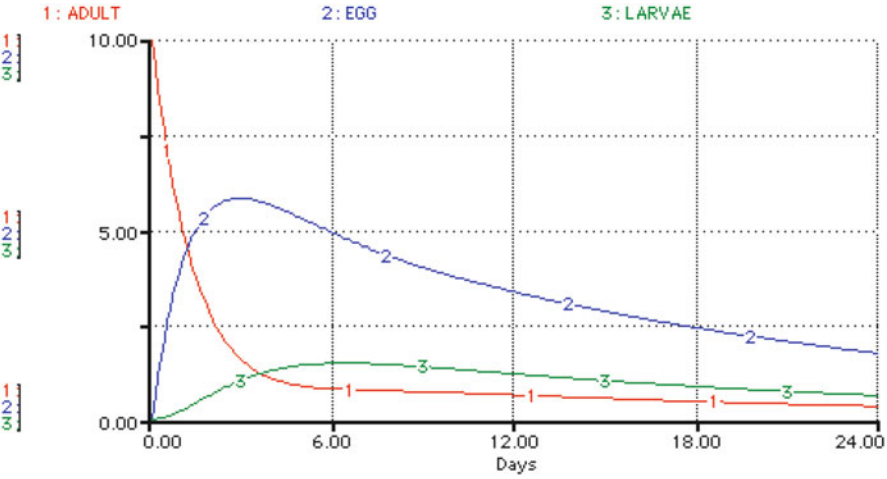

Fig. 26.9

influence on maturation rates of larvae. How does the introduction of the larval stage affect your results?

In the following chapter and Chap. 31 we will combine our insight into the spread of a disease—already modeled in simplified form in Chap. 21—with the knowledge we gained here on insect population dynamics and the ways in which laboratory experiments can be used to set up dynamic models of biological systems.

26.6 Three-Stage Insect Model Equations

ADULT(t) = ADULT(t − dt) + (MATURING − A_DYING) ∗ dt
INIT ADULT = 10 {Number of Adults}
INFLOWS:
MATURING = MLSF*LARVAE/TL {Adults per Time Period}
OUTFLOWS:
A_DYING = (1 − MASF)∗ADULT/DT
EGG(t) = EGG(t − dt) + (BIRTHING − HATCHING − D_DYING) ∗ dt
INIT EGG = 0 {Number of Eggs}
INFLOWS:
BIRTHING = EGG_LAY_RATE∗ADULT {Eggs per Time Period}
OUTFLOWS:
HATCHING = MESF∗EGG/TE {Larvae per Time Period}
D_DYING = (1-MESF)∗EGG/DT {Eggs per Time Period}
LARVAE(t) = LARVAE(t - dt) + (HATCHING − MATURING - L_DYING) ∗ dt
INIT LARVAE = 0 {Number of Larvae}
INFLOWS:
HATCHING = MESF∗EGG/TE {Larvae per Time Period}
OUTFLOWS:
MATURING = MLSF∗LARVAE/TL {Adults per Time Period}
L_DYING = (1 − MLSF)∗LARVAE/DT {Larvae per Time Period}
EASF = .5 {Experimental adult survival per time step.}
EESF = .7 {Experimental egg-larvae survival rate, 1/stage.}
EGG_LAY_RATE = 0.6 {eggs/adult/day. Try 1.112}
ELSF = .8 {Experimental larvae-adult survival rate, 1/stage.}
MASF = EXP(LN(EASF)/TA∗DT)
MESF = EXP(LN(EESF)/TE∗DT)
MLSF = EXP(LN(ELSF)/TL∗DT)
TA = 1
TE = 8
DOCUMENT: DAYS.
TL = 3
DOCUMENT: DAYS

Chapter 27
Two Age-Class Parasites

Men who reject the responsibility of thought and reason can only exist as parasites on the thinking of others.

(Ayn Rand, The Virtue of Selfishness)

27.1 Two Age-Class Parasite Model

This model shows how a disease spreads in an insect population, such as asexually reproducing aphids, consisting of two life stages—nymphs and adults. The model has two parts, one for the healthy population and one for the diseased population. Diseased nymphs can infect healthy nymphs and become diseased adults. Diseased adults cannot infect healthy adults or nymphs but can produce infected nymphs. Note how these features are expressed in the model by the appropriate flows and links.

Unlike in the previous section of this chapter, the infection coefficient is based on an exponential model.

$$\text{INFECTION COEF} = 1 - \text{EXP}(-.3 * \text{NYMPHS} * \text{NYMPHS D}) \qquad (27.1)$$

NYMPHS and NYMPHS D refer to the population sizes of healthy and infected nymph populations, respectively. The INFECTION RATE is calculated as the product of the INFECTION COEF, the number of healthy nymphs, divided by the model maturation rate for survivors, MNSF/TN

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_27,

$$\text{INFECTION RATE} = \text{MNSF/TN} * \text{ INFECTION COEF} * \text{NYMPHS} \quad (27.2)$$

with

$$\begin{aligned} \text{MNSF} &= \text{Model Nymph Survival Fraction} \\ &= \text{EXP}(\text{LOGN}(\text{ENSF})/\text{TN}^* \text{ DT}) \end{aligned} \quad (27.3)$$

where TN is the experimental nymph maturation rate and ENSF is the dimensionless experimental nymph survival fraction. When there are no sick nymphs or healthy nymphs the probability of becoming infected equals zero. The specification of the INFECTION COEF translation variable is a purely empirical formulation but it gives the correct value at the extremes, 0 when the number of diseased nymphs is zero or when the number of healthy nymphs is zero, and near 1 when either at least one of the stocks NYMPHS or NYMPHS D is very large.

Note well the specification of the MATURING function in the model that insures that the total rate of change from nymphs to adults here is still U1*NYMPHS:

$$\text{MATURE} = \text{MNSF/TN}^* \text{ NYMPHS}^* (1 - \text{INFECTION COEF}) \quad (27.4)$$

The complete model is shown in Fig. 27.1.

Figure 27.2 shows the proportions of healthy nymphs and adults, and the number of diseased nymphs and adults. Similar to the previous section of this chapter, we find distinct phases for the outbreak of a disease. Try varying the separate birth and death rates and the infection coefficient and noting the effect on the relative size of the healthy and diseased portions of the populations.

These newly diseased nymphs are converted to diseased adults rather than directly into diseased nymphs in an effort to reflect the fact that these nymphs, who contract rather acquire the disease, have the normal nymph survival rate and, they are not able to convey the disease to other healthy nymphs.

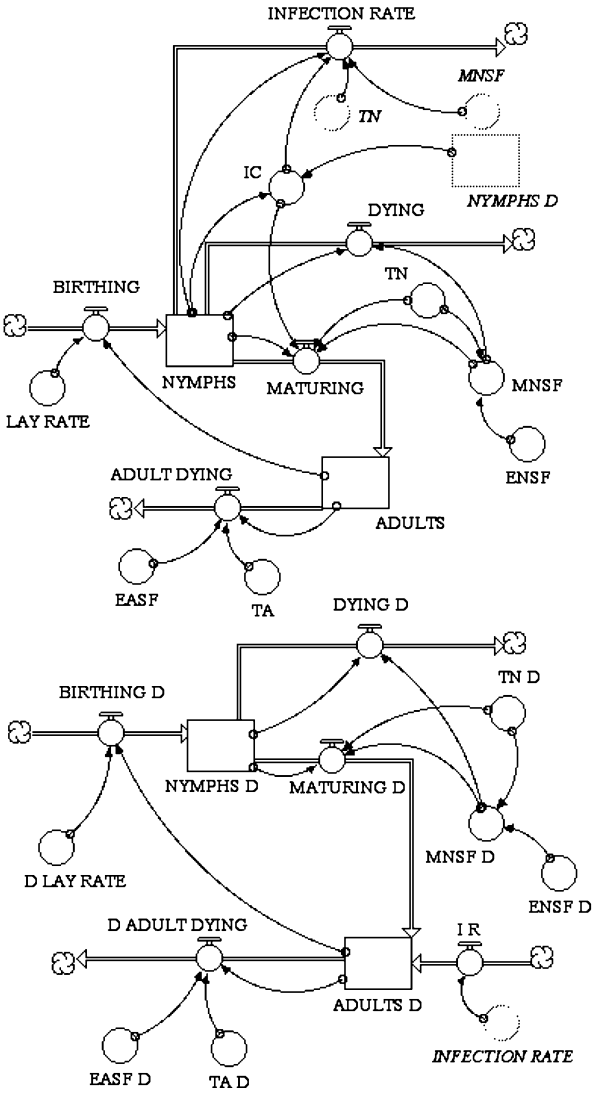
INFECTION RATE
MNSF
TN
IC
NYMPHS D
DYING
TN
BIRTHING
NYMPHS
MATURING
MNSF
LAY RATE
ENSF
ADULT DYING
ADULTS
EASF
TA
DYING D
TN D
BIRTHING D
NYMPHS D
MATURING D
MNSF D
D LAY RATE
ENSF D
D ADULT DYING
I R
ADULTS D
INFECTION RATE
EASF D
TA D

Fig. 27.1

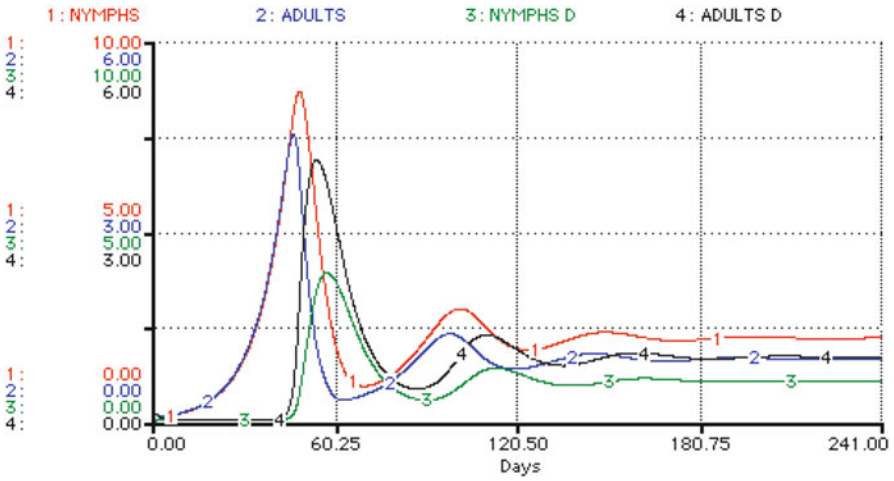

Fig. 27.2

27.2 Two Age-Class Parasite Model Equations

ADULTS(t) = ADULTS(t − dt) + (MATURING − ADULT_DYING) ∗ dt
INIT ADULTS = .10 {Individuals}
INFLOWS:
MATURING = MNSF∗NYMPHS∗(1−IC)/TN {Individuals per Day}
OUTFLOWS:
ADULT_DYING = ADULTS∗(1-EXP(LN(EASF)/TA∗DT))/DT {Individuals per Day}
ADULTS_D(t) = ADULTS_D(t − dt) + (MATURING_D + I_R − D_ADULT_DYING) ∗ dt
INIT ADULTS_D = .10 {Individuals}
INFLOWS:
MATURING_D = MNSF_D∗NYMPHS_D/TN_D
I_R = INFECTION_RATE {Individuals per Day}
OUTFLOWS:
D_ADULT_DYING = ADULTS_D∗(1-EXP(LN(EASF_D)/TA_D∗DT))/DT {Individuals per Day}
NYMPHS(t) = NYMPHS(t − dt) + (BIRTHING − DYING − MATURING − INFECTION_RATE) ∗ dt
INIT NYMPHS = 0 {Individuals}
INFLOWS:
BIRTHING = LAY_RATE∗ADULTS {Individuals per Day}
OUTFLOWS:
DYING = (1 − MNSF)∗NYMPHS/DT
MATURING = MNSF∗NYMPHS∗(1−IC)/TN {Individuals per Day}
INFECTION_RATE = MNSF∗IC∗NYMPHS/TN

NYMPHS_D(t) = NYMPHS_D(t − dt) + (BIRTHING_D − DYING_D − MATURING_D) ∗ dt
INIT NYMPHS_D = 0 {Individuals}
INFLOWS:
BIRTHING_D = D_LAY_RATE∗ADULTS_D
OUTFLOWS:
DYING_D = (1 − MNSF_D)∗NYMPHS_D/DT
MATURING_D = MNSF_D∗NYMPHS_D/TN_D
D_LAY_RATE = .35
EASF = 0.8
EASF_D = .65
ENSF = .7
ENSF_D = .5
IC = 1−EXP(−.3∗NYMPHS∗NYMPHS_D)
DOCUMENT: INFECTION COEFFICIENT
LAY_RATE = .6
MNSF = EXP(LN(ENSF)/TN∗DT)
MNSF_D = EXP(LN(ENSF_D)/TN_D∗DT)
TA = 1
TA_D = 1
TN = 5
TN_D = 5

Chapter 28
Monkey Travels

We are just an advanced breed of monkeys on a minor planet of a very average star. But we can understand the Universe. That makes us something very special.

(Stephen Hawking)

28.1 Model of Monkey Travels

In the rain forests of Peru, a small monkey, a Tamarin, no larger than a squirrel, lives in groups of about a dozen and spends most of its time in the canopy. It eats, travels, and sleeps in the canopy, apparently fearing predators on the ground and in the air above the canopy. The canopy for the most part, is so thick that vision is limited to a few meters at most. They travel on the average about 90 m before finding a sufficient quantity of fruit to stop for a feeding bout. They have about 5–10 such bouts each day, before stopping to sleep in the largest local tree. An interesting question arises as to how these Tamarins find food. One possibility is that they used their noses. This is the central assumption for the model of this chapter, which is based on Garber and Hannon [1].

Let us lay out a horizontal plane, which runs through the canopy of all the taller trees, and is divided into cells of unit width and height. The symbol X measures the distance from the origin in the center of the space and Y measures the distance perpendicular to X. The odor strength is measured vertically, perpendicular to the X–Y plane. The Tamarin troop is considered a point in this X–Y plane and moves in a straight line for the visual sight distance in the canopy. We assumed that this distance, called VISIBILITY in our model, is a constant 2.0 m. At the end of this STEP DISTANCE a new assessment is made and a new direction is chosen.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_28,

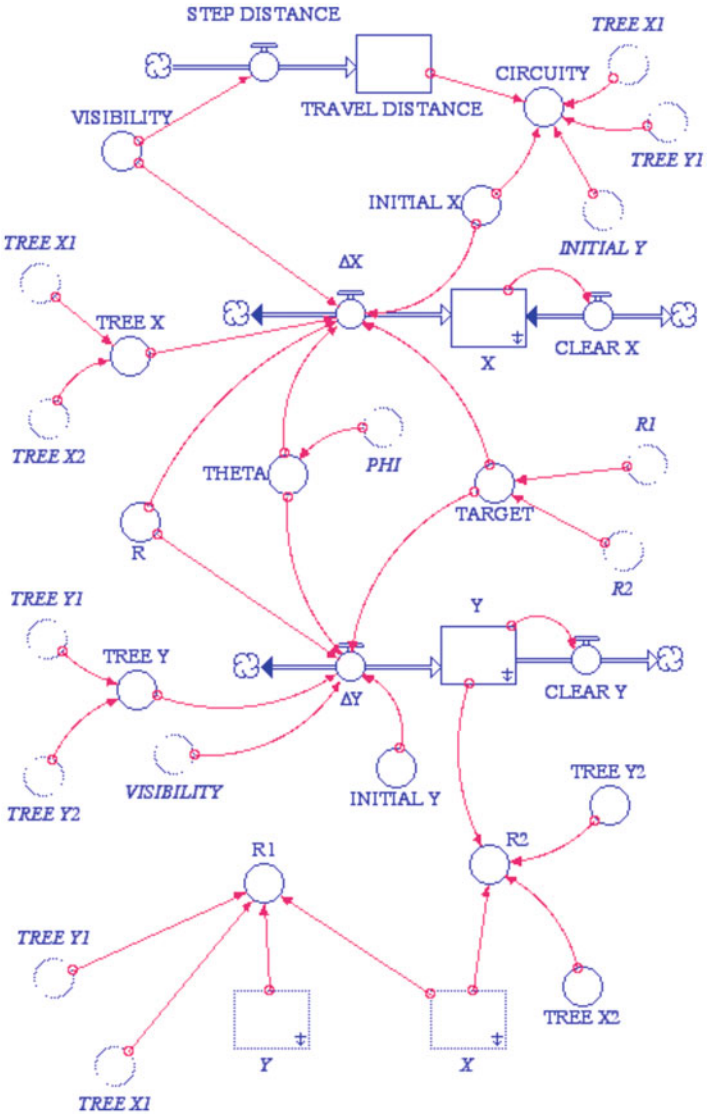

Fig. 28.1

This process is continued until the fruited tree is in sight of the troop. We assume that the typical tree could be seen at a distance of 17 m from the tree. We also assume that the troop cannot travel exactly in the direction it desires to go because of the lack of suitable branches. So the chosen direction is modified in the model by a small random angle.

Figure 28.1 shows that part of the model dealing with the location and movement of the monkeys. Here, the state variables are the coordinates X and Y at which the

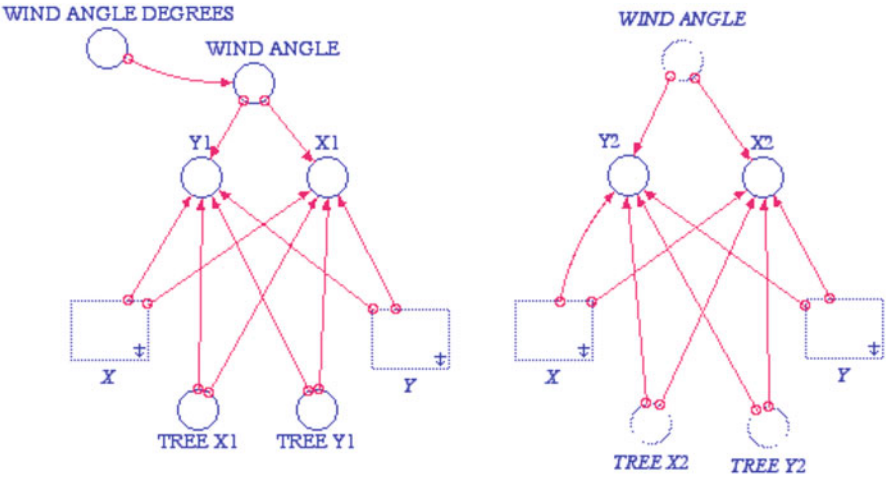

Fig. 28.2

troop is located. For example, the X variable is initialized as a specific position, such as the origin of the grid, and updated by ΔX.

The variables TREE X1, TREE Y1, and TREE X2 TREE Y2 draw the outline of each of the trees present in the plane. VISIBILITY is the specified step distance and it is used, after the trees are drawn, to calculate the total travel distance from the starting point to the sighting point of the tree. The controller TARGET calculates the relation of the troop to the tree circle of a visual influence of 17 m radius. When this circle is reached, it is assumed that the troop can proceed by sight:

$$\text{TARGET} = \text{IF (TIME} > 16) \text{ AND } ((\text{R1} \leq 17) \text{ OR } (\text{R2} \leq 17)\ ) \text{ THEN 0 ELSE 1} \tag{28.1}$$

with R1 and R2 as the straight line distance from the troop to the center of tree 1 and tree 2, respectively. R1 and R2 are defined by

$$\text{R1} = \text{SQRT}((\text{X} - \text{TREE X1})\hat{}\,2 + (\text{Y} - \text{TREE Y1})\hat{}\,2) \tag{28.2}$$

$$\text{R2} = \text{SQRT}((\text{X} - \text{TREE X2})\hat{}\,2 + (\text{Y} - \text{TREE Y2})\hat{}\,2) \tag{28.3}$$

The straight line distances, R1 and R2 from the troop to the center of each tree are used for the calculation of the odor strength at the troop location. This calculation is performed as shown in Fig. 28.2. This part of the model is based on information on the WIND ANGLE DEGREES from which we derive the WIND ANGLE in radians, counter-clockwise from the positive X axis. The WIND ANGLE is then used to calculate the troop location coordinate relative to the tree 1 in terms of the plume.

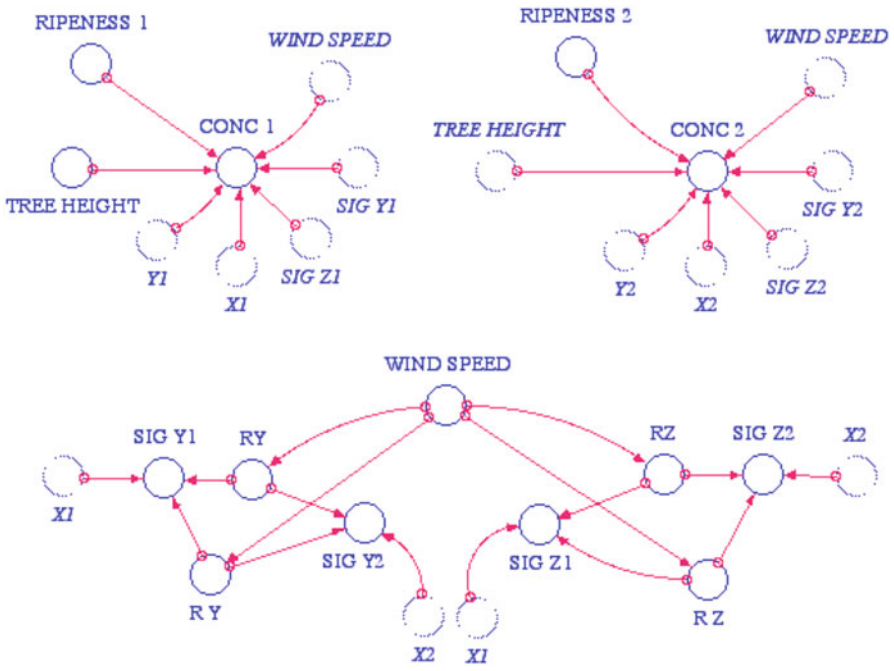

Fig. 28.3

$$\begin{aligned} \text{X1} = &(\text{X} - \text{TREE X1})^{*} \text{ COS (WIND ANGLE)} \\ &+ (\text{Y} - \text{TREE Y1})^{*} \text{ SIN (WIND ANGLE)} \end{aligned} \tag{28.4}$$

$$\begin{aligned} \text{Y1} = &(\text{Y} - \text{TREE Y1})^{*} \text{ COS (WIND ANGLE)} \\ &- (\text{X} - \text{TREE X1})^{*} \text{ SIN (WIND ANGLE)} \end{aligned} \tag{28.5}$$

X1 is measured parallel to wind, Y1 perpendicular to X1.

To determine troop movement, we need to further specify variables for fruit ripeness, tree height, and wind speed and direction. The concentration of the odors caused by each tree is calculated separately (CONC 1 and CONC 2). The basis for this calculation is the air pollution stack emission equations. These are empirically derived equations which give the dispersion of a pollutant such as sulfur dioxide emitted from a chimney, depending on the chimney height, the wind speed, and emission rate. The variables which begin with SIG. . ., X1, X2, Y1, Y2, RY, and RZ are the elements of these equations. The vertical emission velocity is set at zero. The corresponding part of the model is shown in Fig. 28.3.

The behavioral part of the model is shown in Fig. 28.4. First we add the odor concentrations from the two trees to form the total concentration at the troop location, CONCENTRATION. The problem here is to calculate the direction that optimally points up the odor "mountain," the direction of maximum odor increase.

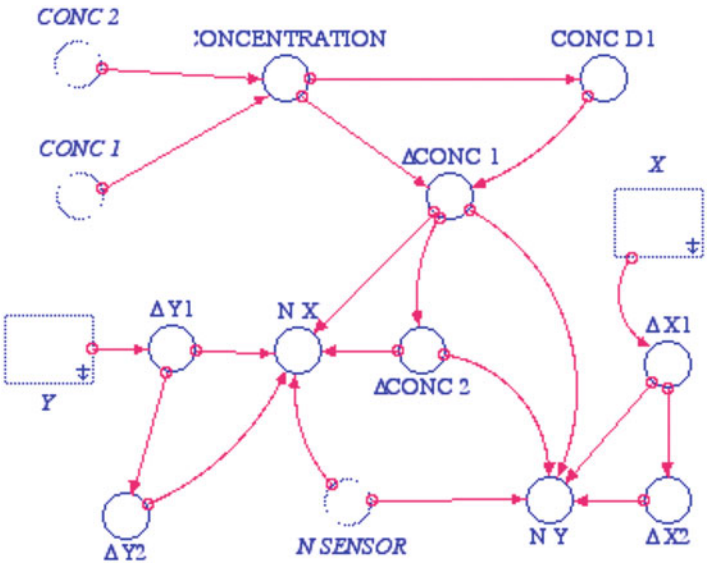

Fig. 28.4

The direction is to be calculated on the minimum necessary information: the location and odor strength at three points. These three points are the current location and the last two locations of the troop. The monkeys must remember the locations and the odor concentrations at the last two points and compare them with their current situation. They then have the minimum necessary information.

Exactly how the monkeys would figure which direction is the maximum ascent up the odor mountain is not known of course, nor do we know how we would do it if we were in the monkeys place. Perhaps each individual has a sort of "stereo" olfactory system and with their two nostrils can detect odor gradients; perhaps they sense these gradients by just moving their heads? Here we assume that they develop a sense of the odor gradient by moving their entire body and refiguring the gradient periodically. If we were to use odor to find its source, we could ascend the odor mountain in the direction of steepest ascent but delete and not solve any computer programs. Yet to mimic the monkey pattern, we assume that they do solve the problem and we proceed with a mathematical solution. That solution requires that we calculate the normal to the plane formed by the last two locations of the troop and the current location. The projection of this normal vector onto the horizontal plane, components N X and N Y, gives the direction of the steepest descent. The opposite direction to this projection is the direction of steepest ascent, or, more precisely, a local approximation to the true direction of steepest ascent.

The plane formed by these three points is approximately a plane tangent to the odor mountain in the area of the three points. It is possible for the normal to point down rather than up on certain occasions. This possibility requires the calculation of a detector for the normal vector's orientation, the N SENSOR (Fig. 28.5).

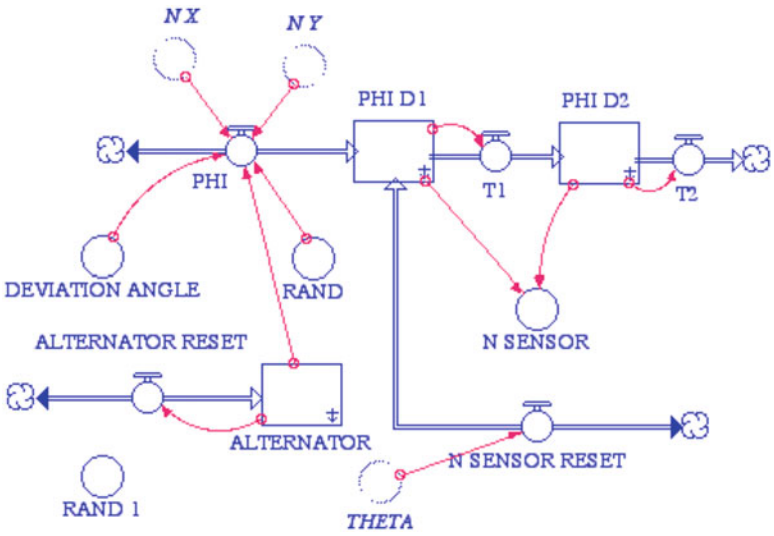

Fig. 28.5

It is calculated from knowledge of the change in the directions taken in the past, PHI D1 and PHI D2, the two previous directions which the troop has taken. Still another possibility exists. The directions could be so accurate that the plane formed is perpendicular to the X–Y plane. In this case, the direction of the normal vector is ambiguous. Fortunately, we need to add a small random variation (RAND) to the angle chosen for the next direction (PHI), and that variation nearly always prevents the ambiguous case. On the rare occasion when the ambiguous case does form, the troop will take a 90° turn to the right or left, randomly.

The model was "calibrated" to the best extent possible by comparing the averaged field data to the trip distances travelled in the model. We devised a CIRCULARITY measure which compares the distance actually travelled to the straight line distance between the starting point and the tree. We adjusted the random angle and visibility variables until the model circularity agreed with the values found in the field. Unfortunately, we have no data on the wind speed or direction in the canopy in the rain forest. We do have a reference on canopy wind speeds in another rain forest and we used an approximate average daily value of 0.5 m/s at the upper canopy level for our model.

Sometimes, the monkeys in the field will be stopped by a wide stream as they proceed in their search for food. In such cases they seem to abandon that search path and start out on another. Such a situation is not covered by the present model. Nor is the model sensitive to absolute odor levels. It is assumed in the model that they can sense the slightest of odors and only relative odor strengths can influence their path. After these brief calibrations, we are as ready as possible to test the model.

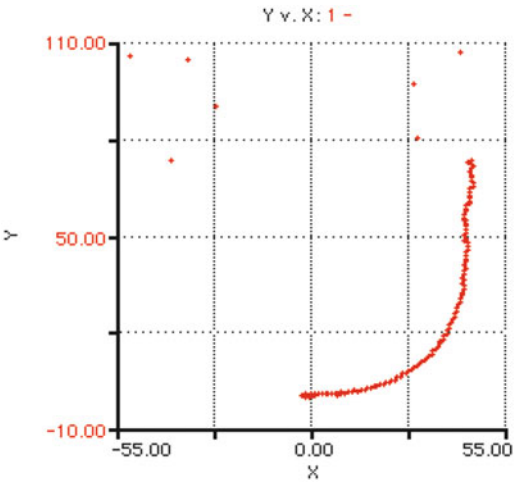

Fig. 28.6

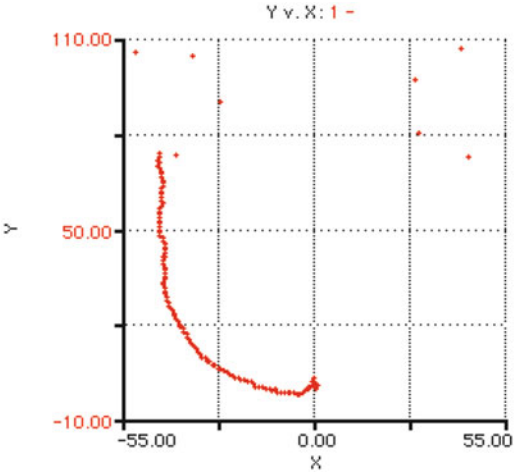

Fig. 28.7

First we set up the troop at the center, lower edge of the X–Y plane, and turn the emissions from the left tree off by setting its "ripeness" to zero. We set the wind at 0.5 m/s from the North. The results are shown in Fig. 28.6.

Next we turn the left tree on with the same odor emission rate as the right tree, and run again. The results are shown in Fig. 28.7.

In the first case (Fig. 28.6), the troop starts out heading to the right of the tree but gradually curves into the goal. With both trees emitting evenly (Fig. 28.7), the troop starts out headed between the trees and rather suddenly turns up the odor "ridge"

formed by the left tree and then proceeds along the ridge to that tree. Position the troop close between the trees and you will see that the "ridge" effect is more pronounced.

One other interesting result seems worth mentioning. If we plot the odor concentration directly downwind of the tree, we find that, due to ground reflection of the odor, the peak odor is at the canopy level only near the emitting tree. As one moves away from that tree, the peak concentration falls both in intensity of course, but also in height. So, for very distant fruit trees, the best place to be is not in the canopy but down the trunk, perhaps half way to the ground. There is another monkey species which travels with the Tamarin and which stays at about the mid-height level of the tree, until the fruit tree is reached. These lower monkeys are in a better position to direct the combined troop in the early stages of the search for a distant tree with ripe fruit.

Although quite elaborate, our model has several shortcomings. The diffusion equations are for a uniform medium, usually air. In the actual forest, we do not have such a medium. The leaves and branches of the trees no doubt cause much more rapid mixing than an air-only medium. The leaves are not uniformly distributed vertically in the rain forest. Above the canopy, the model has a single medium (clear air). How would this affect the readings in the forest? What about temperature inversions caused by the forest? How would the model include such effects? Perhaps the monkeys use a mixture of odor tracking and memory. How would the memory effects be included? It is assumed that the monkeys can detect the difference between under and over ripe fruit. How can this be modeled?

28.2 Monkey Travels Model Equations

ALTERNATOR(t) = ALTERNATOR(t − dt) + (ALTERNATOR_RESET) ∗ dt
INIT ALTERNATOR = 1∗RAND_1
INFLOWS:
ALTERNATOR_RESET = −2∗ALTERNATOR
PHI_D1(t) = PHI_D1(t − dt) + (PHI + N_SENSOR_RESET − T1) ∗ dt
INIT PHI_D1 = 0 {Phi delayed.}
INFLOWS:
PHI = IF (TIME > 16) AND (N_X>=0) THEN ARCTAN(N_Y/N_X) + ALTERNATOR∗(RAND+DEVIATION_ANGLE) ELSE IF (TIME>16) AND (N_X<0) THEN ARCTAN(N_Y/N_X) + PI + ALTERNATOR∗(RAND +DEVIATION_ANGLE) ELSE 0 {The chosen angle for the next step.}
N_SENSOR_RESET = IF TIME =15 THEN THETA ELSE 0
DOCUMENT: (TIME ,â• 14) AND (TIME ,â§ 15)
OUTFLOWS:
T1 = PHI_D1
PHI_D2(t) = PHI_D2(t − dt) + (T1 − T2) ∗ dt
INIT PHI_D2 = 0

INFLOWS:
T1 = PHI_D1
OUTFLOWS:
T2 = PHI_D2
X(t) = X(t − dt) + (ΔX − CLEAR_X) ∗ dt
INIT X = 0 {The X horizontal distance to the troop from the origin of the designated space.}
INFLOWS:
ΔX = IF TIME > 14 THEN VISIBILITY∗COS(THETA)∗TARGET ELSE IF TIME = 13 THEN INITIAL_X ELSE IF TIME <= 11 THEN (R∗COS (TIME) + TREE_X)/DT ELSE 0 {Changing the X location of the troop of monkeys, looking for trees with ripe fruit.}
OUTFLOWS:
CLEAR_X = IF TIME <= 12 THEN X/DT ELSE 0
Y(t) = Y(t − dt) + (ΔY − CLEAR_Y) ∗ dt
INIT Y = 0 {The Y horizontal distance to the troop from the origin of the designated space.}
INFLOWS:
ΔY = IF TIME > 14 THEN VISIBILITY∗SIN(THETA)∗TARGET ELSE IF TIME = 13 THEN INITIAL_Y ELSE IF TIME <= 11 THEN (R∗SIN(TIME) + TREE_Y)/DT ELSE 0 {Changing the y location of the troop of monkeys, looking for trees with ripe fruit.}
OUTFLOWS:
CLEAR_Y = IF TIME <= 12 THEN Y/DT ELSE 0
TRAVEL_DISTANCE(t) = TRAVEL_DISTANCE(t − dt) + (STEP_DISTANCE) ∗ dt
INIT TRAVEL_DISTANCE = 0 {The distance travelled from the starting point of the troop to the circle of influence of the tree.}
INFLOWS:
STEP_DISTANCE = IF TIME >= 15 THEN VISIBILITY∗DT ELSE 0
CIRCUITY = (TRAVEL_DISTANCE+9)/(SQRT((TREE_X1−INITIAL_X)^2+ (TREE_Y1-INITIAL_Y)^2)−7.5)
CONCENTRATION = CONC_1+CONC_2 {The combined odor concentration at the canopy top at the current troop location.}
CONC_1 = IF X1 > 0 THEN EXP(−.5∗(Y1/SIG_Y1)^2)∗EXP(−.5∗(2∗TREE_ HEIGHT/SIG_Z1)^2)/(2∗PI∗SIG_Y1∗SIG_Z1)∗RIPENESS_1/WIND_SPEED ELSE 0 {plume and ground reflection conc at top of tree; ug/m3}
CONC_2 = IF X2 > 0 THEN EXP(−.5∗(Y2/SIG_Y2)^2)∗EXP(−.5∗(2∗TREE_ HEIGHT/SIG_Z2)^2)/(2∗PI∗SIG_Y2∗SIG_Z2)∗RIPENESS_2/WIND_SPEED ELSE 0 {See note in CONC 1}
CONC_D1 = DELAY(CONCENTRATION,1)
DEVIATION_ANGLE = PI/16
INITIAL_X = 0 {Specified X coordinate of starting point for the troop in the designated space.}
INITIAL_Y = 0

N_SENSOR = IF (PHI_D2<PI) AND ((PHI_D1>PHI_D2) AND (PHI_D1<(PHI_D2+PI))) THEN 1 ELSE IF (PHI_D2>=PI) AND ((PHI_D1>PHI_D2) OR (PHI_D1<(PHI_D2−PI))) THEN 1 ELSE −1 {Rel. angle sizes sets normal up or down.}

N_X = (−Δ_Y2∗ΔCONC_1 + ΔCONC_2∗Δ_Y1)∗N_SENSOR {The projection of the normal vector to the plane by the last two steps, on the X-Y plane. This is the X component of the max. rise vector.}

N_Y = (−ΔCONC_2∗Δ_X1 + Δ_X2∗ΔCONC_1)∗N_SENSOR {The y component of the projection of the normal vector to the plane by the last two steps, onto the X-Y plane. This is the y component of the max. rise vector}

R = 17.5 {At this radius (meters), the fruit can be seen by the troop.}

R1 = SQRT((X−TREE_X1)^2 + (Y−TREE_Y1)^2) {The straight line distance from the troop to the center of tree 1.}

R2 = SQRT((X−TREE_X2)^2 + (Y−TREE_Y2)^2) {The straight line distance from the troop to the center of tree 2.}

RAND = 2∗PI∗RANDOM(1,0)∗.05∗1

RAND_1 = IF RANDOM(0,2) > 1 THEN 1 ELSE −1

RIPENESS_1 = 0 {Specified fruit odor emission rate, ug/sec, from tree 1.}

RIPENESS_2 = 7000 {Specified fruit odor emission rate, ug/sec, from tree 2.}

RY = IF WIND_SPEED < 2 THEN .4 ELSE IF (WIND_SPEED >= 2) AND (WIND_SPEED <= 5) THEN .36 ELSE .32

RZ = IF WIND_SPEED < 2 THEN .4 ELSE IF (WIND_SPEED >= 2) AND (WIND_SPEED <= 5) THEN .33 ELSE .22 {For the odor dispersion equation. See SIG Y.}

R_Y = IF WIND_SPEED < 2 THEN .9 ELSE IF (WIND_SPEED >= 2) AND (WIND_SPEED <= 5) THEN .86 ELSE .78

R_Z = IF WIND_SPEED < 2 THEN 2 ELSE IF (WIND_SPEED >= 2) AND (WIND_SPEED <= 5) THEN .86 ELSE .78

SIG_Y1 = IF X1 > 0 THEN RY∗X1^R_Y ELSE 0 {A term in the odor dispersion equation.}

SIG_Y2 = IF X2 > 0 THEN RY∗X2^R_Y ELSE 0 {A term in the odor dispersion equation.}

SIG_Z1 = IF X1 > 0 THEN RZ∗X1^R_Z ELSE 0 {A term in the odor dispersion equation.}

SIG_Z2 = IF X2 > 0 THEN RZ∗X2^R_Z ELSE 0 {A term in the odor dispersion equation.}

TARGET = IF (TIME > 16) AND ((R1 <= 17) OR (R2 <= 17)) THEN 0 ELSE 1 {Senses relation of troop to tree circle of visual influence (17 m. radius). When this circle is reached, it is assumed that the troop can proceed by sight.}

THETA = IF (TIME >= 14) AND (TIME <= 15) THEN -2∗PI∗RANDOM(1,0) ELSE PHI

TREE_HEIGHT = 20 {Specified height of the ripe fruit and canopy of tree, meters.}

TREE_X = IF TIME < 6 THEN TREE_X1 ELSE IF TIME < 12 THEN TREE_X2 ELSE 0

TREE_X1 = −45 {X location of the first tree with ripe fruit in the general coordinate system for the designated space.}
TREE_X2 = 45 {X coordinate of tree 2 in the designated space.}
TREE_Y = IF TIME < 6 THEN TREE_Y1 ELSE IF TIME < 12 THEN TREE_Y2 ELSE 0
TREE_Y1 = 90 {Y location of the first tree w/ripe fruit in the general coord. system for the designated space.}
TREE_Y2 = 90 {Y coordinate of tree 2 in the designated space.}
VISIBILITY = IF TIME <=16 THEN .1 ELSE 1 {Specified step rate for the troop. 2 meters per minute. Actual Tamarin data from P. Garber: 300 meter max. distance or 86 minute max. travel time. Avg. travel time = 26 min.}
WIND_ANGLE = WIND_ANGLE_DEGREES*PI/180 {wind angle in radians, counter-clockwise from + x axis.}
WIND_ANGLE_DEGREES = -90 {Specified wind angle, degrees; counter-clockwise from the + x axis, from the origin.}
WIND_SPEED = 0.5
X1 = (X−TREE_X1)*COS(WIND_ANGLE) + (Y−TREE_Y1)*SIN (WIND_ANGLE) {The troop location coordinate, relative to tree 1, in terms of the plume; X1 parallel to wind, Y1 perpendicular to X1.}
X2 = (X−TREE_X2)*COS(WIND_ANGLE) + (Y−TREE_Y2)*SIN (WIND_ANGLE) {The troop location coordinate, relative to tree 2, in terms of the plume; X2 parallel to wind, Y2 perpendicular to X2.}
Y1 = (Y − TREE_Y1)*COS(WIND_ANGLE) − (X − TREE_X1)*SIN (WIND_ANGLE) {The troop location coordinate, relative to the tree 1, in terms of the plume; X1 parallel to wind, Y1 perpendicular to X1.}
Y2 = (Y−TREE_Y2)*COS(WIND_ANGLE) − (X−TREE_X2)*SIN (WIND_ANGLE) {The troop location coordinate, relative to tree 2, in terms of the plume; X2 parallel to wind, Y2 perpendicular to X2.}
Δ_X1 = X−DELAY(X,1) {See note ΔY2.}
Δ_X2 = DELAY(Δ_X1,1) {See note ΔY2.}
Δ_Y1 = Y−DELAY(Y,1)
Δ_Y2 = DELAY(Δ_Y1,1) {The Y location of the troop 2 steps ago. We now have 3 odor readings at 3 known locations. This forms a plane, the normal to which gives us the local approximation of the direction of steepest ascent up the odor hill.}
ΔCONC_1 = CONCENTRATION−CONC_D1
ΔCONC_2 = DELAY(ΔCONC_1,1)

Reference

1. Garber P, Hannon B (1993) Modeling monkeys: a comparison of computer generated and empirical measures. Int J Primatol 14:827–852

Chapter 29
Biosynchronicity

With Heav'nly touch of instrumental sounds In full harmonic number joind.

(1667 Milton)

29.1 Firefly Model

We know from the observation of fireflies in India that whole trees containing tens of thousands of these insects begin to blink in unison shortly after dusk [1]. Casual observation of the sounds of nighttime insects around the common suburban home shows us that audio-synchronous behavior occurs. We assume some group reproduction advantage is conferred by such synchrony. The pacemakers in the heart of every mammal are really the synchronous pulsing of thousands of special cells, yielding sufficient signal to cause a muscle action. What process allows such synchronization? How can these organisms and even cells conform to each other's signal?

Apparently, Charles Peskin of New York University first successfully formulated a model of this process. Nearly any electrical engineer would understand the process immediately, as he began with an electrical analogy: a resistance and a capacitor in parallel, subjected to a steady electrical current input. The voltage builds on the capacitor to a limit when it suddenly discharges and the voltage drops quickly to zero, only to repeat the process. This system is analogous to a weight hanging on a damper, subjected to a constant extending force. When the damper reaches its limiting extension the velocity of the weight becomes zero.

In the model, we represent four fireflies by four cells of a spatial model. The state variable for each cell is the "voltage" V—or brightness—of the cell. We choose different starting values for each cell in order to get the cells initially out of phase.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_29,

When the TOTAL V—total voltage, or in our case the brightness of the combined flashes—exceeds a given level, each cell boosts its own voltage by a small amount (BOOST STRENGTH). That boost voltage hastens the time for the receiver to reach its peak and discharge. Assuming the parameters are within certain limits, this process generates an outcome where cells gradually merge into synchronous firing.

Even if the cells are significantly different, they can still be driven into synchrony but the reactions times of the cells (DT) must be shorter, the booster signal strength must be larger, and the resulting synchronous frequency is greater. These requirements probably mean that the organisms must be more highly developed and more energy demanding if they are not identical and therefore, the organic forms of these cells would tend to become genetically more similar through time, at least with regard to their flashing mechanisms. Such an evolutionary direction would tend to reduce or even avoid the biological cost of faster reaction times and high strength booster signals.

In this model we represent four such systems subjected to a common steady INPUT CURRENT. Reactance (R) is the value of the resistance constant and Capacitance (C) is the measure of the capacitor constant (Fig. 29.1).

The governing differential equation for each firefly is a simple linear one:

$$C^* \, dV/dt + V/R = \text{INPUT CURRENT} = 0.15 \tag{29.1}$$

that is solved for each of the four such cells.

As shown in Figs. 29.2 and 29.3, the individual and TOTAL V is erratic at first but grows in size and fluctuation as the cells become synchronized.

Experiment with this model. See what the first DT and BOOST STRENGTH settings give when the cells are not the same. Change the input current level. All sorts of interesting results can occur. You should even find, as Strogatz and Stewart [1] report, that there are a variety of steady conditions where the peaks are not synchronous. All sorts of interesting possibilities lurk in the dynamics of these two interconnected pulsing cells.

Think of those thousands of fireflies. Are each of them interconnected to only one other? Or does each connect to only its nearest neighbors, in a kind of regional association, with the regions eventually acting as a single unit that must swing somehow into synchrony with other regional units—a kind of hierarchy of synchronous behavior? Or does each somehow average the peak of the signal from the whole and adjust its own flash initiation? Add more cells and try out these and perhaps other ideas. You will find no doubt that this glorious process of nature is not as complex a process as you might have thought.

We may imagine that the group flashing is the behavior of individuals whose reproductive chances are enhanced by synchronous behavior—to attract distant mates into the proximity from a long distance. But what happens when the attracted mate is close? The appeal of belonging to a group is lost—the act of mating is not a many-to-one relationship, it is the ultimate in one-to-one behavior. Maybe those in the vicinity of attractee stop flashing once they realize the situation. Maybe, once those locals notice the newcomer of the opposite sex, only these locals begin to

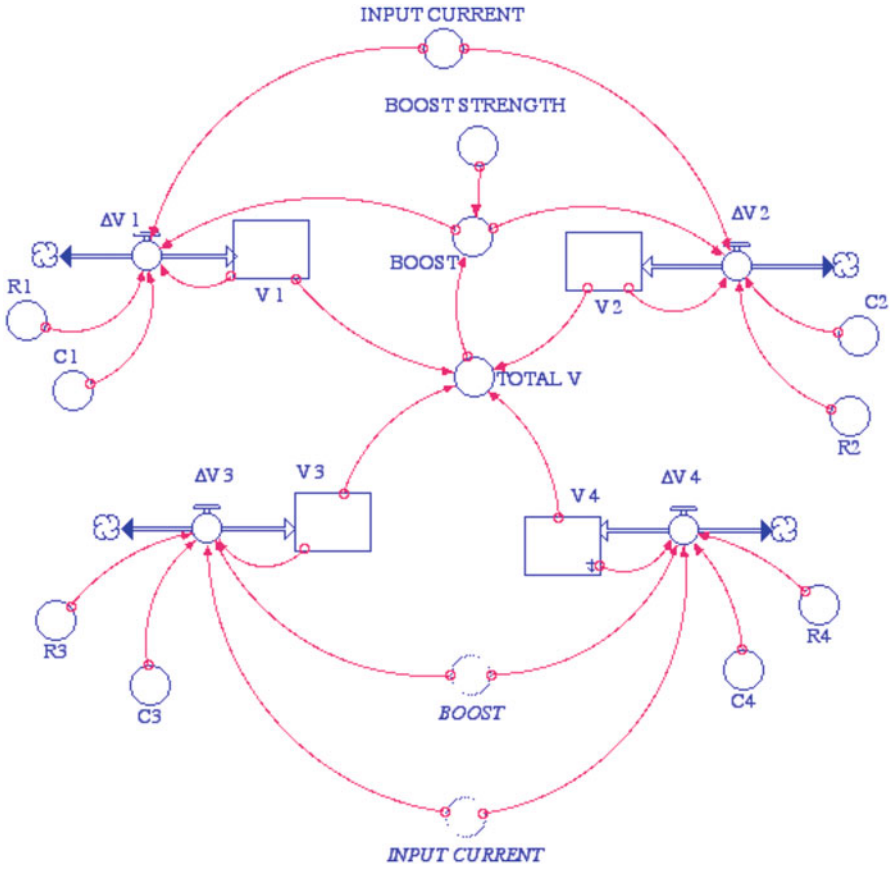

Fig. 29.1

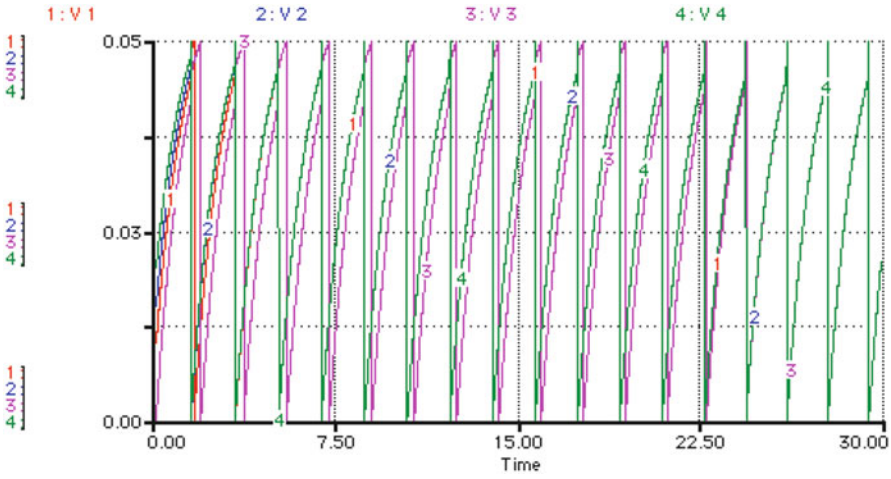

Fig. 29.2

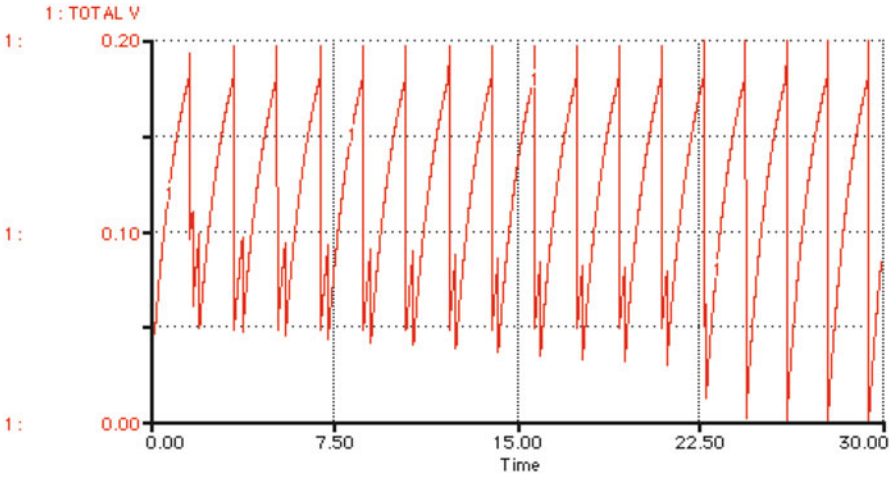

Fig. 29.3

flash synchronously with that of the attractee, while the bulk of the group continues to flash in synchrony. With enough attractees the group synchrony falls apart, destroying the potential for success.

29.2 Biosynchronicity Model Equations

V_4(t) = V_4(t − dt) + (ΔV_4) ∗ dt
INIT V_4 = 0.02
INFLOWS:
ΔV_4 = if (V_4 <= .05) then (INPUT_CURRENT−V_4/R4)/C4+BOOST else −V_4/dt
V_1(t) = V_1(t − dt) + (ΔV_1) ∗ dt
INIT V_1 = 0.01
INFLOWS:
ΔV_1 = if V_1<=.05 then (INPUT_CURRENT−V_1/R1)/C1+BOOST else −V_1/dt
V_2(t) = V_2(t − dt) + (ΔV_2) ∗ dt
INIT V_2 = 0.015
INFLOWS:
ΔV_2 = if (V_2 <= .05) then (INPUT_CURRENT−V_2/R2)/C2+BOOST else −V_2/dt
V_3(t) = V_3(t − dt) + (ΔV_3) ∗ dt
INIT V_3 = 0

INFLOWS:
$\Delta V\_3 = \text{if } V\_3 <= .05 \text{ then } (INPUT\_CURRENT - V\_3/R3)/C3 + BOOST \text{ else } -V\_3/dt$
$BOOST = IF\ (TOTAL\_V > .18)\ THEN\ BOOST\_STRENGTH\ ELSE\ 0$
$BOOST\_STRENGTH = .001/dt$
$C1 = 3$
$C2 = 3$
$C3 = 3$
$C4 = 3$
$INPUT\_CURRENT = 0.15$
$R1 = 0.4$
$R2 = 0.4$
$R3 = 0.4$
$R4 = 0.4$
$TOTAL\_V = V\_1 + V\_2 + V\_3 + V\_4$

Reference

1. Strogatz S, Stewart I (1993) Coupled oscillators and biological synchronization. Sci Am 269 (6):102–109

Part VI
Multiple Population Models

Chapter 30
Plant–Microbe Interaction

The soil is the great connector of lives, the source and destination of all. It is the healer and restorer and resurrector, by which disease passes into health, age into youth, death into life. Without proper care for it we can have no community, because without proper care for it we can have no life.

(Wendell Berry, The Unsettling of America: Culture and Agriculture)

30.1 Plant–Microbe Interaction Model

Soil microbial communities actively interact with terrestrial plant communities, and they are documented to influence the community structure of the plant communities. We use here the soil feedback model introduced by Bever [1] to predict how soil microbes may influence plant community structure by providing beneficial or detrimental feedbacks. Figure 30.1, from Bever [1] shows the structure of the soil feedback model. N_A and N_B represent the populations of two plant species, and S_A and S_B represent the soil microbe communities that associate with A and B respectively. The effect of plant A on its soil community S_A is l, and the effect of plant B on its soil community is ν. The effect from soil communities S_A to plant A is represented by α_A, the effect to plant B is represented by α_B. Similarly, the effect from soil community B on plants A and B are represented by β_A and β_B respectively. The competition between plant species is assumed to follow the Lotka-Volterra model (see Chap. 2 for basic illustrations). The competition effect of species A on species B is C_A, and the competition effect of species B on species A is C_B.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_30,

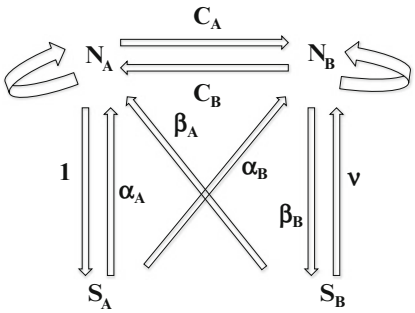

Fig. 30.1

The corresponding STELLA model is shown in Fig. 30.2. In the model the feedback from soil community to the plants is quantified by the interaction coefficient I_s, which in turn is defined as

$$I_s = \alpha_A + \beta_B - \alpha_B - \beta_A \tag{30.1}$$

The direction of interaction coefficient, I_s, captures co-existence of two plant species. Additionally, in natural systems, the succession of plant communities may be affected by the delayed feedback effect from the soil communities [2].

With the assumption that both plant and microbe communities start with an even relative abundance of 0.5, we change the feedback coefficient to arrive at alternate co-existence patterns of plant A and plant B. Data used in this model is from the paper published by Bever [1], van Wesenbeeck et al. [3] and Yamazaki et al. [4].

Figures 30.3 and 30.4 show the plant community dynamics without the soil community feedback ($I_s = 0$). As shown in Fig. 30.1, plant A goes extinct in the given field within 50 years. This is because of the fact that we assumed, for this model run, plant B to be a stronger competitor ($C_A = 0.885$, $K_A = 100$; $C_B = 0.98$, $K_B = 120$).

What are the effects of strong negative feedbacks on the coexistence of plant A and plant B? To answer this question we first adde a strong negative soil community feedback ($I_s = -0.43$) to the system (Figs. 30.5 and 30.6). Instead of extinction of one species, plant A and plant B take turns to dominant the system. The dynamics of the relatively dominant soil microbial communities changes over time, Fig. 30.5).

When we increase the strength of feedback from strongly negative ($I_s = -0.43$) to strongly positive ($I_s = 0.43$), we observe that the plant community dynamics quickly changes from one where plant A and B take turns to dominant to one where plant A becomes distinct (Fig. 30.7). As the feedback strength becomes more positive, plant A goes to extinction more quickly.

So far, the model does not allow for any randomness in the system. How may the randomness of the growth coefficients RA and RB, the carrying capacities KA and KB, and the delayed effect change the coexistence of plants A and B? To answer this question we introduced randomness to the growth coefficients RA and RB (Fig. 30.8), and to the carrying capacities KA and KB (Fig. 30.9) separately.

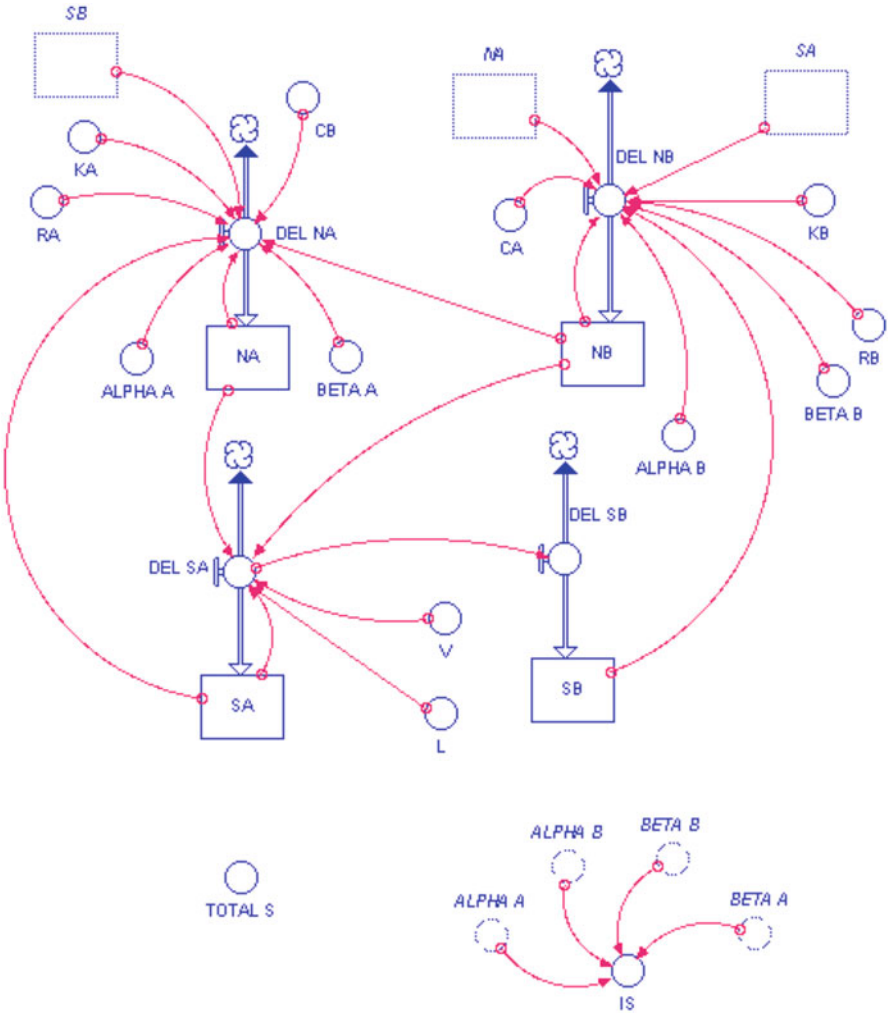

Fig. 30.2

As shown in Fig. 30.8, where RA = RANDOM(0.5,0.9) and RB = RANDOM (0.3,0.7), adding randomness to the growth factors barely changes the plant community dynamics. In contrast, adding randomness to the carrying capacities brings in a lot of fluctuation in the local stable points, and also slows down the pace of plant community changes. This is shown in Fig. 30.9, where KA = RANDOM (80,120), and KB = RANDOM(110, 130).

For some final experimentation with our model, we assume that soil communities only have a 1-year delayed effect on the plant seeds (Fig. 30.10). Adding this delay slows down the plant community changes, and also affects the way in which the community changes over time.

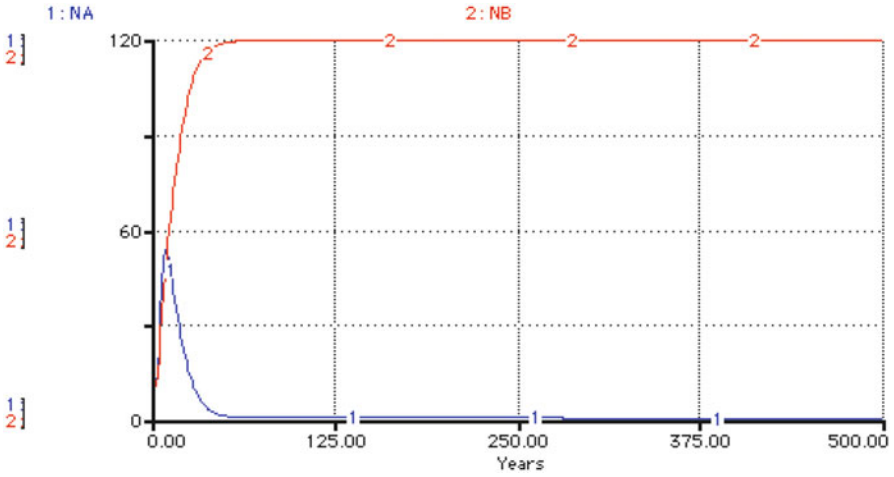

Fig. 30.3

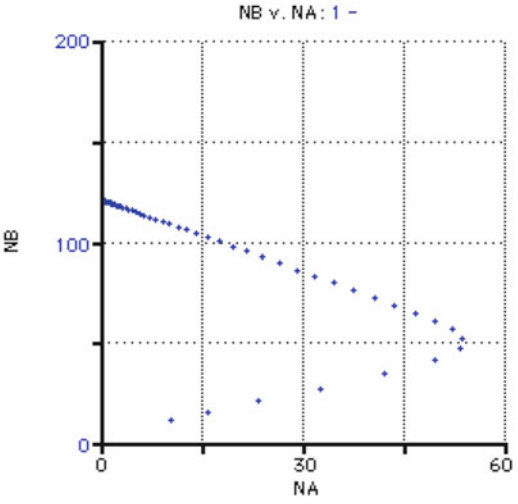

Fig. 30.4

In this model, soil communities are stable systems. In reality, however, soil communities may be dynamically changed, which consequently influences both soil community and plant species. For example, one common impact on soil communities comes from microbial immigration and dispersion. Dispersion may weaken the influence of plants on soil microbes, then our model should underestimate the delay brought by randomness. Does change of soil community over time affect plant coexistence? The model above provides a basis to answer this question. You can also use the model to capture the density dependence of feedback effect.

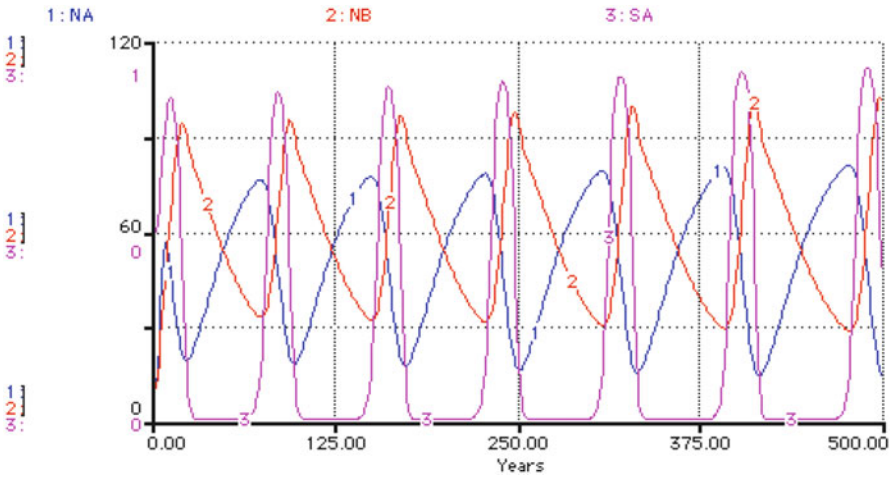

Fig. 30.5

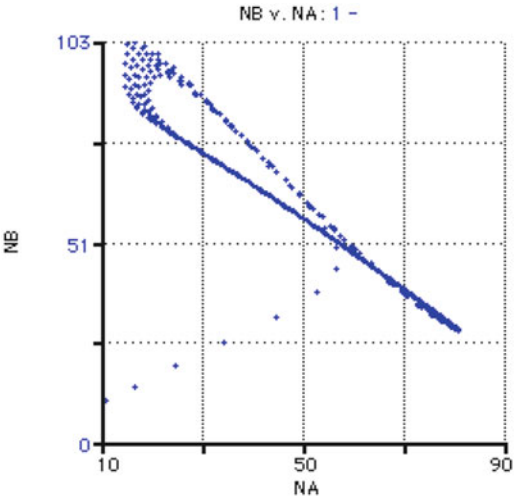

Fig. 30.6

The effect of soil community to plant is dependent on the density of plant species. For example, the negative effect of a pathogen is expected to be increased as the density of plant is increased, or in other words, the sensitivity of plants to pathogens is increased when their relative abundance is high [4].

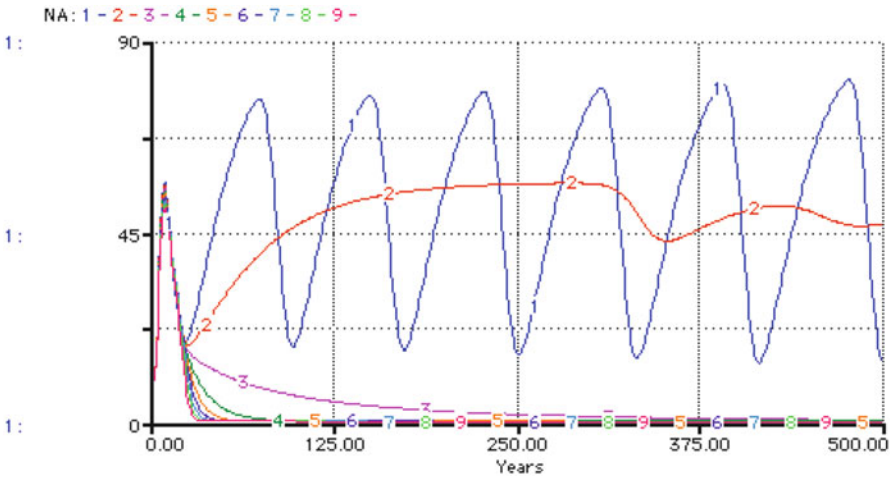

Fig. 30.7

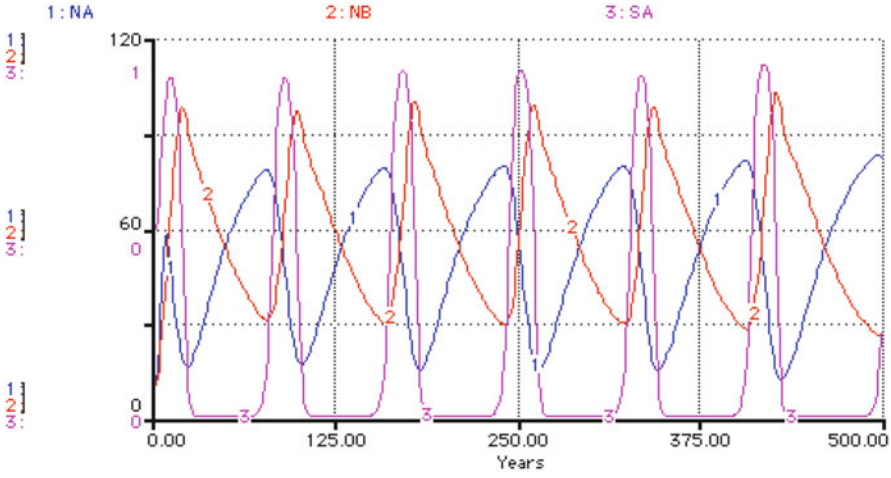

Fig. 30.8

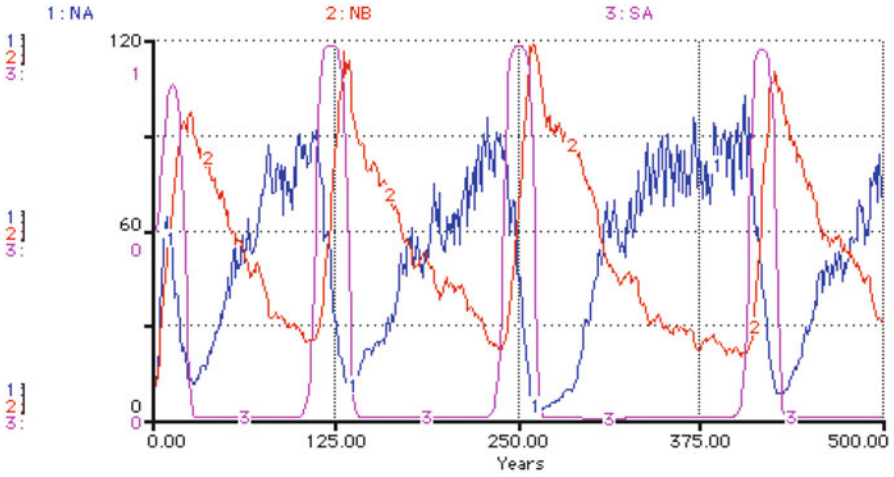

Fig. 30.9

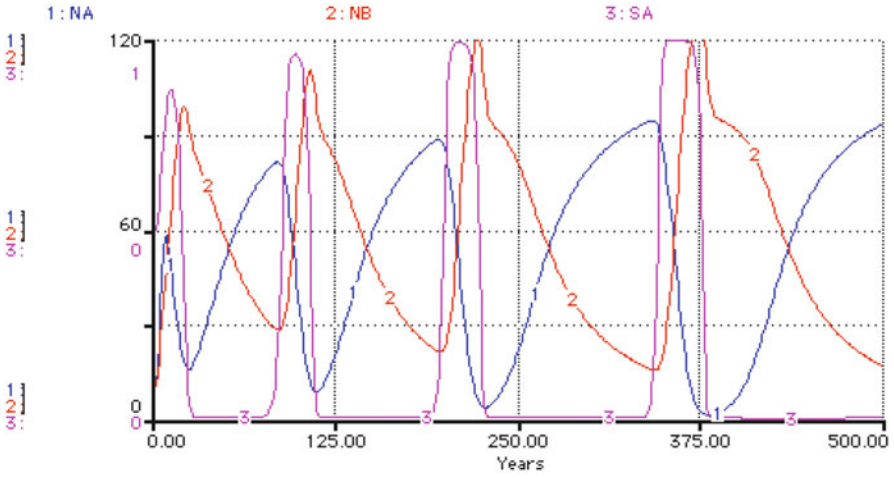

Fig. 30.10

30.2 Plant–Microbe Interaction Model Equations

NA(t) = NA(t − dt) + (DEL_NA) ∗ dt
INIT NA = 10
INFLOWS:
DEL_NA = RA∗NA∗(1+ALPHA_A∗Delay(SA, 1)+BETA_A∗Delay(SB, 1)−(NA+CB∗NB)/KA)∗DT
NB(t) = NB(t − dt) + (DEL_NB) ∗ dt
INIT NB = 10
INFLOWS:
DEL_NB = RB∗NB∗(1+ALPHA_B∗Delay(SA, 1)+BETA_B∗Delay(SB, 1)−(NB+CA∗NA)/KB)∗DT
SA(t) = SA(t − dt) + (DEL_SA) ∗ dt
INIT SA = 0.5
INFLOWS:
DEL_SA = SA∗(1−SA)∗(L∗NA/(NA+NB)−V∗NB/(NA+NB))∗DT
SB(t) = SB(t − dt) + (DEL_SB) ∗ dt
INIT SB = TOTAL_S−SA
INFLOWS:
DEL_SB = −DEL_SA
ALPHA_A = −0.03
ALPHA_B = 0.1
BETA_A = 0.1
BETA_B = −0.2
CA = 0.885
CB = 0.98
IS = ALPHA_A−ALPHA_B−BETA_A+BETA_B
KA = 100
KB = 120
L = 1
RA = 0.7
RB = 0.5
TOTAL_S = 1
V = 0.8

References

1. Bever JD (2003) Soil community feedback and the coexistence of competitors: conceptual frameworks and empirical tests. New Phytol 157(3):465–473
2. Fukami T, Nakajima M (2013) Complex plant-soil interactions enhance plant species diversity by delaying community convergence. J Ecol 101(2):316–324

3. van Wesenbeeck BK, van de Koppel J, Herman PMJ et al (2008) Potential for sudden shifts in transient systems: distinguishing between local and landscape-scale processes. Ecosystems 11 (7):1133–1141
4. Yamazaki M, Iwamoto S, Seiwa K (2009) Distance- and density-dependent seedling mortality caused by several diseases in eight tree species co-occurring in a temperate forest. Forest Ecol Recent Adv Plant Ecol 201(1):181–196

Chapter 31
Wildebeest

There is another species of wild ox, called by the natives gnoo.

(G. Forster, 1777)

31.1 Wildebeest Model

The Wildebeest model is developed for a wildlife park on African grassland. Wildebeest are eaten by lions, and shot by park rangers attempting to manage the ecosystem. The data on the number of lions in the park is thought to be between 400 and 600. We choose 500 for the initial runs of the model. The rate at which wildebeest are killed by the lions varies between 4.5 and 3.8 wildebeest per lion in the first 6 years of the data (Fig. 31.1).

The wildebeest were shot by the park rangers during the first 4 years. The calf survival rate varies from 0.35 to 0.48 during the first 6 years (Fig. 31.2). Females constitute 70 % of the total population. The first 6 years have population census estimates for the calves, yearlings, 2-year olds, and adults (Fig. 31.3).

The lion population of our model should depend on the availability of food, which is in our case is mostly the non-calf population of wildebeest. This dependency is given as shown in Fig. 31.4. Note that it is critical here to specify the graphical function in ways that extrapolate from the last entry on the non-calf population to higher, rather than constant, lion numbers.

Given the population of YEARLINGS, TWO YEAR OLDS and ADULTS we can calculate an average death rate assumed to apply to these "cohorts". This death rate is assumed to be distinct from that for calves (Fig. 31.5).

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_31,

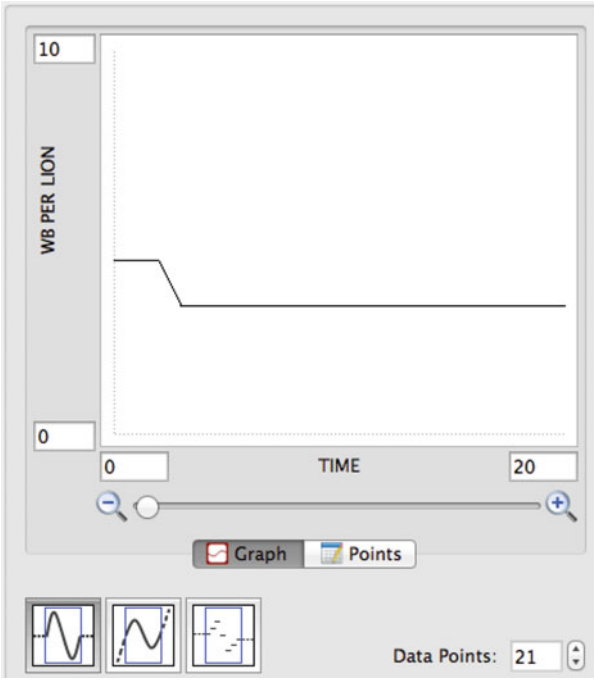

Fig. 31.1

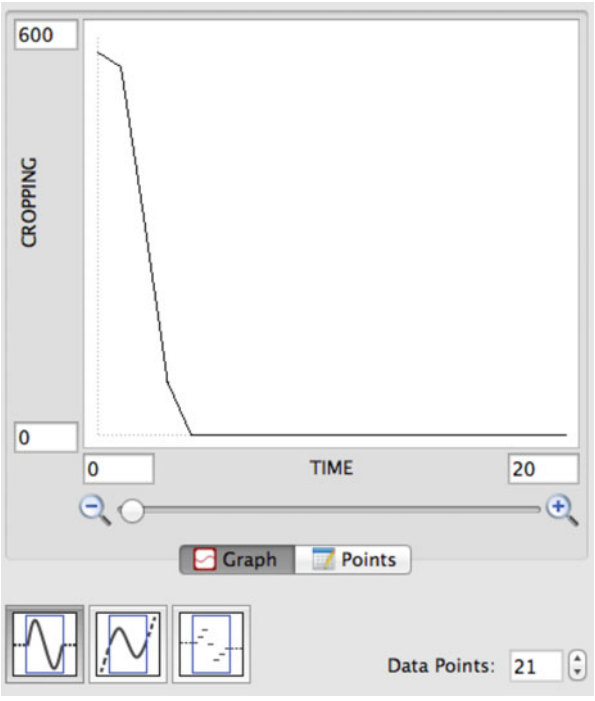

Fig. 31.2

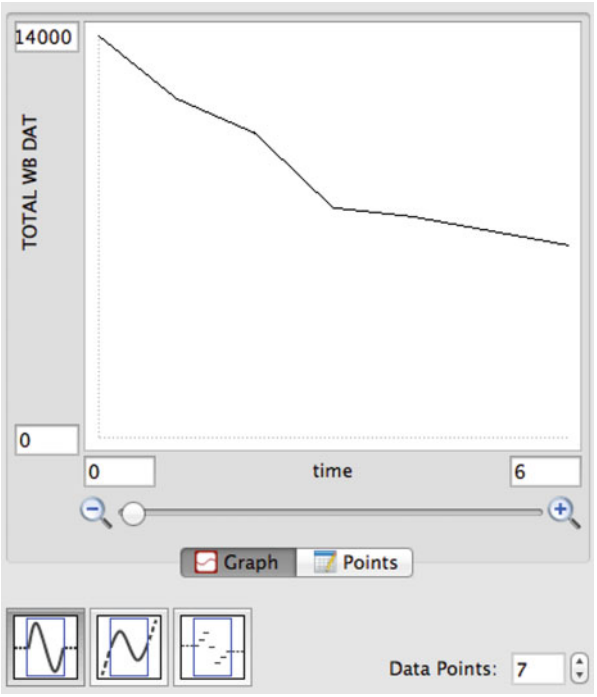

Fig. 31.3

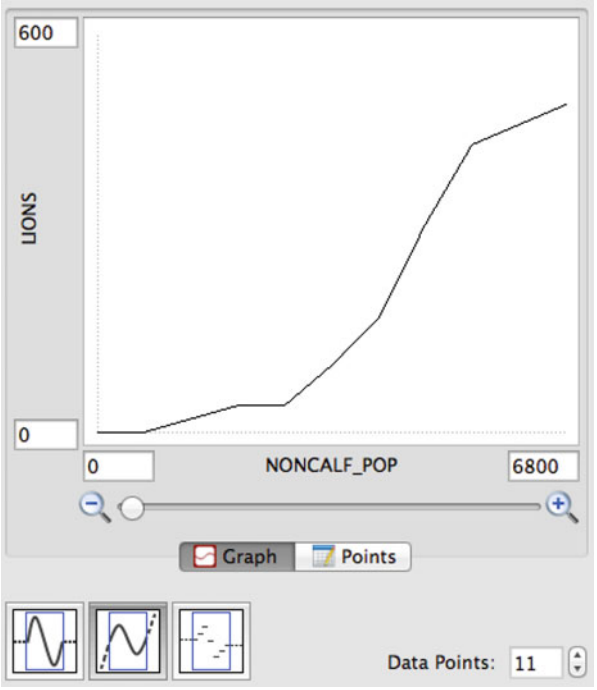

Fig. 31.4

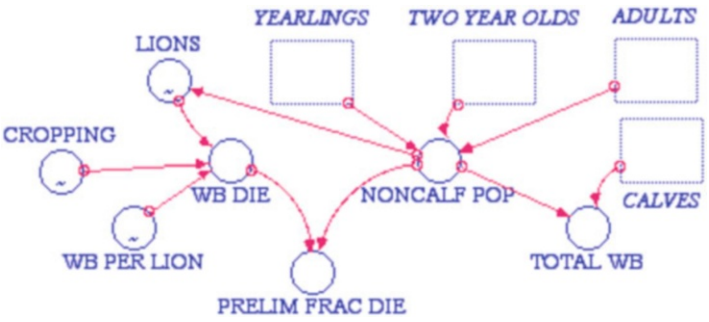

Fig. 31.5

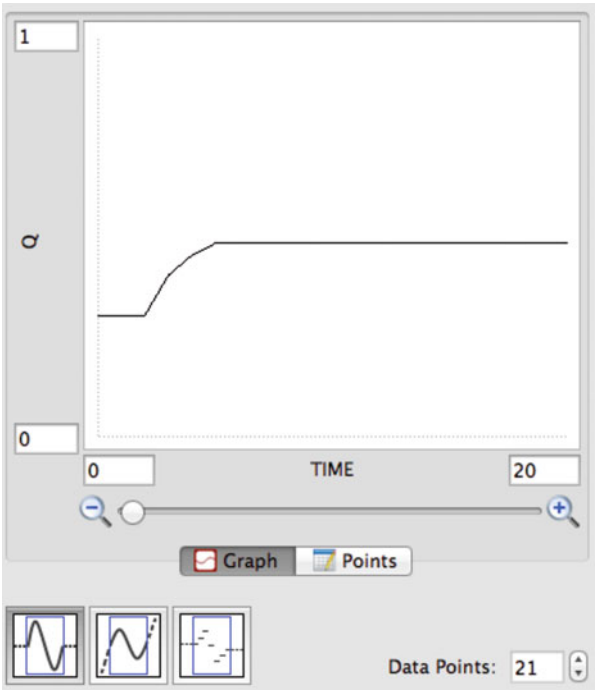

Fig. 31.6

The death rate for calves is calculated from data on calf survival rates Q as (1-Q). We fit the data (see [1]) by adjusting the calf survival rate Q downward in the early years for the given estimates. This is done in Fig. 31.6.

We are now ready to put the pieces of our model together to assess changes in the wildebeest population given the influences of lions and rangers. Specifically, the remaining population cohort model for wildebeest is shown in Fig. 31.7.

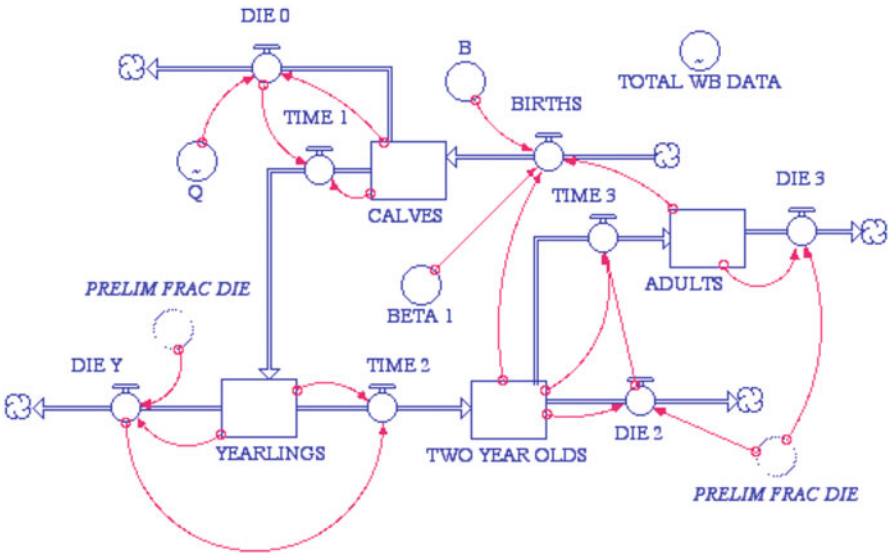

Fig. 31.7

One can now change all of the parameters in turn to find the sensitivity of each, i.e., the kill rate, the number of lions, the calf survival rate, the fecundity coefficients BETA and B for the 2 year olds and for adults, respectively, and the female fraction.

Note well how the Dying of each stock is subtracted from the TIME flows. This is done since this is a model of aging and the two alternatives die or advance to the next age group. This is always true in modeling populations where the independent variables include age cohort. It is not true for example in water flowing from a reservoir where the flow options are evaporation and release. . .the water does need to move on. It is true in say the hatching of eggs: in any model time period the egg may die, hatch, or continue to mature. In these cases, the dying flows (evaporation, egg death) are not subtracted from the advancing flow (water release, hatching). In general, the aging flow is the donor stock minus the dying rate, with this difference divided by the residence time in the donor.

The results of our model are shown in Figs. 31.8 and 31.9. Total wildebeest numbers, as calculated in the model, and available data on the population size are compared in Fig. 31.8. The individual cohort numbers are shown in Fig. 31.9. That figure also contains information on the corresponding lion population, which stabilizes as well.

Run the model yourself and test for the sensitivity of the parameters. How would you modify this model to correct the lion population and eating rate to bring the wildebeest herd to a 5,000 animal steady state?

Lions are just one factor in the system affecting the wildebeest population. The amount of rainfall influences grass height during the calving season which in turn influences the predation rate on calves. Making predictions about the rainfall

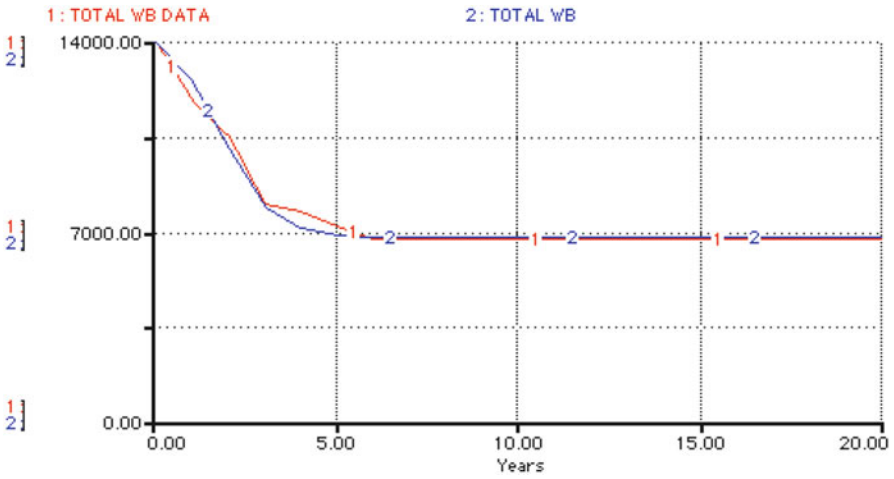

Fig. 31.8

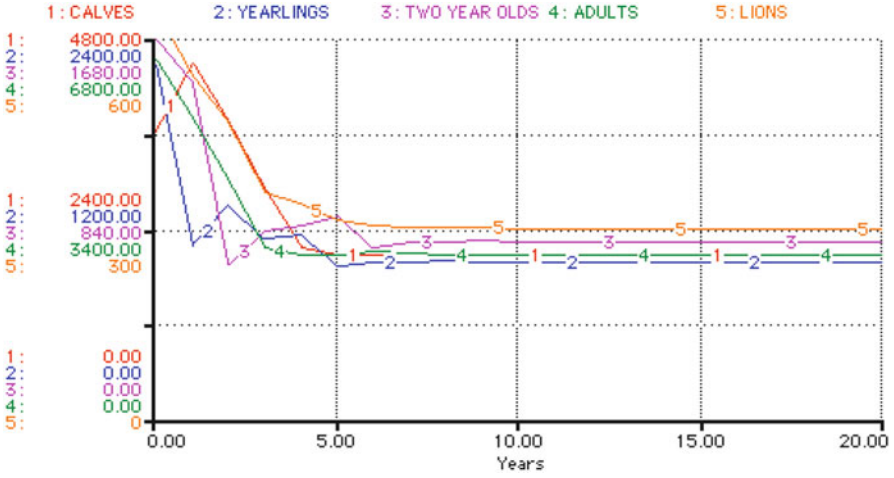

Fig. 31.9

variations can determine the expected appropriate cropping rates for wildebeests and lions. Introduce seasonal rainfall into your model and find a (non-zero) lion population and eating rate that prevent the wildebeest population from crashing.

31.2 Wildebeest Model Equations

$$\text{ADULTS}(t) = \text{ADULTS}(t - dt) + (\text{TIME\_3} - \text{DIE\_3}) * dt$$
$$\text{INIT ADULTS} = 6440 \ \{\text{Individuals}\}$$

INFLOWS:
TIME_3 = TWO_YEAR_OLDS−DIE_2 {Net graduation of the two year olds to adulthood. Individuals per Time Period}
OUTFLOWS:
DIE_3 = ADULTS∗PRELIM_FRAC_DIE {Individuals per Time Period}
CALVES(t) = CALVES(t − dt) + (BIRTHS − TIME_1 − DIE_0) ∗ dt
INIT CALVES = 3640 {Individuals}

INFLOWS:
BIRTHS = (BETA∗TWO_YEAR_OLDS + B∗ADULTS)∗.7 {.7 is the fraction of the adult population which is female. Fecundity is greater among the adults. Individuals per Time Period}
OUTFLOWS:
TIME_1 = CALVES − DIE_0 {Individuals per Time Period}
DIE_0 = (1 - Q)∗(CALVES) {The survival rate q improves with time, reducing the death rate of the calves. Individuals per Time Period}
TWO_YEAR_OLDS(t) = TWO_YEAR_OLDS(t − dt) + (TIME_2 − TIME_3 − DIE_2) ∗ dt
INIT TWO_YEAR_OLDS = 1680 {Individuals}

INFLOWS:
TIME_2 = YEARLINGS−DIE_Y {The net number of yearlings graduating to two year olds. Individuals per Time Period}
OUTFLOWS:
TIME_3 = TWO_YEAR_OLDS−DIE_2 {Net graduation of the two year olds to adulthood. Individuals per Time Period}
DIE_2 = TWO_YEAR_OLDS∗PRELIM_FRAC_DIE {Individuals per Time Period}
YEARLINGS(t) = YEARLINGS(t − dt) + (TIME_1 − TIME_2 − DIE_Y) ∗ dt
INIT YEARLINGS = 2240 {Individuals}

INFLOWS:
TIME_1 = CALVES − DIE_0 {Individuals per Time Period}
OUTFLOWS:
TIME_2 = YEARLINGS−DIE_Y {The net number of yearlings graduating to two year olds. Individuals per Time Period}
DIE_Y = YEARLINGS∗PRELIM_FRAC_DIE {Individuals per Time Period}
B = .92 {Given adult female fecundity. Births per Female}
BETA = .3 {Given two year old female fecundity. Births per Female}
NONCALF_POP = YEARLINGS + TWO_YEAR_OLDS + ADULTS
PRELIM_FRAC_DIE = WB_DIE/NONCALF_POP {A death rate assumed to apply to yearlings, two year olds and adults. Individuals per Individuals per Time Period}
TOTAL_WB = CALVES+NONCALF_POP {Individuals}
WB_DIE = (WB_PER_LION∗LIONS + CROPPING∗1) {The number of wildebeest eaten and shot in the game park per year.}

CROPPING = GRAPH(TIME)
(0.00, 572), (1.00, 550), (2.00, 320), (3.00, 78.0), (4.00, 0.00), (5.00, 0.00), (6.00, 0.00), (7.00, 0.00), (8.00, 0.00), (9.00, 0.00), (10.0, 0.00), (11.0, 0.00), (12.0, 0.00), (13.0, 0.00), (14.0, 0.00), (15.0, 0.00), (16.0, 0.00), (17.0, 0.00), (18.0, 0.00), (19.0, 0.00), (20.0, 0.00.
LIONS = GRAPH(NONCALF_POP)
(0.00, 0.00), (680, 0.00), (1360, 20.0), (2040, 40.0), (2720, 40.0), (3400, 100), (4080, 170), (4760, 310), (5440, 430), (6120, 460), (6800, 490.
Q = GRAPH(TIME)
(0.00, 0.3), (1.00, 0.3), (2.00, 0.3), (3.00, 0.4), (4.00, 0.45), (5.00, 0.48), (6.00, 0.48), (7.00, 0.48), (8.00, 0.48), (9.00, 0.48), (10.0, 0.48), (11.0, 0.48), (12.0, 0.48), (13.0, 0.48), (14.0, 0.48), (15.0, 0.48), (16.0, 0.48), (17.0, 0.48), (18.0, 0.48), (19.0, 0.48), (20.0, 0.48.
TOTAL_WB_DATA = GRAPH(time)
(0.00, 14000), (1.00, 11800), (2.00, 10600), (3.00, 8000), (4.00, 7700), (5.00, 7200), (6.00, 6700.
WB_PER_LION = GRAPH(TIME)
(0.00, 4.50), (1.00, 4.50), (2.00, 4.50), (3.00, 3.30), (4.00, 3.30), (5.00, 3.30), (6.00, 3.30), (7.00, 3.30), (8.00, 3.30), (9.00, 3.30), (10.0, 3.30), (11.0, 3.30), (12.0, 3.30), (13.0, 3.30), (14.0, 3.30), (15.0, 3.30), (16.0, 3.30), (17.0, 3.30), (18.0, 3.30), (19.0, 3.30), (20.0, 3.30.

Reference

1. Starfield AM, Bleloch AL (1986) Building models for conservation and wildlife. MacMillian, New York (Chapter 2)

Chapter 32
Nicholson–Bailey Host–Parasite Interaction

You knot of Mouth-Friends:..Most smiling, smooth, detested Parasites.

(Shakespeare, Timon of Athens, 1607)

32.1 Nicholson–Bailey Host–Parasitoid Model

In Chap. 26 we modeled the spread of a parasitic infection in an insect population of two life stages. The focus of that model was the spread of the infection. Therefore, we did not pay any explicit attention to the fate of the parasitoid. In this chapter however, we model explicitly the interactions between the host and the parasitoid populations. Rather than setting up our model in terms of population sizes, we specify host–parasitoid interactions in terms of population densities. What follows here closely mirrors prior modeling by Ederstein-Keshet [1] and Brown and Rothery [2].

In order to model the host–parasitoid interactions, we abstract away from the fact that only specific lifecycle stages exhibit those interactions. After you worked through this chapter, you may want to refine the model to account for the fact that, for example, adult parasitoids lay their eggs in the pupae of hosts, but not in the eggs of their hosts or with the larvae or adults.

Denote, respectively, H(t) and P(t) as the host and parasitoid densities in time period t, and F(H(t), P(t)) as the fraction of hosts that is not parasitized. Then

$$H(t+1) = \lambda * H(t) * F(H(t), P(t)) \tag{32.1}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_32,

$$P(t+1) = C * H(t) * [1 - F(H(t), P(t))] \tag{32.2}$$

where $\lambda(H(t))$ is the host growth rate and C is the parasitoid fecundity.

Let us assume that the fraction of hosts that become parasitized depends on the density-dependent rate of encounter of parasitoids and hosts. Encounters occur randomly, allowing us to invoke the law of mass action that we already discussed in Chaps. 6 and 21. Accordingly, the number of encounters of hosts HE, with parasitoids is

$$HE(t) = A * H(t) * P(t) \tag{32.3}$$

where A is the searching efficiency of the parasitoids.

Unlike the models of the spread of a disease from an infected to a nonimmune population, subsequent encounters of individuals in the two populations do not alter the rate at which parasitoids are propagated. Therefore, we need to modify the law of mass action to account for the fact that only the first encounter of hosts and parasitoids is significant in propagating the parasitoid. Once a host carries the parasitoid's eggs, subsequent encounters with parasitoids will not change the number of parasitoid progeny that hatch from the host. We need only to distinguish between hosts that had no encounter and hosts that had at least one encounter with parasitoids.

The Poisson distribution describes the occurrence of such discrete, random events as encounters of hosts and parasitoids. We can make use of the Poisson probability distribution to calculate the probability that there is no attack of parasitoids on a host within a certain time period. In general, therefore

$$P(X) = \frac{EXP\left(-\frac{HE(t)}{H(t)}\right)\left(\frac{HE(t)}{H(t)}\right)^X}{X!} \tag{32.4}$$

is the probability of X attacks. This probability depends on the average number of attacks in the given time interval, HE/H. From Eq. (32.3) we know

$$HE(t)/P(t) = A * P(t) \tag{32.5}$$

Thus, for zero attacks by the parasitoids, Eq. (32.4) yields?

$$\begin{aligned} P(0) &= \frac{EXP(-A * P(t))(A * P(t))^0}{0!} = \frac{EXP(-A * P(t)) * 1}{1} \\ &= EXP(-A * P(t)) \end{aligned} \tag{32.6}$$

Equations (32.1) and (32.2) can therefore be re-written as

$$H(t+1) = H(t) * \lambda * EXP(-A * P(t)) \tag{32.7}$$

$$P(t+1) = C * H(t) * [1 - EXP(-A * P(t))] \tag{32.8}$$

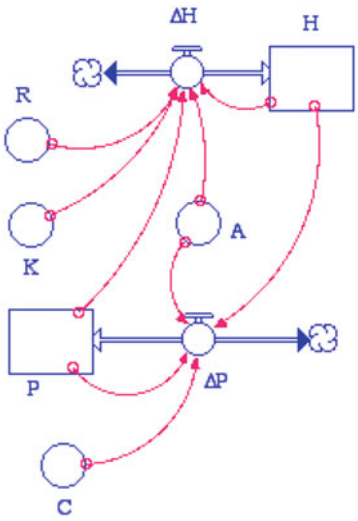

Fig. 32.1

Let us also assume that without parasitoids, the hosts will grow toward a carrying capacity K set by the environment. To capture growth of the host population up to a density $H(t) = K$ and decline of the host population for $H(t) > K$, we replace in Eq. (32.7) the growth rate $\lambda(H(t))$ with

$$\lambda = \mathrm{EXP}\left(R * \left(1 - \frac{H(t)}{K}\right)\right) \tag{32.9}$$

where R is the maximum host growth rate. Thus, the equation governing the size of the host population in time $t+1$ becomes

$$H(t+1) = H(t) * \mathrm{EXP}\left(R * \left(1 - \frac{H(t)}{K}\right) - A * P(t)\right) \tag{32.10}$$

and after subtracting the respective state variables in time period t from Eqs. (32.8) and (32.10), we have a set of differential equations that capture the change of host and parasitoid densities from time period t to $t+1$:

$$\Delta H(t) = H(t) * \mathrm{EXP}\left(R * \left(1 - \frac{H(t)}{K}\right) - A * P(t)\right) - H(t) \tag{32.11}$$

$$\Delta P(t) = C * H(t) * [1 - \mathrm{EXP}(-A * P(t))] - P(t) \tag{32.12}$$

The complete STELLA model is shown in Fig. 32.1. We can now see the dynamics that it exhibits. These equations describing changes in the host and parasitoid densities can yield a variety of results, from the production of steady-state

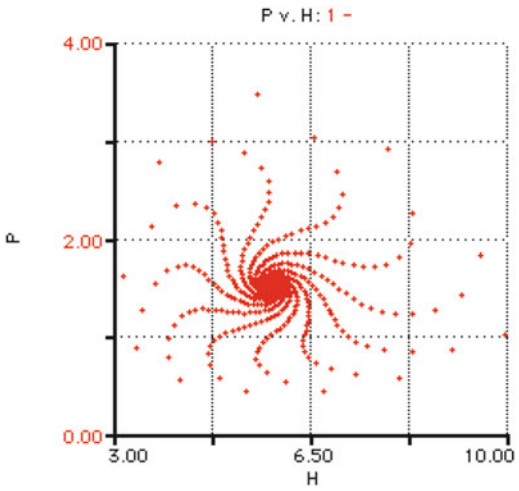

Fig. 32.2

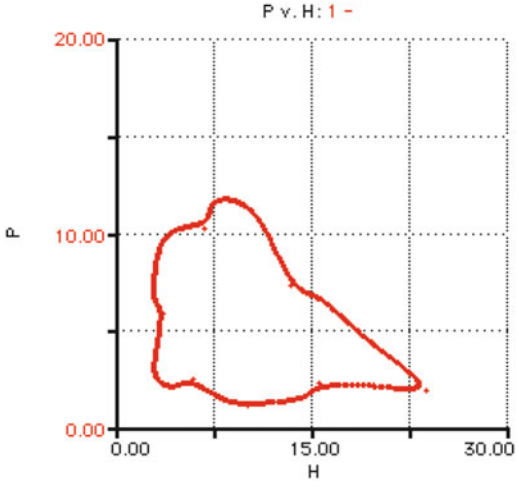

Fig. 32.3

conditions for the host and parasitoid, to their lock in a limit cycle, to chaos. Chaotic system behavior is discussed in more detail in Part VII of the book.

The result in Figs. 32.2, 32.3, and 32.4 is the product of the parameter choices and initial conditions in Table 32.1, and a DT = 1.

Try reducing the DT. For the graphs above it is set at one. A smaller DT fetches a completely different answer. What is going on here? Is the DT of 1.00 required by the host or the parasitoid? The Nicholson–Bailey model views t = 1 as one generation and all the dynamics for one DT go on inside that time period of one. It is as though

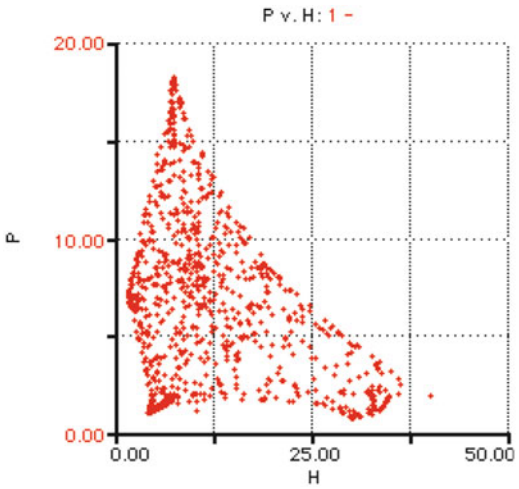

Fig. 32.4

Table 32.1 Parameter choices and initial conditions (DT = 1)

Graph	Description	R	A	K	H(t = 0)	P(t = 0)
1	Steady state	0.50	0.20	14.5	10.00	1.00
2	Limit cycle	2.00	0.20	21.5	10.00	1.00
3	Chaos	2.65	0.20	25.0	10.00	1.00

the whole new generation of the two populations is formed just before the beginning of that generation. So in the sense of this model, a DT less than one has no meaning. Do a sensitivity analysis on the initial values of H and P, on R, A, and K.

We have modeled in this chapter one type of species interaction that is almost exclusively found among insects. Typically, both the parasitoid and host have a number of lifecycle stages—eggs, larvae, pupae, and adults—and their interaction is limited to a subset of these. Can you modify the model to account for the fact that it is typically only the larvae of the host that get parasitized by adult parasitoids? How does this disaggregation of the parasitoid and host population affect your results? Can you find parameters and initial values that generate alternatively steady state, limit cycles, or chaos? What is the appropriate DT to use here and how are the results affected by its choice?

32.2 Nicholson–Bailey Host–Parasitoid Model Equations

$H(t) = H(t - dt) + (\Delta H) * dt$
INIT H = 10

INFLOWS:
$\Delta H = H*EXP(R*(1-H/K)-A*P)-H$
$P(t) = P(t - dt) + (\Delta P) * dt$
INIT $P = 1$

INFLOWS:
$\Delta P = C*H*(1-EXP(-A*P))-P$
$A = .2$
$C = 1$
$K = 14.5$
$R = .5$

References

1. Ederstein-Keshet L (1988) Mathematical models in biology. Random House, New York, pp 79–85
2. Brown D, Rothery P (1993) Models in biology: mathematics statistics and computing. Wiley, New York, pp 399–406

Chapter 33
Diseased and Healthy Immigrating Insects

And migrant tribes these fruitful shorelands hail.

(Joel Barlow, 1807, The Columbiad, ii. 178)

33.1 Immigrating Insects Model

This chapter builds on the models developed in Chap. 26 by distinguishing two cohorts of a population infected with a disease. The two populations modeled here are insects that suffer from a disease that increases mortality for the infected nymphs and adults and also decreases their egg-laying rate. Unlike the previous chapters we assume here two populations of insects, living in two fields. One of the fields has generally better living conditions than the other, although the current year's carrying capacities are randomly generated and there is some overlap in the ranges within which the carrying capacity fluctuates. Carrying capacity has a direct effect on birth rates.

The carrying capacities of the two fields are defined as

$$\begin{aligned} \text{K1} = {} & \text{IF CARRY\_R1} > .666 \text{ THEN } 2 \\ & \text{ELSE IF CARRY R1} < .333 \text{ THEN } .5 \\ & \text{ELSE } 1 \end{aligned} \tag{33.1}$$

and

$$\begin{aligned} \text{K2} = {} & \text{IF CARRY R2} > .666 \text{ THEN } 4 \\ & \text{ELSE IF CARRY R2} < .333 \text{ THEN } 1 \\ & \text{ELSE } 2 \end{aligned} \tag{33.2}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_33,

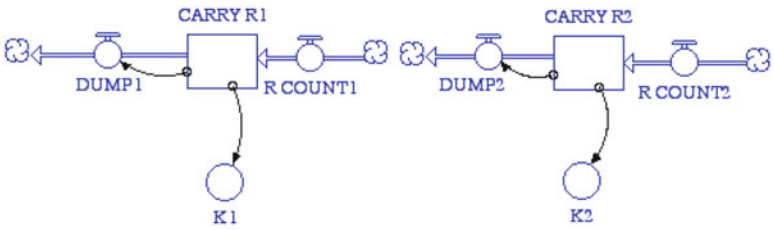

Fig. 33.1

respectively, with CARRY R1 and CARRY R2 as random numbers between 0 and 1. These random numbers are calculated in the following module with

$$\begin{aligned} \text{R COUNT1} &= \text{IF MOD}(\text{TIME}, 52) \\ &= 0 \text{ THEN RANDOM}(0, 1)/\text{DT ELSE } 0 \end{aligned} \tag{33.3}$$

and

$$\begin{aligned} \text{R COUNT2} &= \text{IF MOD}(\text{TIME}, 52) \\ &= 0 \text{ THEN RANDOM}(0, 1)/\text{DT ELSE } 0. \end{aligned} \tag{33.4}$$

As before, we make use of the MOD function here to set up a recurring counter. Note that with some DT values, whose fractional representation does not have n^2 in the denominator, STELLA rounds the remainder in the MOD function; so the re-starting values of R COUNT1 and R COUNT2 for each new year are not exactly zero (Fig. 33.1).

When over-crowding develops, healthy adult insects leave their home field and join the other population. Furthermore, it is assumed that 10 % of healthy adults migrate under all circumstances. Changes in population sizes are no longer only dependent on births and on deaths but additionally on migration.

The model is composed of the following additional modules (Figs. 33.2, 33.3, and 33.4). The first captures the population dynamics of healthy insects in the first field. The structure and workings of this module are analogous to the ones outlined in Chap. 26 with the additional feature of migration from and to that region.

The second module (Fig. 33.3) is set up to calculate the change in nymph and adult population in field 1 that are affected by the disease.

A virtually identical second set of these modules capture the dynamics of the populations in field 2. Parameters relevant to both healthy and diseased insects in both fields are calculated in the following modules (Fig. 33.3). They include

- A calculation of the total number of adults in each fields, ALL ADULTS 1 and ALL ADULTS 2;
- The ratio of the total number of adults in each region to the carrying capacity of the respective region, FRXNL CAP1 and FRXNL CAP2;
- Experimental maturation times for healthy and diseased insects, Tx1 H, Tx1 D;
- Model survival fractions MxSF H, MxSF D;

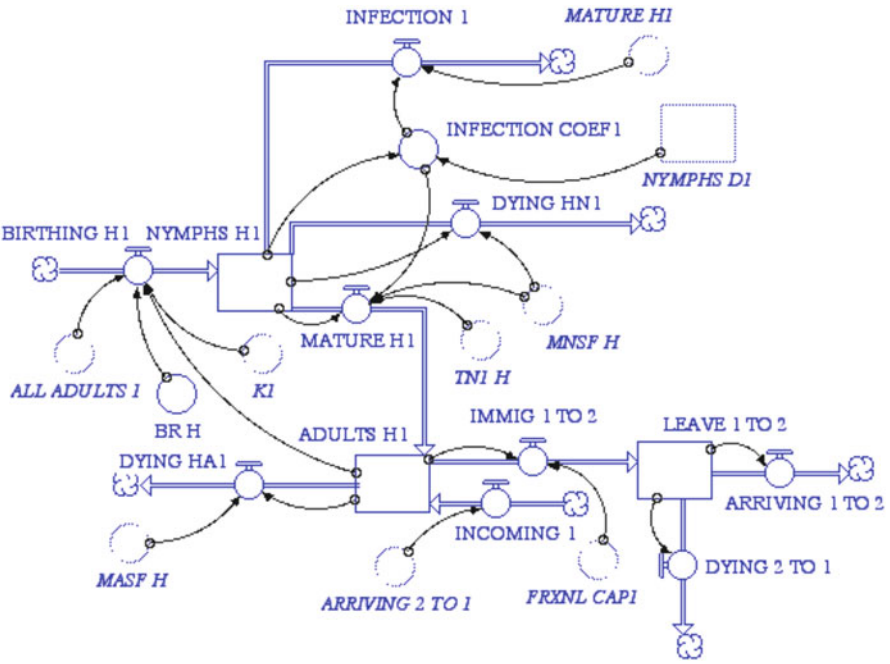

Fig. 33.2

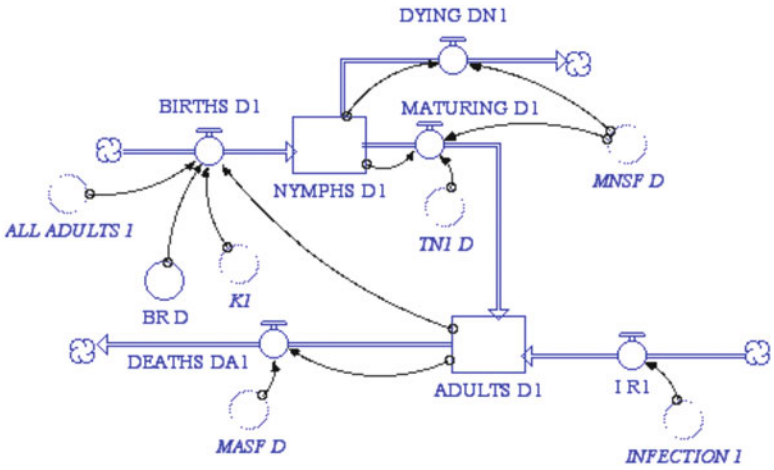

Fig. 33.3

- Experimental laying rates BR H, BR D;
- Experimental daily adult survival fractions per stage, ExSF H , ExSF D.

The model rates are calculated from the experimental data using the functions discussed in Chaps. 26 and 27.

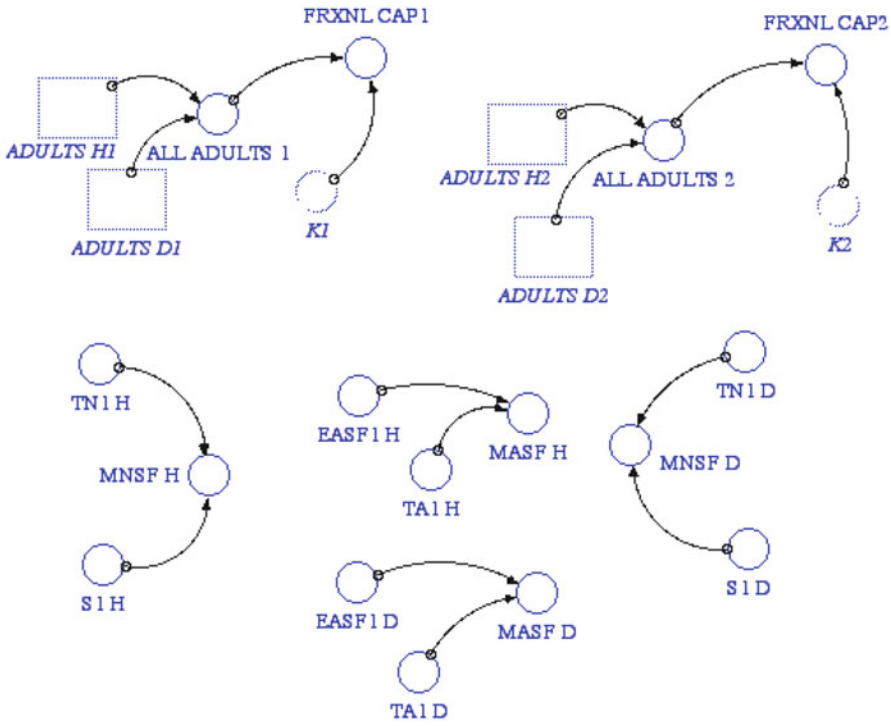

Fig. 33.4

Figure 33.5 shows the combined result of the population dynamics due to natural increases and deaths as well as migration. Note how ensuing runs with the same parameters and initial conditions are significantly different. Why is this difference occurring? The diseased insects are at a "disadvantage" here. What would you change within realm of the biologically likely, to favor the diseased population? Is it possible that by studying insect population dynamics from an ecological perspective we can provide a more useful means for biological control? Can you implement such a control in the model?

Over the long run, the population of each field is clearly responding to changes in the local carrying capacity. Both fields take a while to build up numbers from the low start, 0.1 adults each of healthy and diseased. Surprisingly, in the run we have graphed neither field's total population hugs the carrying capacity very well, but in other runs we made it sometimes did. Clearly there are other factors at work limiting population besides carrying capacity. While we've required that 10 % of healthy adults migrate, we aren't seeing the diseased population expand to fill that gap. Run the model repeatedly and note the differences. Introduce additional factors such as seasonality into the population model.

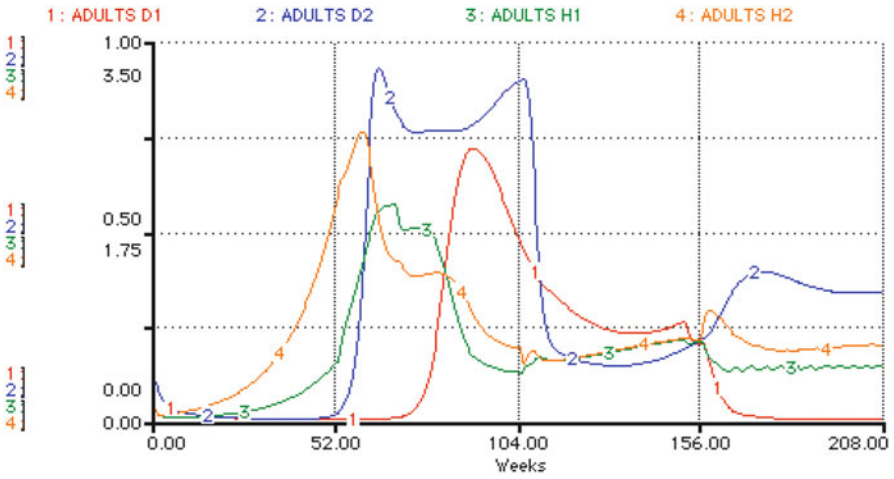

Fig. 33.5

33.2 Immigrating Insects Model Equations

ADULTS_D1(t) = ADULTS_D1(t − dt) + (MATURING_D1 + I_R1 − DEATHS_DA1) ∗ dt
INIT ADULTS_D1 = .1 {Initial diseased adults.}
INFLOWS:
MATURING_D1 = MNSF_D∗NYMPHS_D1/TN1_D {Individuals per Time Period}
I_R1 = INFECTION_1 {Individuals per Time Period}
OUTFLOWS:
DEATHS_DA1 = ADULTS_D1∗(1-MASF_D)/DT {Individuals per Time Period}
ADULTS_D2(t) = ADULTS_D2(t − dt) + (MATURING_D2 + I_R2 − DYING_DA2) ∗ dt
INIT ADULTS_D2 = .1 {Initial diseased adults.}
INFLOWS:
MATURING_D2 = MNSF_D∗NYMPHS_D2/TN1_D {Individuals per Time Period}
I_R2 = INFECTION_2 {Individuals per Time Period}
OUTFLOWS:
DYING_DA2 = ADULTS_D2∗(1 − MASF_D)/DT {Individuals per Time Period}
ADULTS_H1(t) = ADULTS_H1(t − dt) + (MATURE_H1 + INCOMING_1 − DYING_HA1 − IMMIG_1_TO_2) ∗ dt
INIT ADULTS_H1 = .1 {Initial healthy adults}
INFLOWS:
MATURE_H1 = MNSF_H∗NYMPHS_H1/TN1_H∗(1−INFECTION_COEF1)
INCOMING_1 = ARRIVING_2_TO_1 {Individuals per Time Period}

OUTFLOWS:
DYING_HA1 = ADULTS_H1*(1 − MASF_H)/DT {Individuals per Time Period}
IMMIG_1_TO_2 = IF ADULTS_H1 − (.1*ADULTS_H1 + .9*FRXNL_CAP1) > 0 THEN (.1*ADULTS_H1 + .9*FRXNL_CAP1) ELSE IF ADULTS_H1 > 0 THEN ADULTS_H1 ELSE 0 {Individuals per Time Period; Only healthy adults migrate. At least ten percent of the healthy adults always migrate. Under the noted conditions the 10% healthy and an additional fraction of the healthy adults based empirically on the total number of adults also migrate. Note well the order of the nested IF statement; the first one is checked first and if the condition holds the first statement is executed and the program goes no further. Otherwise all the adults flee. This same statement is also true of the adults in the other field.}
ADULTS_H2(t) = ADULTS_H2(t − dt) + (MATURE_H2 + INCOMING_2 − DYING_HA2 − IMMIG_2_TO_1) * dt
INIT ADULTS_H2 = .1 {Initial healthy adults}
INFLOWS:
MATURE_H2 = MNSF_H*NYMPHS_H2*(1−INFECTION_COEF2)/TN1_H {Individuals per Time Period}
INCOMING_2 = ARRIVING_1_TO_2 {Individuals per Time Period}
OUTFLOWS:
DYING_HA2 = ADULTS_H2*(1 − MASF_H)/DT {Individuals per Time Period}
IMMIG_2_TO_1 = IF ADULTS_H2 − (.1*ADULTS_H2 + .9*FRXNL_CAP2) > 0 THEN (.1*ADULTS_H2 + .9*FRXNL_CAP2) ELSE IF ADULTS_H2 > 0 THEN ADULTS_H2 ELSE 0 {Individuals per Time Period}
CARRY_R1(t) = CARRY_R1(t − dt) + (R_COUNT1 − DUMP1) * dt
INIT CARRY_R1 = 0
INFLOWS:
R_COUNT1 = IF (time MOD 52) = 0 THEN RANDOM(0,1)/DT ELSE 0
OUTFLOWS:
DUMP1 = IF (time MOD 52) = 0 THEN CARRY_R1/DT ELSE 0 {Insures a new number between 0-1 each integer time step.}
CARRY_R2(t) = CARRY_R2(t − dt) + (R_COUNT2 − DUMP2) * dt
INIT CARRY_R2 = 0 {See note in Carry_R.}
INFLOWS:
R_COUNT2 = IF (time MOD 52) = 0 THEN RANDOM(0,1)/DT ELSE 0
OUTFLOWS:
DUMP2 = IF (time MOD 52) = 0 THEN CARRY_R2/DT ELSE 0
LEAVE_1_TO_2(t) = LEAVE_1_TO_2(t − dt) + (IMMIG_1_TO_2 − DYING_2_TO_1 − ARRIVING_1_TO_2) * dt
INIT LEAVE_1_TO_2 = 0
INFLOWS:
IMMIG_1_TO_2 = IF ADULTS_H1 - (.1*ADULTS_H1 + .9*FRXNL_CAP1) > 0 THEN (.1*ADULTS_H1 + .9*FRXNL_CAP1) ELSE IF ADULTS_H1 > 0 THEN ADULTS_H1 ELSE 0 {Individuals per Time Period; Only healthy adults migrate. At least ten percent of the healthy adults always migrate.

Under the noted conditions the 10% healthy and an additional fraction of the healthy adults based empirically on the total number of adults also migrate. Note well the order of the nested IF statement; the first one is checked first and if the condition holds the first statement is executed and the program goes no further. Otherwise all the adults flee. This same statement is also true of the adults in the other field.}

OUTFLOWS:

DYING_2_TO_1 = .25*LEAVE_1_TO_2 {Individuals per Time Period}

ARRIVING_1_TO_2 = .75*LEAVE_1_TO_2 {Individuals per Time Period}

LEAVE_2_TO_1(t) = LEAVE_2_TO_1(t − dt) + (IMMIG_2_TO_1 − DYING_2TO1 − ARRIVING_2_TO_1) * dt

INIT LEAVE_2_TO_1 = 0

INFLOWS:

IMMIG_2_TO_1 = IF ADULTS_H2 − (.1*ADULTS_H2 + .9*FRXNL_CAP2) > 0 THEN (.1*ADULTS_H2 + .9*FRXNL_CAP2) ELSE IF ADULTS_H2 > 0 THEN ADULTS_H2 ELSE 0 {Individuals per Time Period}

OUTFLOWS:

DYING_2TO1 = .25*LEAVE_2_TO_1 {Individuals per Time Period}

ARRIVING_2_TO_1 = .75*LEAVE_2_TO_1 {Individuals per Time Period}

NYMPHS_D1(t) = NYMPHS_D1(t − dt) + (BIRTHS_D1 − DYING_DN1 − MATURING_D1) * dt

INIT NYMPHS_D1 = 0 {Initial diseased eggs}

INFLOWS:

BIRTHS_D1 = IF (K1-ALL_ADULTS_1) > 0 THEN BR_D*ADULTS_D1 ELSE 0 {Individuals per Time Period}

OUTFLOWS:

DYING_DN1 = (1 − MNSF_D)*NYMPHS_D1/DT {Individuals per Time Period}

MATURING_D1 = MNSF_D*NYMPHS_D1/TN1_D {Individuals per Time Period}

NYMPHS_D2(t) = NYMPHS_D2(t − dt) + (BIRTHS_D2 − DYING_DN2 − MATURING_D2) * dt

INIT NYMPHS_D2 = 0 {Initial diseased eggs}

INFLOWS:

BIRTHS_D2 = IF (K2-ALL_ADULTS_2) > 0 THEN BR_D*ADULTS_D2 ELSE 0 {Individuals per Time Period}

OUTFLOWS:

DYING_DN2 = (1 − MNSF_D)*NYMPHS_D2/DT {Individuals per Time Period}

MATURING_D2 = MNSF_D*NYMPHS_D2/TN1_D {Individuals per Time Period}

NYMPHS_H1(t) = NYMPHS_H1(t − dt) + (BIRTHING_H1 − DYING_HN1 − MATURE_H1 − INFECTION_1) * dt

INIT NYMPHS_H1 = 0 {Initial Healthy eggs}

INFLOWS:
BIRTHING_H1 = IF (K1 − ALL_ADULTS_1) > 0 THEN BR_H∗ADULTS_H1 ELSE 0 {Individuals per Time Period}
OUTFLOWS:
DYING_HN1 = (1 − MNSF_H)∗NYMPHS_H1/DT
MATURE_H1 = MNSF_H∗NYMPHS_H1/TN1_H∗(1−INFECTION_COEF1)
INFECTION_1 = INFECTION_COEF1∗MATURE_H1 {Individuals per Time Period}
NYMPHS_H2(t) = NYMPHS_H2(t − dt) + (BIRTHS_H2 − DYING_HN2 − MATURE_H2 − INFECTION_2) ∗ dt
INIT NYMPHS_H2 = 0 {Initial Healthy eggs}
INFLOWS:
BIRTHS_H2 = IF (K2-ALL_ADULTS_2) > 0 THEN BR_H∗ADULTS_H2 ELSE 0 {Individuals per Time Period}
OUTFLOWS:
DYING_HN2 = (1 − MNSF_H)∗NYMPHS_H2/DT {Individuals per Time Period}
MATURE_H2 = MNSF_H∗NYMPHS_H2∗(1-INFECTION_COEF2)/TN1_H {Individuals per Time Period}
INFECTION_2 = INFECTION_COEF2∗MATURE_H2 {Individuals per Time Period}
ALL_ADULTS_1 = ADULTS_H1+ADULTS_D1
ALL_ADULTS_2 = ADULTS_H2+ADULTS_D2
BR_D = .35 {Experimental laying rate. DISEASED EGGS PER ADULT PER DAY.}
BR_H = 0.75
DOCUMENT: NYMPH BIRTH RATE, NYMPHS PER DAY PER ADULT
EASF1_D = .65 {Experimental daily diseased adult survival fraction per stage, dimensionless.}
EASF1_H = 0.8
FRXNL_CAP1 = ALL_ADULTS_1/K1
DOCUMENT: This fraction is the degree to which both healthy and diseased adults have reached their carrying capacity.
FRXNL_CAP2 = ALL_ADULTS_2/K2
INFECTION_COEF1 = 1 − EXP(−.3∗NYMPHS_H1∗NYMPHS_D1) {Constructed function giving the desired 0 to 1 probability.}
INFECTION_COEF2 = 1 − EXP(−.3∗NYMPHS_H2∗NYMPHS_D2) {Constructed function giving the desired 0 to 1 probability.}
K1 = IF CARRY_R1 > .666 THEN 2 ELSE IF CARRY_R1 < .333 THEN .5 ELSE 1 {This is the carrying capacity of the area the insects area}
K2 = IF CARRY_R2 > .666 THEN 4 ELSE IF CARRY_R2 < .333 THEN 1 ELSE 2 {This is the carrying capacity of the area the insects area}
MASF_D = EXP(LN(EASF1_D)/TA1_D∗DT)
MASF_H = EXP(LN(EASF1_H)/TA1_H∗DT)
MNSF_D = EXP(LN(S1_D)/TN1_D∗DT)
MNSF_H = EXP(LN(S1_H)/TN1_H∗DT)

S1_D = .5 {Experimental diseased egg survival fraction, dimensionless, per stage. Stage = 1/F1, i.e., 30 eggs per 100 eggs survive each 1/F1 days, as noted in the experiment.}
S1_H = .7 {Experimental egg survival fraction, dimensionless, per stage. Stage = 1/F1, i.e., 70 eggs per 100 eggs survive each 1/F1 days, as noted in the experiment.}
TA1_D = 1
TA1_H = 1
TN1_D = 5
TN1_H = 5

Chapter 34
Two-Species Colonization Model

Thou eternal fugitive, Hovering over all that live.

(1847 Emerson Poems, Ode to Beauty)

34.1 Basic Colonization Model

In Chap. 5 we have modeled spatial dynamics in a rather abstract way. Let us take up the issue of spatial dynamics in this chapter and deal specifically with the competition between two species for space. We will begin this chapter with a simple version of this model, then introduce disturbances on the physical landscape and observe the implications for population dynamics.

Assume you are the manager of forestland on which two species of trees can grow [1]. Both species are able to colonize open patches of land. The colonization coefficient C is different for each of the species. One of the species has a higher ability to colonize, but after colonization took place it is easily outcompeted by the other species. Call the species that loses in competition the INFERIOR or "fugitive" species, and the other one the SUPERIOR species. Both species have an extinction rate, E, which we assume—only to keep things simple—to be the same for each species. The constant E is multiplied by the number of patches occupied by a species, to obtain the extinction rate of that species. Once extinction from a patch on the landscape takes place, the area that was previously covered by a particular species is converted into an open patch.

There are three state variables of this system: One state variable for the amount of open land that can be colonized, and one each for the land occupied by a Superior and Inferior species. In the model we normalize the total habitable forest area

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_34,

$$\text{TOTAL} = 1 \tag{34.1}$$

and express the land that is open for colonization and the land that is colonized as a fraction of the total.

Additional to the three state variables, there are three main driving forces for the dynamics of this system that you must consider. One is the colonization of, and competition for, patches of land by the two tree species—the conversion of empty space to either INFERIOR or SUPERIOR species. The second is the removal of species through extinction. Finally, there is the conversion of INFERIOR species space through encounters with the SUPERIOR ones.

To calculate the number of patches that become occupied by inferior species, we multiply the colonization rate of the inferior species, CI, by the product of open patches and the area occupied by the inferior species. From this product we must subtract the loss of inferior species due to extinction at a rate EI,

$$\text{I COLONIZES} = \text{CI} * \text{INFERIOR} * \text{OPEN} - \text{EI} * \text{INFERIOR} \tag{34.2}$$

to obtain the Inferior colonization rate.

Multiplying the two state variables INFERIOR and OPEN with each other and with the colonization coefficient C to calculate the conversion of uncolonized to colonized patches is analogous to the way in which chemists calculate the product of two chemical reactions. We have made use of this idea, for example, in our epidemiology models in Chap. 21 and the host–parasite model of Chap. 32.

Again, an analogous application of the law of mass action yields the colonization rate by SUPERIOR species

$$\text{S COLONIZES} = \text{CS} * \text{SUPERIOR} * \text{OPEN} - \text{ES} * \text{SUPERIOR} \tag{34.3}$$

where CS is the colonization rate of the superior species and ES is the extinction rate of the superior species. The rate at which the superior species replaces the inferior one is

$$\text{S DISPLACES I} = \text{CS} * \text{INFERIOR} * \text{SUPERIOR} \tag{34.4}$$

No resistance to this displacement is offered by the Inferior species.

The relationships between the superior and the inferior species are listed below. In general, for different extinction rates EI and ES for the inferior and superior species, the inferior species is defined by the following inequality:

$$\text{CS}/\text{ES} < \text{CI}/\text{EI} \tag{34.5}$$

which can be derived by setting the derivatives in the exchange Eqs. (34.2), (34.3), and (34.4) equal to zero. The inequality means that either species must have a relatively low extinction rate or a relatively high colonization rate in order to stay in this area.

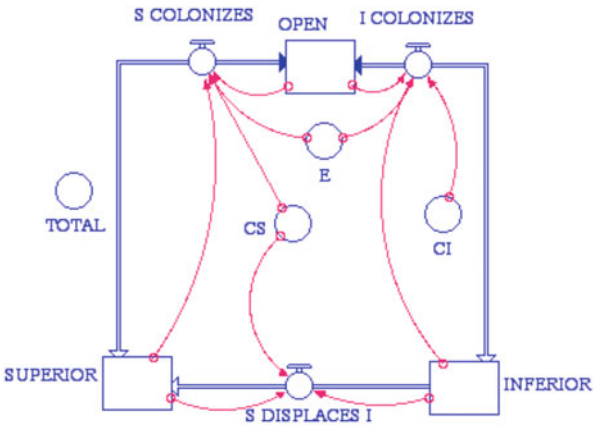

Fig. 34.1

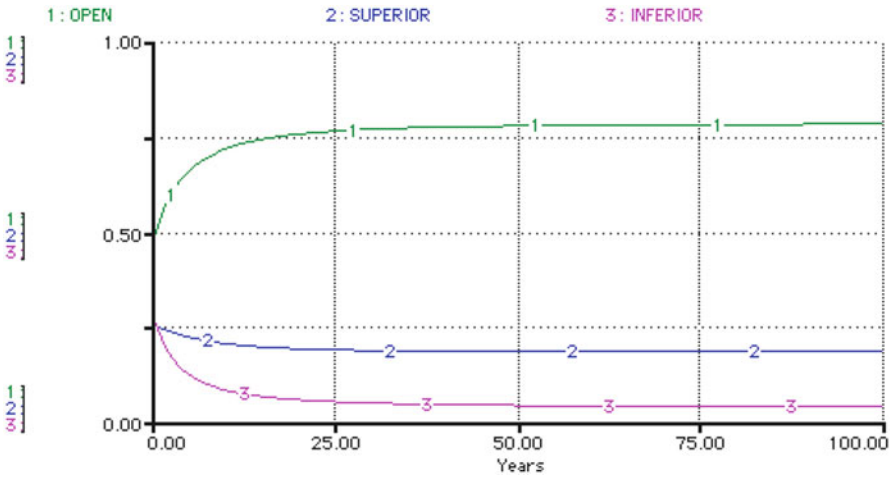

Fig. 34.2

The model is shown in Fig. 34.1. Run this model for initial values of INFERIOR = 0.256 and SUPERIOR = 0.25, the colonization rates CS = 0.55 and CI = 0.7, and extinction rate ES = EI = E = 0.45.

The results of the model (Fig. 34.2) show that the system soon reaches a steady state in which most of the land is open patches. With ecological succession, the inferior species is becoming increasingly replaced by superior species, leading to a dominance of the superior species in the steady state.

The stocks in the model shown in Fig. 34.1 could be construed as stocks of biomass with the "OPEN" stock as the remaining potential. Then crimping the "TOTAL" would be reducing the total potential biomass. Can you set up the model

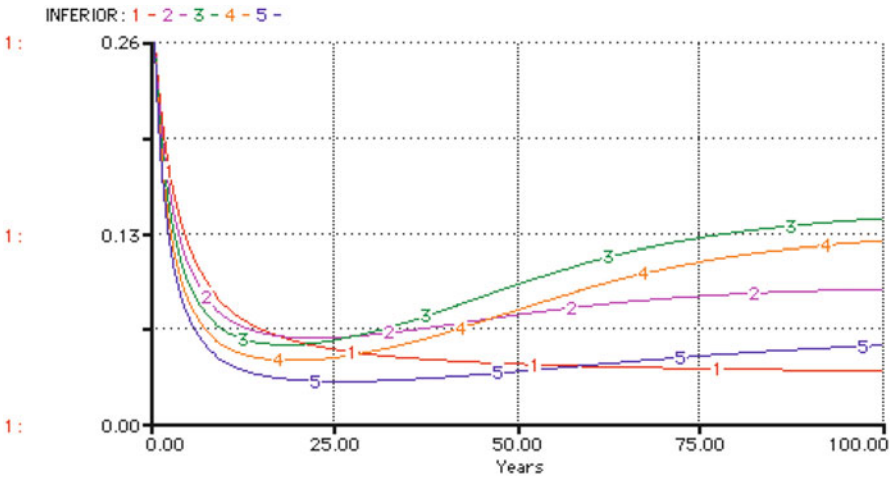

Fig. 34.3

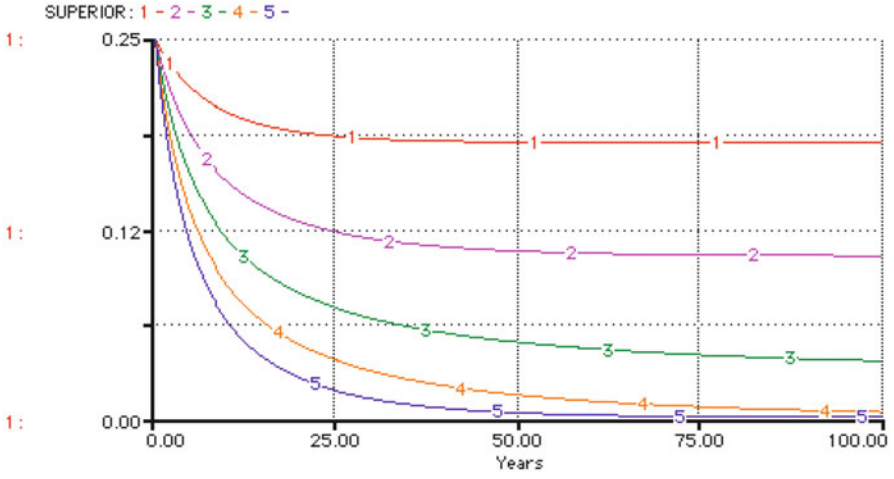

Fig. 34.4

to investigate the implications of habitat loss for the colonization process? One way to set up that model is to specify a series of sensitivity runs for which the total land available for colonization becomes smaller with each run. The results in Figs. 34.3, 34.4, and 34.5 are derived for five runs with the total habitable area declining from 1 to 0.7. Can you explain why the fraction of open patches temporarily increases in some of the runs, and then declines again?

A certain range of reduced habitable space actually favors the dominance of the inferior species. This result is due to the higher colonization rate of the inferior

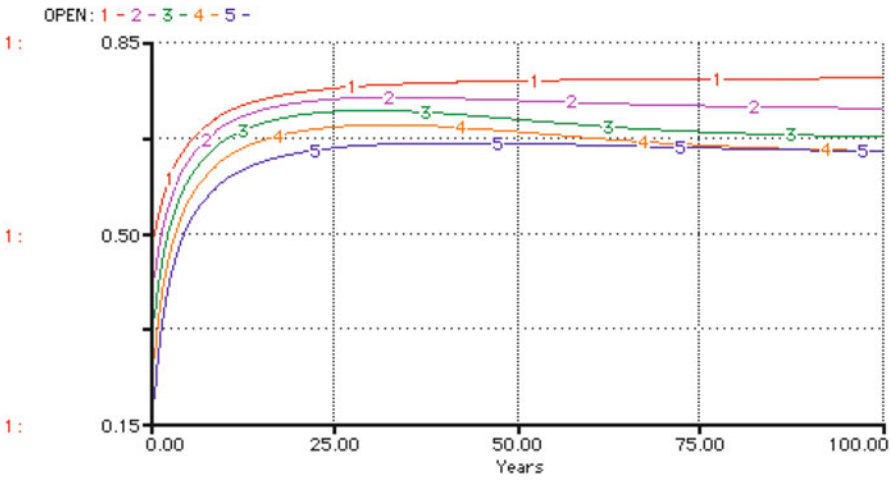

Fig. 34.5

species, a fugitive-like species, as compared to the superior species. The inferior species is able to recover more quickly from their sudden "extinction" from certain patches and, thus, may "evade the bulldozers" more easily whereas the chances that the superior species will get caught in a patch that is being destroyed is greater because of their slower movement.

Matthias use this graph instead of the previous one: As habitable space is decreased in the example we show above, the superior population declines and the inferior species declines. When the space has been preempted down to the level E/CS or below, the superior species disappears and the population begins to decline from its maximum steady-state level. When the habitable space is reduced to the level E/CI, both species disappear from the landscape.

Still another interesting possibility exists. Let the extinction coefficient E be the same for both species. Under a special range of choice of CS, E, and CN, only the superior species exists at the steady state, until the fractional level of habitat has been reduced below E∗CN/(CS)^2, in the cases where this term is less than one. Our model indicates a subtle nuance [2]: under those conditions where the fractional level of the habitat stands between 1 and E∗Cn/(Cs), only repeated disturbance can possibly keep the inferior species in existence on this patch. Then those disturbances must not be too severe or they may speed the demise of the inferior species. The range of frequency and severity of the disturbance are critical as one can apparently only learn by experiment with numerical analysis. Try to show this "window" by selecting new values for these coefficients.

A true field of science has its own unique principles. Ecology has few and the concept of a limiting territorial size may be one of those unique principles.

34.2 Basic Patch Dynamics Model Equations

INFERIOR(t) = INFERIOR(t − dt) + (I_COLONIZES − S_DISPLACES_I) ∗ dt
INIT INFERIOR = .256

INFLOWS:
I_COLONIZES = CI∗INFERIOR∗OPEN−E∗INFERIOR
OUTFLOWS:
S_DISPLACES_I = CS∗INFERIOR∗SUPERIOR
OPEN(t) = OPEN(t − dt) + (− S_COLONIZES − I_COLONIZES) ∗ dt
INIT OPEN = TOTAL − SUPERIOR − INFERIOR

OUTFLOWS:
S_COLONIZES = CS∗SUPERIOR∗OPEN−E∗SUPERIOR
I_COLONIZES = CI∗INFERIOR∗OPEN−E∗INFERIOR
SUPERIOR(t) = SUPERIOR(t − dt) + (S_DISPLACES_I + S_COLONIZES) ∗ dt
INIT SUPERIOR = .25

INFLOWS:
S_DISPLACES_I = CS∗INFERIOR∗SUPERIOR
S_COLONIZES = CS∗SUPERIOR∗OPEN−E∗SUPERIOR
CI = .75
CS = .55
E = .45
TOTAL = 1

34.3 Two-Species Colonization Model with Fire

Real-world ecosystems are not maintained in permanent steady state. Rather, natural events such as insect pest outbreaks or forest fires may lead to significant changes in those systems, "re-setting" them to a state in their early successional cycle. Let us set up the model such that forest fires occur when the forest reached a steady state and the investigate impacts of fire on the structure of the forest community. Towards this end, we first specify a new variable that measures the change in open patches

$$\text{D OPEN} = 5000 * \text{DERIVN}(\text{OPEN}, 1) \tag{34.6}$$

using the built-in function DERIVN to calculate the derivative of the state variable OPEN with respect to time. We use D OPEN to generate a random occurrence of fires,

$$\text{FIRE YEARS} = \text{IF}(\text{TIME} > 1)\text{AND}\big(\big((\text{RANDOM}(0,1))^2 > \text{ABS}(\text{D OPEN})\big)\big) \text{ THEN 1 ELSE 0} \tag{34.7}$$

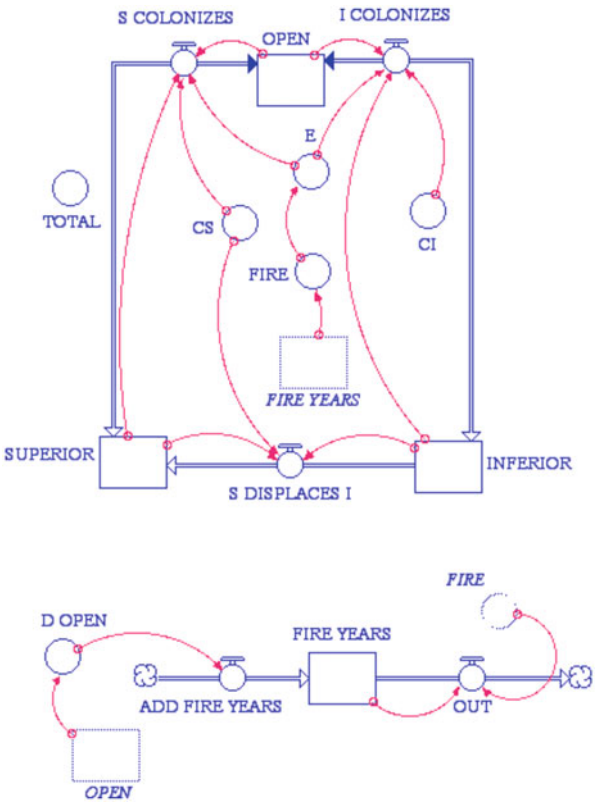

Fig. 34.6

Once fifteen such fire years accumulated, we assume a fire occurs that is large enough to affect the extinction rates of species from their colonized patches. We calculate a variable FIRE as

$$\text{FIRE} = \text{IF FIRE YEARS} = 15 \text{ THEN } 1.5 \text{ ELSE } 0 \tag{34.8}$$

and capture its impacts on the extinction of the species from a particular forest patch, by modifying the extinction rate to

$$\text{E} = 0.4 + \text{RANDOM}(.45, .65) * \text{FIRE/DT} \tag{34.9}$$

The complete model is shown in Fig. 34.6.

As the model runs, a steady state is approached (Fig. 34.7). As the steady state is reached, random fires increase the proportion of empty patches. The inferior species rapidly colonize the empty habitable patches after a fire occurs. As the inferior species colonizes an increasing portion of the space, the conditions for the superior

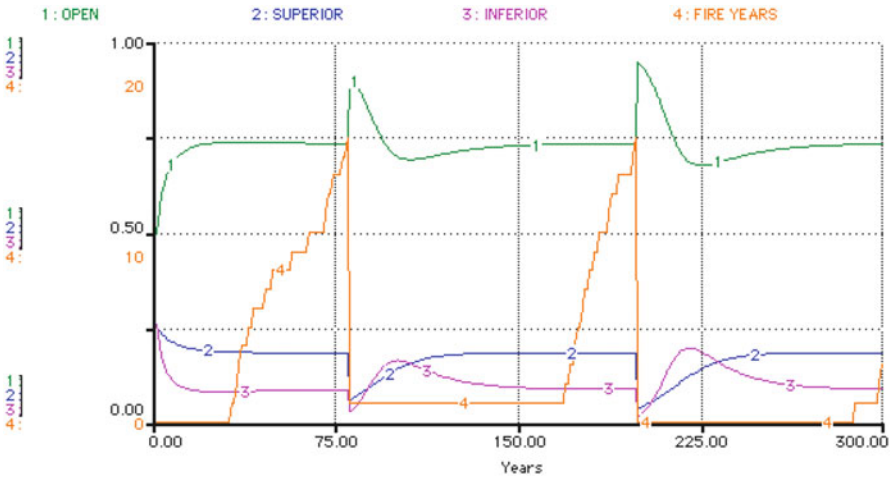

Fig. 34.7

species become more favorable, leading to a replacement of inferior species. In the long run, the system returns to a near steady state, until fire breaks out again. Choose alternative parameter values to modify the severity of the fire. Figure 34.7 (DT = 1/4) shows the case of fire leading to a temporary dominance of inferior species over superior ones.

Investigate the combined effects of habitat loss discussed in the previous section of this chapter and forest fires through a series of sensitivity runs. What are the implications of your findings for ecosystem management?

34.4 Patch Dynamics With Fire Model Equations

FIRE_YEARS(t) = FIRE_YEARS(t − dt) + (ADD_FIRE_YEARS − OUT) ∗ dt
INIT FIRE_YEARS = 0
INFLOWS:
ADD_FIRE_YEARS = IF (TIME > 1) AND (((RANDOM(0,1))^2 > ABS (D_OPEN))) THEN 1 ELSE 0
OUTFLOWS:
OUT = IF FIRE= 1.5 THEN FIRE_YEARS/dt ELSE 0
INFERIOR(t) = INFERIOR(t − dt) + (I_COLONIZES − S_DISPLACES_I) ∗ dt
INIT INFERIOR = 0.256
INFLOWS:
I_COLONIZES = CI∗INFERIOR∗OPEN−E∗INFERIOR
OUTFLOWS:
S_DISPLACES_I = CS∗INFERIOR∗SUPERIOR
OPEN(t) = OPEN(t − dt) + (−S_COLONIZES − I_COLONIZES) ∗ dt
INIT OPEN = TOTAL − SUPERIOR − INFERIOR

OUTFLOWS:
S_COLONIZES = CS*SUPERIOR*OPEN−E*SUPERIOR
I_COLONIZES = CI*INFERIOR*OPEN−E*INFERIOR
SUPERIOR(t) = SUPERIOR(t − dt) + (S_DISPLACES_I + S_COLONIZES) * dt
INIT SUPERIOR = 0.25
INFLOWS:
S_DISPLACES_I = CS*INFERIOR*SUPERIOR
S_COLONIZES = CS*SUPERIOR*OPEN−E*SUPERIOR
CI = 0.75
CS = 0.55
D_OPEN = 5000*DERIVN(Open,1)
E = .45 + RANDOM(.45,.65)*FIRE/dt
FIRE = IF FIRE_YEARS=15 THEN 1.5 ELSE 0
TOTAL = 1

34.5 Landscape and Patch Dynamics

Let us expand on the model of the previous section that captured a disturbance (fire) that occurs in landscape near steady state and converted some fraction of occupied patches to empty patches. The collection of patches in the model of the previous section forms a *region*, and the modeling in this section will group the regions into an interacting set. By creating a larger landscape, made of multiple smaller regions on the landscapes, can we achieve a more steady distribution of these populations of competitors? If we consider each smaller landscape a region of this larger landscape, how can we model the movement of species between regions, and what effect will this movement have on the equilibrium of the total landscape?

Duplicate the model of the previous chapter to generate a 3×3 grid to study the effects of adding spatial dimensions to this model. Each patch has some specific characteristics, which help to simulate a somewhat diverse landscape. One characteristic is the region's "affinity for fire" which is contained as a coefficient in the ∂OPEN variable. This coefficient determines how close a region is to steady-state, and therefore how quickly it accumulates fire days.

Assume that only the inferior species move between regions, due to their higher colonization rate. Inferior species only colonize in adjacent regions when the region is not at equilibrium (i.e., after a disturbance). We have assumed here that the colonization of inferiors in adjacent regions resembles a seeding process; that is, inferiors need not leave their own region to colonize in an adjacent region, but the success of their seeds in adjacent regions is dependent on a larger amount of open space than normal. These inferiors cannot colonize adjacent open space instantaneously either—there is a lag time in years associated with moving between regions, which we have called COL YEARS and arbitrarily set at 10. So the idea is to have INFERIORS only as interregional colonizers and it takes them ten years

1	2	3
4	5	6
7	8	9

Fig. 34.8

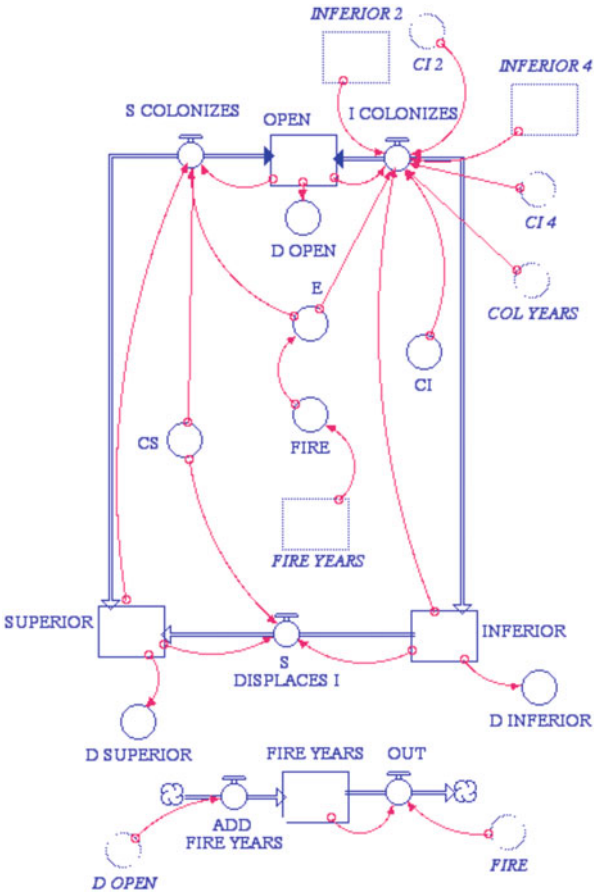

Fig. 34.9

to effect their desired boundary crossing. Between two adjacent regions, that desire is based on how many inferiors there are in the home region and how many inferiors there are in the region to be colonized.

The general layout of our model is shown in Fig. 34.8. The STELLA diagram of Fig. 34.9 shows, as an example, how to calculate the amount of the three types of space for the upper left-hand cell (Region 1) of the 3×3 grid of nine interconnected models. Only the inferior species in the adjacent regions to the right and below can

Fig. 34.10

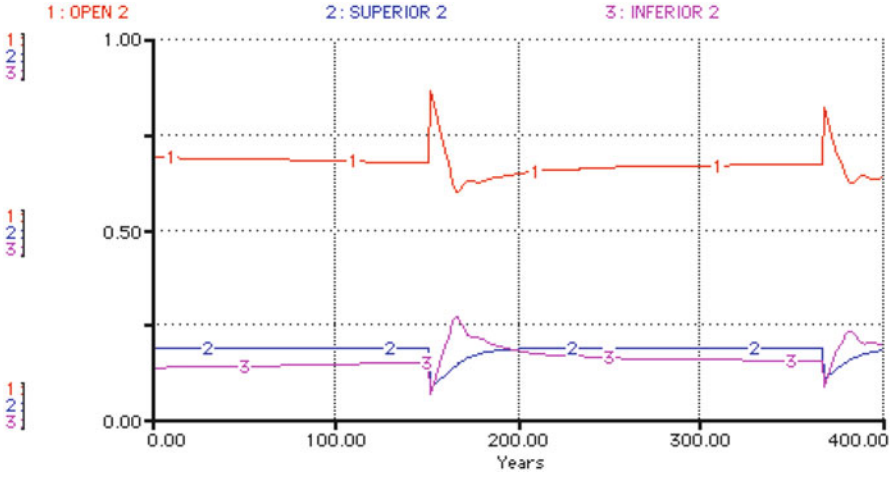

Fig. 34.11

disperse their seeds into that portion of the habitable area. Therefore, we need to calculate I COLONIZES as a function of the presence of inferior species in those two adjacent regions—INFERIOR 2 and INFERIOR 4.

With these rules, we can begin to model the effects of patch disturbances on the overall landscape, and how the landscape as a whole responds to this single-patch disturbance. This result is part of a body of theoretical ecosystem speculation (see, e.g., [3–7]).

Our results in Figs. 34.10, 34.11, 34.12, 34.13, and 34.14 are shown for the first two of the nine patches, for the open area—unsmoothed and smoothed with the

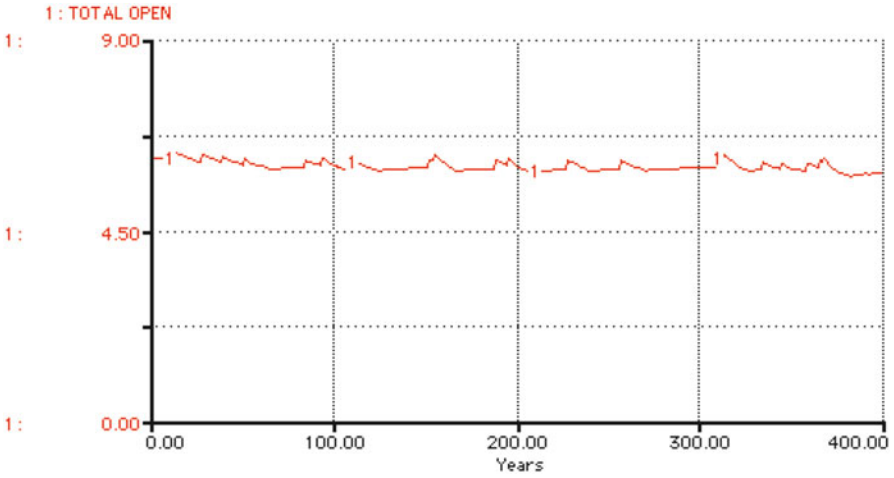

Fig. 34.12

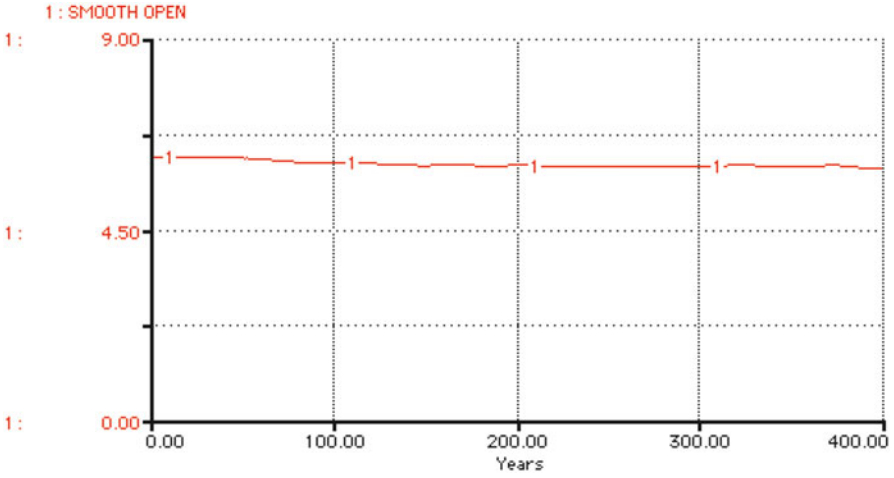

Fig. 34.13

built-in function SMTH1—and the area colonized by the inferior species over the entire landscape.

When comparing the results of this mosaic-type landscape with the single-patch model (with the same parameter values) we find that the disturbances in individual regions are damped by both the greater number of total regions, and by the colonization of adjacent inferiors. The fact that these disturbances occur out-of-phase helps to give the total landscape more stability than that of any individual patch. This result

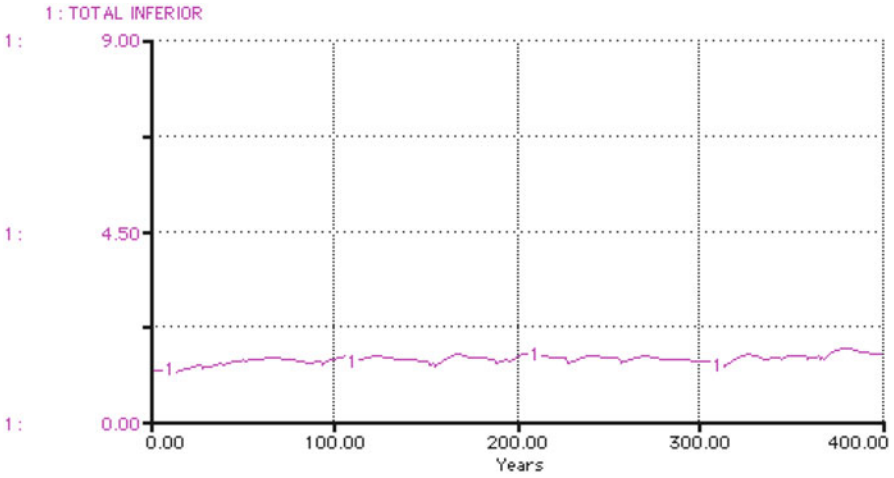

Fig. 34.14

may be even more evident if we allow each fire to destroy a whole patch, instead of only a fraction of the patch. If this were the case, the disturbed patch would have to rely solely on the species from adjacent patches to rebuild its population. It would be interesting to see if the total landscape could maintain a somewhat steady population over time, even when whole patches are being eliminated at various times. Set up the model to investigate this case.

The realization that extreme variation of species presence at the patch (cell) level results in their steady presence at the aggregate or landscape level seems to be a unique principle for the science of ecology.

Modify the model of this section to allow the separate patches to have different colonization rates. How will this affect the overall dynamics of the system? Is it possible for the system as a whole to support inferior life in patches which would not be able to support this species on their own?

In this chapter, we captured the competition for space. In the following four chapters we will model a different type of species interaction as the one modeled here. There, we will concentrate on predator–prey interactions. The first of these models deals with algae and herbivore, using hypothetical data. The second is more elaborate. It is built on real data for grass carp populations. The third predator–prey model shown below concentrates on population management methods that are built on predator–prey interactions. Finally, in Chap. 35 we will return to the issue of spatial dynamics already discussed here in the context of spatial competition.

34.6 Landscape and Patch Dynamics Model Equations

FIREYEARS_2(t) = FIREYEARS_2(t − dt) + (ADD__FIRE_YEARS_2 − OUT_2) ∗ dt
INIT FIREYEARS_2 = 0
INFLOWS:
ADD__FIRE_YEARS_2 = if (time > 1) and (((random(0,1))^2 > ABS (D_OPEN_2))) then 1 else 0
OUTFLOWS:
OUT_2 = if FIRE_2=1 then FIREYEARS_2 else 0
FIREYEARS_7(t) = FIREYEARS_7(t − dt) + (ADD__FIRE_YEARS_7 − OUT_7) ∗ dt
INIT FIREYEARS_7 = 0
INFLOWS:
ADD__FIRE_YEARS_7 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_7))) then 1 else 0
OUTFLOWS:
OUT_7 = if FIRE_7=1 then FIREYEARS_7 else 0
FIRE_YEARS(t) = FIRE_YEARS(t − dt) + (ADD__FIRE_YEARS − OUT) ∗ dt
INIT FIRE_YEARS = 0
INFLOWS:
ADD__FIRE_YEARS = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN))) then 1 else 0
OUTFLOWS:
OUT = if FIRE=1 then FIRE_YEARS else 0
FIRE_YEARSS_3(t) = FIRE_YEARSS_3(t − dt) + (ADD__FIRE_YEARS_3 − OUT_3) ∗ dt
INIT FIRE_YEARSS_3 = 0
INFLOWS:
ADD__FIRE_YEARS_3 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_3))) then 1 else 0
OUTFLOWS:
OUT_3 = if FIRE_3=1 then FIRE_YEARSS_3 else 0
FIRE_YEARSS_9(t) = FIRE_YEARSS_9(t − dt) + (ADD__FIRE_YEARS_9 − OUT_9) ∗ dt
INIT FIRE_YEARSS_9 = 0
INFLOWS:
ADD__FIRE_YEARS_9 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_9))) then 1 else 0
OUTFLOWS:
OUT_9 = if FIRE_9=1 then FIRE_YEARSS_9 else 0
FIRE_YEARS_4(t) = FIRE_YEARS_4(t − dt) + (ADD__FIRE_YEARS_4 − OUT_4_4) ∗ dt
INIT FIRE_YEARS_4 = 0

INFLOWS:
ADD__FIRE_YEARS_4 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_4))) then 1 else 0
OUTFLOWS:
OUT_4_4 = if FIRE_4=1 then FIRE_YEARS_4 else 0
FIRE_YEARS_5(t) = FIRE_YEARS_5(t − dt) + (ADD__FIRE_YEARS_5 − OUT_5) ∗ dt
INIT FIRE_YEARS_5 = 0
INFLOWS:
ADD__FIRE_YEARS_5 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_5))) then 1 else 0
OUTFLOWS:
OUT_5 = if FIRE_5=1 then FIRE_YEARS_5 else 0
FIRE_YEARS_6(t) = FIRE_YEARS_6(t − dt) + (ADD__FIRE_YEARS_6 − OUT_6) ∗ dt
INIT FIRE_YEARS_6 = 0
INFLOWS:
ADD__FIRE_YEARS_6 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_6))) then 1 else 0
OUTFLOWS:
OUT_6 = if FIRE_6=1 then FIRE_YEARS_6 else 0
FIRE_YEARS_8(t) = FIRE_YEARS_8(t − dt) + (ADD__FIRE_YEARS_8 − OUT_8) ∗ dt
INIT FIRE_YEARS_8 = 0
INFLOWS:
ADD__FIRE_YEARS_8 = if (time > 1) and (((random(0,1))^2 > ABS(D_OPEN_8))) then 1 else 0
OUTFLOWS:
OUT_8 = if FIRE_8=1 then FIRE_YEARS_8 else 0
INFERIOR(t) = INFERIOR(t − dt) + (I_COLONIZES − S__DISPLACES_I) ∗ dt
INIT INFERIOR = 0.1304
INFLOWS:
I_COLONIZES = (CI∗INFERIOR∗OPEN−E∗INFERIOR) + (DELAY(ABS((.69−OPEN))/2∗CI_2∗INFERIOR_2∗OPEN,COL_YEARS))+ (DELAY(ABS((.69−OPEN))/2∗CI_4∗INFERIOR_4∗OPEN,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I = CS∗INFERIOR∗SUPERIOR
INFERIOR_2(t) = INFERIOR_2(t − dt) + (I__COLONIZES__2 − S__DISPLACES_I_2) ∗ dt
INIT INFERIOR_2 = 0.1304
INFLOWS:
I__COLONIZES__2 = (CI_2∗INFERIOR_2∗OPEN_2−E_2∗INFERIOR_2) + (DELAY (ABS((.69−OPEN_2))/2.5∗CI∗INFERIOR∗OPEN_2 ,COL_YEARS)) + (DELAY (ABS((.69−OPEN_2))/2.5∗CI_3∗INFERIOR_3∗OPEN_2 ,COL_YEARS))+ (DELAY (ABS((.69−OPEN_2))/2.5∗CI_5∗INFERIOR_5∗OPEN_2,COL_YEARS))

```
OUTFLOWS:
S__DISPLACES_I_2 = CS_2*INFERIOR_2*SUPERIOR_2
INFERIOR_3(t) = INFERIOR_3(t - dt) + (I__COLONIZES__3 - S__DISPLACES_I_3) * dt
INIT INFERIOR_3 = 0.1304
INFLOWS:
I__COLONIZES__3 = (CI_3*INFERIOR_3*OPEN_3-E_3*INFERIOR_3)+ (DELAY(ABS((.69-OPEN_3))/2*CI_2*INFERIOR_2*OPEN_3,COL_YEARS))+ (DELAY(ABS((.69-OPEN_3))/2*CI_6*INFERIOR_6*OPEN_3,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_3 = CS_3*INFERIOR_3*SUPERIOR_3
INFERIOR_4(t) = INFERIOR_4(t - dt) + (I__COLONIZES__4 - S__DISPLACES_I_4) * dt
INIT INFERIOR_4 = 0.1304
INFLOWS:
I__COLONIZES__4 = (CI_4*INFERIOR_4*OPEN_4-E_4*INFERIOR_4)+ (DELAY (ABS((.69-OPEN_4))/2.5*CI*INFERIOR*OPEN_4,COL_YEARS))+ (DELAY (ABS((.69-OPEN_4))/2.5*CI_5*INFERIOR_5*OPEN_4,COL_YEARS))+ (DELAY(ABS((.69-OPEN_4))/2.5*CI_7*INFERIOR_7*OPEN_4,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_4 = CS_4*INFERIOR_4*SUPERIOR_4
INFERIOR_5(t) = INFERIOR_5(t - dt) + (I__COLONIZES__5 - S__DISPLACES_I_5) * dt
INIT INFERIOR_5 = 0.1304
INFLOWS:
I__COLONIZES__5 = (CI_5*INFERIOR_5*OPEN_5-E_5*INFERIOR_5) + (DELAY(ABS((.69-OPEN_5))/3*CI_2*INFERIOR_2*OPEN_5,COL_YEARS))+ (DELAY(ABS((.69-OPEN_5))/3*CI_4*INFERIOR_4*OPEN_5,COL_YEARS))+ (DELAY(ABS((.69-OPEN_5))/3*CI_6*INFERIOR_6*OPEN_5,COL_YEARS))+ (DELAY(ABS((.69-OPEN_5))/3*CI_8*INFERIOR_8*OPEN_5,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_5 = CS_5*INFERIOR_5*SUPERIOR_5
INFERIOR_6(t) = INFERIOR_6(t - dt) + (I__COLONIZES__6 - S__DISPLACES_I_6) * dt
INIT INFERIOR_6 = 0.1304
INFLOWS:
I__COLONIZES__6 = (CI_6*INFERIOR_6*OPEN_6-E_6*INFERIOR_6) + (DELAY (ABS((.69-OPEN_6))/2.5*CI_5*INFERIOR_5*OPEN_6,COL_YEARS))+ (DELAY (ABS((.69-OPEN_6))/2.5*CI_3*INFERIOR_3*OPEN_6,COL_YEARS))+ (DELAY (ABS((.69-OPEN_6))/2.5*CI_9*INFERIOR_9*OPEN_6,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_6 = CS_6*INFERIOR_6*SUPERIOR_6
INFERIOR_7(t) = INFERIOR_7(t - dt) + (I__COLONIZES__7 - S__DISPLACES_I_7) * dt
INIT INFERIOR_7 = 0.1304
```

INFLOWS:
I__COLONIZES__7 = (CI_7∗INFERIOR_7∗OPEN_7−E_7∗INFERIOR_7) + (DELAY(ABS((.69−OPEN_7))/2∗CI_4∗INFERIOR_4∗OPEN_7,COL_YEARS))+ (DELAY(ABS((.69−OPEN_7))/2∗CI_8∗INFERIOR_8∗OPEN_7,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_7 = CS_7∗INFERIOR_7∗SUPERIOR_7
INFERIOR_8(t) = INFERIOR_8(t − dt) + (I__COLONIZES__8 − S__DISPLACES_I_8) ∗ dt
INIT INFERIOR_8 = 0.1304
INFLOWS:
I__COLONIZES__8 = (CI_8∗INFERIOR_8∗OPEN_8−E_8∗INFERIOR_8) + (DELAY (ABS((.69−OPEN_8))/2.5∗CI_5∗INFERIOR_5∗OPEN_8,COL_YEARS))+ (DELAY (ABS((.69−OPEN_8))/2.5∗CI_7∗INFERIOR_7∗OPEN_8,COL_YEARS))+ (DELAY (ABS((.69−OPEN_8))/2.5∗CI_9∗INFERIOR_9∗OPEN_8,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_8 = CS_8∗INFERIOR_8∗SUPERIOR8
INFERIOR_9(t) = INFERIOR_9(t − dt) + (I__COLONIZES__9 − S__DISPLACES_I_9) ∗ dt
INIT INFERIOR_9 = 0.1304
INFLOWS:
I__COLONIZES__9 = (CI_9∗INFERIOR_9∗OPEN_9−E_9∗INFERIOR_9) + (DELAY(ABS((.69−OPEN_9))/2∗CI_6∗INFERIOR_6∗OPEN_9,COL_YEARS))+ (DELAY(ABS((.69−OPEN_9))/2∗CI_8∗INFERIOR_8∗OPEN_9,COL_YEARS))
OUTFLOWS:
S__DISPLACES_I_9 = CS_9∗INFERIOR_9∗SUPEIOR_9
OPEN(t) = OPEN(t − dt) + (−S_COLONIZES − I_COLONIZES) ∗ dt
INIT OPEN = 1 − SUPERIOR − INFERIOR
OUTFLOWS:
S_COLONIZES = (CS∗SUPERIOR∗OPEN − E∗SUPERIOR)
I_COLONIZES = (CI∗INFERIOR∗OPEN−E∗INFERIOR) + (DELAY(ABS ((.69−OPEN))/2∗CI_2∗INFERIOR_2∗OPEN,COL_YEARS))+ (DELAY(ABS ((.69−OPEN))/2∗CI_4∗INFERIOR_4∗OPEN,COL_YEARS))
OPEN_2(t) = OPEN_2(t − dt) + (−S_COLONIZES_2 − I__COLONIZES__2) ∗ dt
INIT OPEN_2 = 1 − SUPERIOR_2 − INFERIOR_2
OUTFLOWS:
S_COLONIZES_2 = CS_2∗SUPERIOR_2∗OPEN_2 − E_2∗SUPERIOR_2
I__COLONIZES__2 = (CI_2∗INFERIOR_2∗OPEN_2−E_2∗INFERIOR_2) + (DELAY (ABS((.69−OPEN_2))/2.5∗CI∗INFERIOR∗OPEN_2 ,COL_YEARS)) + (DELAY (ABS((.69−OPEN_2))/2.5∗CI_3∗INFERIOR_3∗OPEN_2 ,COL_YEARS))+ (DELAY (ABS((.69−OPEN_2))/2.5∗CI_5∗INFERIOR_5∗OPEN_2,COL_YEARS))
OPEN_3(t) = OPEN_3(t − dt) + (−S_COLONIZES_3 − I__ COLONIZES__3) ∗ dt
INIT OPEN_3 = 1 − SUPERIOR_3 − INFERIOR_3
OUTFLOWS:
S_COLONIZES_3 = CS_3∗SUPERIOR_3∗OPEN_3 − E_3∗SUPERIOR_3

I__COLONIZES__3 = (CI_3∗INFERIOR_3∗OPEN_3−E_3∗INFERIOR_3)+
(DELAY(ABS((.69−OPEN_3))/2∗CI_2∗INFERIOR_2∗OPEN_3,COL_YEARS))+
(DELAY(ABS((.69−OPEN_3))/2∗CI_6∗INFERIOR_6∗OPEN_3,COL_YEARS))
OPEN_4(t) = OPEN_4(t − dt) + (−S_COLONIZES_4 − I__COLONIZES__4) ∗ dt
INIT OPEN_4 = 1 − SUPERIOR_4 − INFERIOR_4
OUTFLOWS:
S_COLONIZES_4 = CS_4∗SUPERIOR_4∗OPEN_4 − E_4∗SUPERIOR_4
I__COLONIZES__4 = (CI_4∗INFERIOR_4∗OPEN_4−E_4∗INFERIOR_4)+ (DELAY
(ABS((.69−OPEN_4))/2.5∗CI∗INFERIOR∗OPEN_4,COL_YEARS))+ (DELAY(ABS
((.69−OPEN_4))/2.5∗CI_5∗INFERIOR_5∗OPEN_4,COL_YEARS))+ (DELAY(ABS
((.69−OPEN_4))/2.5∗CI_7∗INFERIOR_7∗OPEN_4,COL_YEARS))
OPEN_5(t) = OPEN_5(t − dt) + (−S_COLONIZES_5 − I__COLONIZES__5) ∗ dt
INIT OPEN_5 = 1 − SUPERIOR_5 − INFERIOR_5
OUTFLOWS:
S_COLONIZES_5 = CS_5∗SUPERIOR_5∗OPEN_5 − E_5∗SUPERIOR_5
I__COLONIZES__5 = (CI_5∗INFERIOR_5∗OPEN_5−E_5∗INFERIOR_5) +
(DELAY(ABS((.69−OPEN_5))/3∗CI_2∗INFERIOR_2∗OPEN_5,COL_YEARS))+
(DELAY(ABS((.69−OPEN_5))/3∗CI_4∗INFERIOR_4∗OPEN_5,COL_YEARS))+
(DELAY(ABS((.69−OPEN_5))/3∗CI_6∗INFERIOR_6∗OPEN_5,COL_YEARS))+
(DELAY(ABS((.69−OPEN_5))/3∗CI_8∗INFERIOR_8∗OPEN_5,COL_YEARS))
OPEN_6(t) = OPEN_6(t − dt) + (−S_COLONIZES_6 − I__COLONIZES__6) ∗ dt
INIT OPEN_6 = 1 − SUPERIOR_6 − INFERIOR_6
OUTFLOWS:
S_COLONIZES_6 = CS_6∗SUPERIOR_6∗OPEN_6 − E_6∗SUPERIOR_6
I__COLONIZES__6 = (CI_6∗INFERIOR_6∗OPEN_6−E_6∗INFERIOR_6) + (DELAY
(ABS((.69−OPEN_6))/2.5∗CI_5∗INFERIOR_5∗OPEN_6,COL_YEARS))+ (DELAY
(ABS((.69−OPEN_6))/2.5∗CI_3∗INFERIOR_3∗OPEN_6,COL_YEARS))+ (DELAY
(ABS((.69−OPEN_6))/2.5∗CI_9∗INFERIOR_9∗OPEN_6,COL_YEARS))
OPEN_7(t) = OPEN_7(t − dt) + (−S_COLONIZES_7 − I__COLONIZES__7) ∗ dt
INIT OPEN_7 = 1 − SUPERIOR_7 − INFERIOR_7
OUTFLOWS:
S_COLONIZES_7 = CS_7∗SUPERIOR_7∗OPEN_7 − E_7∗SUPERIOR_7
I__COLONIZES__7 = (CI_7∗INFERIOR_7∗OPEN_7−E_7∗INFERIOR_7) +
(DELAY(ABS((.69−OPEN_7))/2∗CI_4∗INFERIOR_4∗OPEN_7,COL_YEARS))+
(DELAY(ABS((.69−OPEN_7))/2∗CI_8∗INFERIOR_8∗OPEN_7,COL_YEARS))
OPEN_8(t) = OPEN_8(t − dt) + (−S_COLONIZES_8 − I__COLONIZES__8) ∗ dt
INIT OPEN_8 = 1 − SUPERIOR8 − INFERIOR_8
OUTFLOWS:
S_COLONIZES_8 = CS_8∗SUPERIOR8∗OPEN_8 − E_8∗SUPERIOR8
I__COLONIZES__8 = (CI_8∗INFERIOR_8∗OPEN_8−E_8∗INFERIOR_8) + (DELAY
(ABS((.69−OPEN_8))/2.5∗CI_5∗INFERIOR_5∗OPEN_8,COL_YEARS))+ (DELAY
(ABS((.69−OPEN_8))/2.5∗CI_7∗INFERIOR_7∗OPEN_8,COL_YEARS))+ (DELAY
(ABS((.69−OPEN_8))/2.5∗CI_9∗INFERIOR_9∗OPEN_8,COL_YEARS))
OPEN_9(t) = OPEN_9(t − dt) + (−S_COLONIZES_9 − I__COLONIZES__9) ∗ dt
INIT OPEN_9 = 1 − SUPEIOR_9 − INFERIOR_9

OUTFLOWS:
S_COLONIZES_9 = CS_9∗SUPEIOR_9∗OPEN_9 − E_9∗SUPEIOR_9
I__COLONIZES__9 = (CI_9∗INFERIOR_9∗OPEN_9−E_9∗INFERIOR_9) + (DELAY(ABS((.69−OPEN_9))/2∗CI_6∗INFERIOR_6∗OPEN_9,COL_YEARS))+ (DELAY(ABS((.69−OPEN_9))/2∗CI_8∗INFERIOR_8∗OPEN_9,COL_YEARS))
SUPEIOR_9(t) = SUPEIOR_9(t − dt) + (S__DISPLACES_I_9 + S_COLONIZES_9) ∗ dt
INIT SUPEIOR_9 = 0.1818
INFLOWS:
S__DISPLACES_I_9 = CS_9∗INFERIOR_9∗SUPEIOR_9
S_COLONIZES_9 = CS_9∗SUPEIOR_9∗OPEN_9 − E_9∗SUPEIOR_9
SUPERIOR(t) = SUPERIOR(t − dt) + (S__DISPLACES_I + S_COLONIZES) ∗ dt
INIT SUPERIOR = 0.1818
INFLOWS:
S__DISPLACES_I = CS∗INFERIOR∗SUPERIOR
S_COLONIZES = (CS∗SUPERIOR∗OPEN − E∗SUPERIOR)
SUPERIOR8(t) = SUPERIOR8(t − dt) + (S__DISPLACES_I_8 + S_COLONIZES_8) ∗ dt
INIT SUPERIOR8 = 0.1818
INFLOWS:
S__DISPLACES_I_8 = CS_8∗INFERIOR_8∗SUPERIOR8
S_COLONIZES_8 = CS_8∗SUPERIOR8∗OPEN_8 − E_8∗SUPERIOR8
SUPERIOR_2(t) = SUPERIOR_2(t − dt) + (S__DISPLACES_I_2 + S_COLONIZES_2) ∗ dt
INIT SUPERIOR_2 = 0.1818
INFLOWS:
S__DISPLACES_I_2 = CS_2∗INFERIOR_2∗SUPERIOR_2
S_COLONIZES_2 = CS_2∗SUPERIOR_2∗OPEN_2 − E_2∗SUPERIOR_2
SUPERIOR_3(t) = SUPERIOR_3(t − dt) + (S__DISPLACES_I_3 + S_COLONIZES_3) ∗ dt
INIT SUPERIOR_3 = 0.1818
INFLOWS:
S__DISPLACES_I_3 = CS_3∗INFERIOR_3∗SUPERIOR_3
S_COLONIZES_3 = CS_3∗SUPERIOR_3∗OPEN_3 − E_3∗SUPERIOR_3
SUPERIOR_4(t) = SUPERIOR_4(t − dt) + (S__DISPLACES_I_4 + S_COLONIZES_4) ∗ dt
INIT SUPERIOR_4 = 0.1818
INFLOWS:
S__DISPLACES_I_4 = CS_4∗INFERIOR_4∗SUPERIOR_4
S_COLONIZES_4 = CS_4∗SUPERIOR_4∗OPEN_4 − E_4∗SUPERIOR_4
SUPERIOR_5(t) = SUPERIOR_5(t − dt) + (S__DISPLACES_I_5 + S_COLONIZES_5) ∗ dt
INIT SUPERIOR_5 = 0.1818
INFLOWS:
S__DISPLACES_I_5 = CS_5∗INFERIOR_5∗SUPERIOR_5
S_COLONIZES_5 = CS_5∗SUPERIOR_5∗OPEN_5 − E_5∗SUPERIOR_5

SUPERIOR_6(t) = SUPERIOR_6(t − dt) + (S__DISPLACES_I_6 + S_COLONIZES_6) ∗ dt
INIT SUPERIOR_6 = 0.1818
INFLOWS:
S__DISPLACES_I_6 = CS_6∗INFERIOR_6∗SUPERIOR_6
S_COLONIZES_6 = CS_6∗SUPERIOR_6∗OPEN_6 − E_6∗SUPERIOR_6
SUPERIOR_7(t) = SUPERIOR_7(t − dt) + (S__DISPLACES_I_7 + S_COLONIZES_7) ∗ dt
INIT SUPERIOR_7 = 0.1818
INFLOWS:
S__DISPLACES_I_7 = CS_7∗INFERIOR_7∗SUPERIOR_7
S_COLONIZES_7 = CS_7∗SUPERIOR_7∗OPEN_7 − E_7∗SUPERIOR_7
CI = 0.8
CI_2 = 0.8
CI_3 = 0.8
CI_4 = 0.8
CI_5 = 0.8
CI_6 = 0.8
CI_7 = 0.8
CI_8 = 0.8
CI_9 = 0.8
COL_YEARS = 10
CS = 0.55
CS_2 = 0.55
CS_3 = 0.55
CS_4 = 0.55
CS_5 = 0.55
CS_6 = 0.55
CS_7 = 0.55
CS_8 = 0.55
CS_9 = 0.55
D_INFERIOR = 100∗DERIVN(INFERIOR,1)
D_INFERIOR_2 = 100∗DERIVN(INFERIOR_2,1)
D_INFERIOR_3 = 100∗DERIVN(INFERIOR_3,1)
D_INFERIOR_4 = 100∗DERIVN(INFERIOR_4,1)
D_INFERIOR_5 = 100∗DERIVN(INFERIOR_5,1)
D_INFERIOR_6 = 100∗DERIVN(INFERIOR_6,1)
D_INFERIOR_7 = 100∗DERIVN(INFERIOR_7,1)
D_INFERIOR_8 = 100∗DERIVN(INFERIOR_8,1)
D_INFERIOR_9 = 100∗DERIVN(INFERIOR_9,1)
D_OPEN = 5000∗DERIVN(OPEN,1)
D_OPEN_2 = 10000∗DERIVN(OPEN_2,1)
D_OPEN_3 = 15000∗DERIVN(OPEN_3,1)
D_OPEN_4 = 20000∗DERIVN(OPEN_4,1)
D_OPEN_5 = 5000∗DERIVN(OPEN_5,1)

```
D_OPEN_6 = 10000*DERIVN(OPEN_6,1)
D_OPEN_7 = 15000*DERIVN(OPEN_7,1)
D_OPEN_8 = 20000*DERIVN(OPEN_8,1)
D_OPEN_9 = 5000*DERIVN(OPEN_9,1)
D_SUPERIOR = 100*DERIVN(SUPERIOR,1)
D_SUPERIOR_2 = 100*DERIVN(SUPERIOR_2,1)
D_SUPERIOR_3 = 100*DERIVN(SUPERIOR_3,1)
D_SUPERIOR_4 = 100*DERIVN(SUPERIOR_4,1)
D_SUPERIOR_5 = 100*DERIVN(SUPERIOR_5,1)
D_SUPERIOR_6 = 100*DERIVN(SUPERIOR_6,1)
D_SUPERIOR_7 = 100*DERIVN(SUPERIOR_7,1)
D_SUPERIOR_8 = 100*DERIVN(SUPERIOR8,1)
D_SUPERIOR_9 = 100*DERIVN(SUPEIOR_9,1)
E = .45 + random(0.45,0.65)*FIRE
E_2 = .45 + random(0.45,0.65)*FIRE_2
E_3 = .45 + random(0.45,0.65)*FIRE_3
E_4 = .45 + random(0.45,0.65)*FIRE_4
E_5 = .45 + random(0.45,0.65)*FIRE_5
E_6 = .45 + random(0.45,0.65)*FIRE_6
E_7 = .45 + random(0.45,0.65)*FIRE_7
E_8 = .45 + random(0.45,0.65)*FIRE_8
E_9 = .45 + random(0.45,0.65)*FIRE_9
FIRE = If  FIRE_YEARS=15 then 1 else 0
FIRE_2 = If  FIREYEARS_2=15 then 1 else 0
FIRE_3 = If  FIRE_YEARSS_3=15 then 1 else 0
FIRE_4 = IF TIME = 10 THEN 1 ELSE If  FIRE_YEARS_4=15 then 1 else 0
FIRE_5 = If  FIRE_YEARS_5=15 then 1 else 0
FIRE_6 = If  FIRE_YEARS_6=15 then 1 else 0
FIRE_7 = If  FIREYEARS_7=15 then 1 else 0
FIRE_8 = IF TIME = 25 THEN 1 ELSE IF FIRE_YEARS_8=15 THEN 1 ELSE 0
FIRE_9 = If  FIRE_YEARSS_9=15 then 1 else 0
SMOOTH_OPEN = SMTH1(TOTAL_OPEN,40)
TOTAL_INFERIOR = INFERIOR+INFERIOR_2+INFERIOR_3+INFERIOR_4
   +INFERIOR_5+INFERIOR_6+INFERIOR_7+INFERIOR_8+INFERIOR_9
TOTAL_OPEN = OPEN+OPEN_2+OPEN_3+OPEN_4+OPEN_5+OPEN_6+OPEN_7
   +OPEN_8+OPEN_9
```

References

1. Nee S, May R (1992) Dynamics of metapopulations: habitat destruction and competitive coexistence. J Anim Ecol 61(1):37–40
2. Hannon B (1996) Adding to the Nee-May model. J Anim Ecol 65(6):850

3. Johnson A (1993) Spatiotemporal hierarchies in ecological theory and modeling. In: Second international conference on integrating geographic inform. systems and environmental modeling, Breckenridge
4. Loucks O, Plumb-Mentjes M, Rodgers D (1985) Gap processes and large-scale disturbances in sand prairies. In: The ecology of natural disturbances and patch dynamics, Academic, Waltham, pp 71–83
5. Lavorel S, Gardner R, O'Neill R (1993) Analysis of patterns in hierarchically structured landscapes. Oikos 67(3):521–528
6. Levin S (1992) The problem of pattern and scale in ecology. Ecology 73(6)
7. Hastings A (1991) Structured models of metapopulation dynamics. Biol J Linn Soc 42(1–2):57–71

Chapter 35
Herbivore-Algae Predator–Prey Dynamics

That the carnivore may live herbivores must die.

(H. Spencer Data of Ethics, 1879)

35.1 Herbivore-Algae Predator–Prey Model

Let us return to the simple predator–prey model to see that even without migration the system can exhibit a wide range of responses, not just a simple population crash. Assume that the prey are algae in a pond on which an herbivore grazes. The data for this problem has been invented. Its input data, parameters, and initial conditions would normally be determined by experiment.

The model consists of two main parts, one for the change in the algae population, one for the herbivore. The algae-growth portion of the model we have seen before in various forms. The growth rate is a function of the algal density, ALGAE. This function is monotonic and declining (Fig. 35.1). Algal growth is calculated as the product of the density and the growth rate.

The algae density is reduced through consumption by the herbivore. The consumption per head is a nonlinear function of the algal density: the greater the density, the higher the consumption per head. The consumption rate is simply the product of the number of herbivore and the consumption per head (Fig. 35.2).

The herbivore death rate is determined by their average life span, which is a nonlinear function of the consumption per head: the higher the consumption per head, the longer the life span, within limits (Fig. 35.3). Indirectly, the denser the algae, the lower the herbivore death rate.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_35,

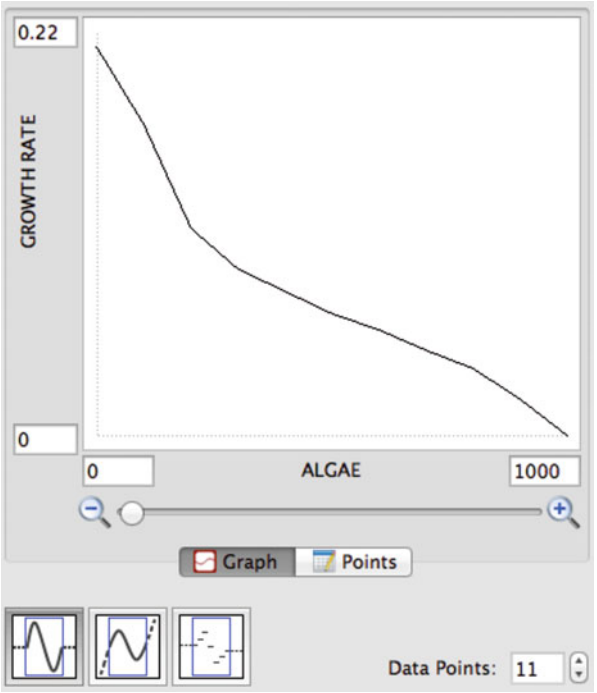

Fig. 35.1

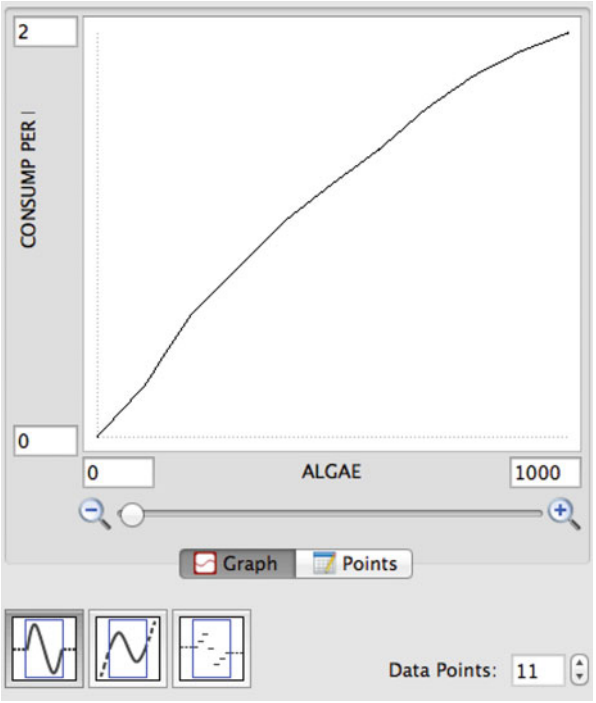

Fig. 35.2

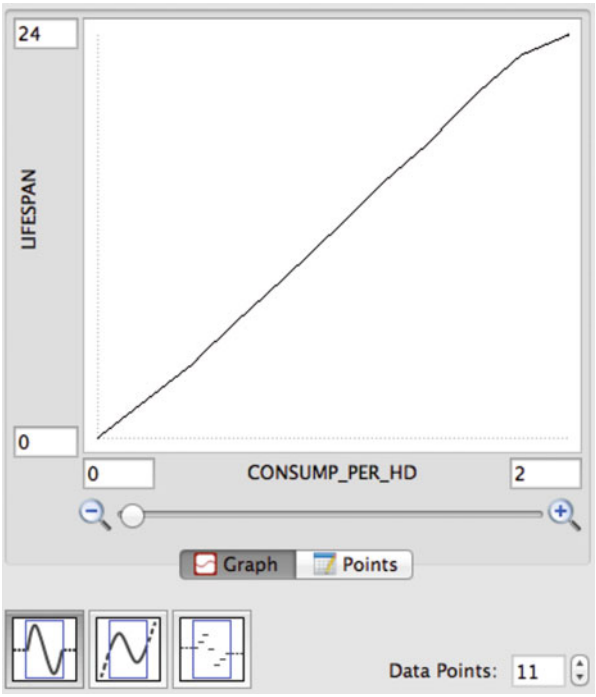

Fig. 35.3

The herbivore growth rate is a product of the herbivore stock and the fractional herbivore growth rate, FCN HERB GROW. To increase realism of the model, we make FCN HERB GROW a function of the algae density in the previous time period (Fig. 35.4). This is done by producing an additional stock called ALGAE DELAY, shown in Fig. 35.5. In general, it makes sense to represent herbivore behavior in this way. Herbivore gestation time reflects the origin of this lagged behavior.

Figure 35.6 shows the wide swings in algal density and herbivore population over time and Fig. 35.7 presents a plot of algal density against the herbivore population, which shows the limit cycle resulting from this particular choice of the variables.

Now it is your turn to try changing things. Can you make the herbivore crash and not re-emerge? Try to maximize the herbivore population. Can you do this by adjusting only the variable FCN HERB GROW, without changing the maximum and minimum rates?

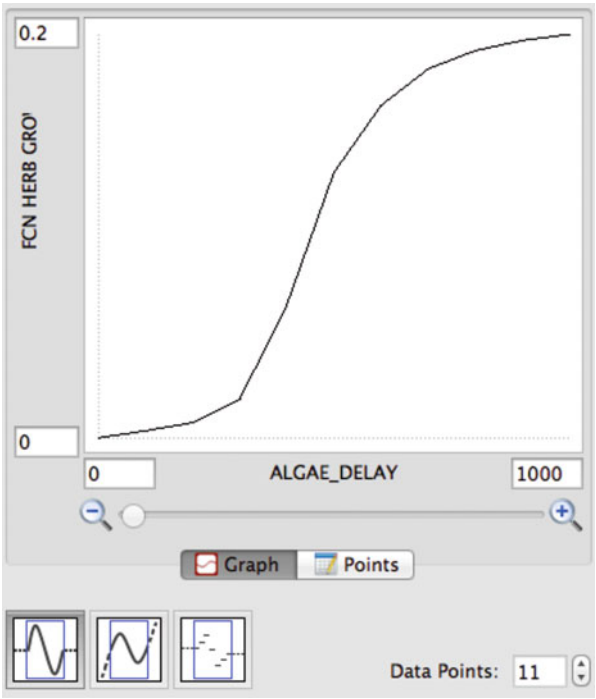

Fig. 35.4

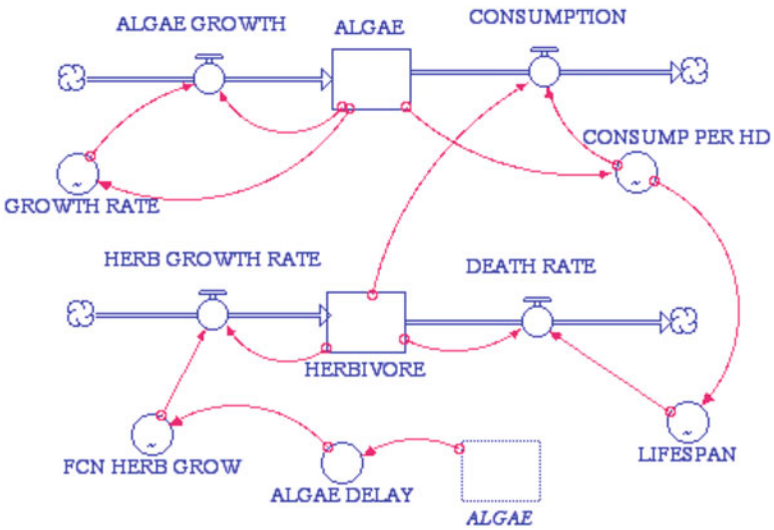

Fig. 35.5

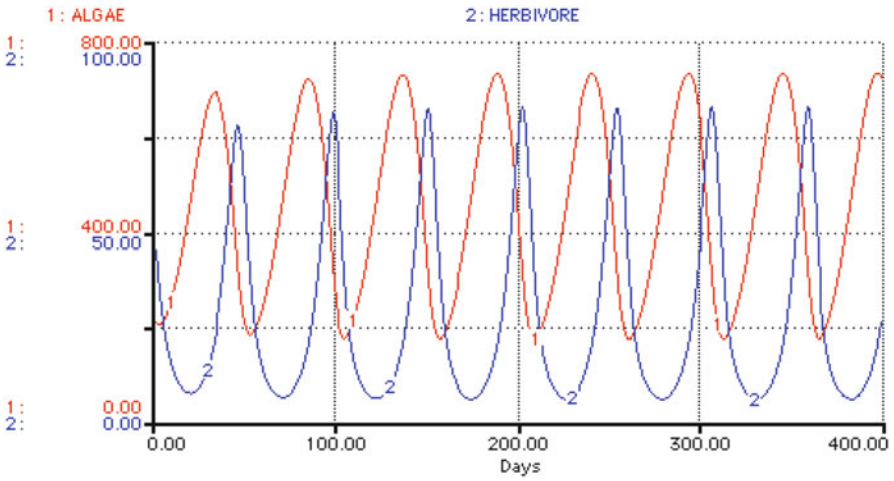

Fig. 35.6

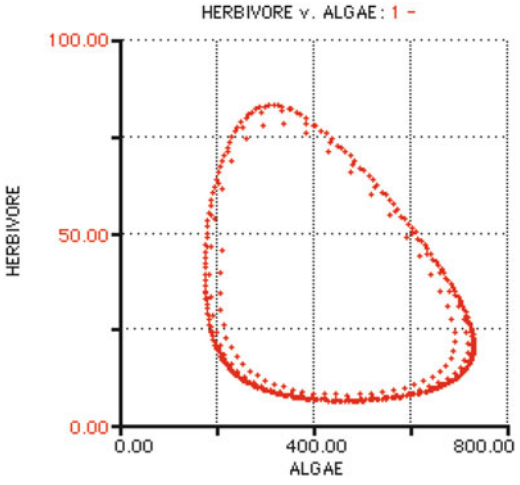

Fig. 35.7

35.2 Herbivore-Algae Predator–Prey Model Equations

ALGAE(t) = ALGAE(t − dt) + (ALGAE_GROWTH − CONSUMPTION) ∗ dt
INIT ALGAE = 210 {Algae per Area}
INFLOWS:
ALGAE_GROWTH = ALGAE∗GROWTH_RATE {Algae per Area per Time Period}

OUTFLOWS:
CONSUMPTION = HERBIVORE*CONSUMP_PER_HD {Algae per Area per Time Period}

HERBIVORE(t) = HERBIVORE(t − dt) + (HERB_GROWTH_RATE − DEATH_RATE) * dt
INIT HERBIVORE = 45 {Individuals}
INFLOWS:
HERB_GROWTH_RATE = HERBIVORE*FCN_HERB_GROW {Individuals per Time Period}
OUTFLOWS:
DEATH_RATE = HERBIVORE/LIFESPAN {Individuals per Time Period}

ALGAE_DELAY = DELAY(ALGAE,2) {Individuals}
CONSUMP_PER_HD = GRAPH(ALGAE)
(0.00, 0.00), (100, 0.25), (200, 0.6), (300, 0.83), (400, 1.06), (500, 1.24), (600, 1.41), (700, 1.61), (800, 1.77), (900, 1.89), (1000, 1.98)
FCN_HERB_GROW = GRAPH(ALGAE_DELAY)
(0.00, 0.00), (100, 0.0035), (200, 0.0075), (300, 0.019), (400, 0.065), (500, 0.13), (600, 0.163), (700, 0.181), (800, 0.19), (900, 0.195), (1000, 0.198)
GROWTH_RATE = GRAPH(ALGAE)
(0.00, 0.21), (100, 0.168), (200, 0.112), (300, 0.0902), (400, 0.0781), (500, 0.066), (600, 0.0572), (700, 0.0462), (800, 0.0363), (900, 0.0198), (1000, 0.00)
LIFESPAN = GRAPH(CONSUMP_PER_HD)
(0.00, 0.00), (0.2, 2.16), (0.4, 4.32), (0.6, 6.96), (0.8, 9.48), (1.00, 12.1), (1.20, 14.9), (1.40, 17.3), (1.60, 20.2), (1.80, 22.6), (2.00, 23.8)

Chapter 36
The Grass Carp

The Carp is the Queen of Rivers: a stately, a good, and a very subtle fish.

(Walton Angler, 1653)

36.1 Grass Carp Model

The Grass Carp model is a large one. It combines insight from the herbivore-algae model discussed in the previous chapter with the need for human management of the predator–prey relationship. A management practice known as "biomanipulation" has sprung from the idea of manipulating predator–prey relationships and is gaining popularity among lake management organizations. The model was motivated by the need of controlling the growth of grass in ponds and lakes. Nutrient rich waters flow into these bodies producing prodigious growth rates of a variety of plants. Plant growth is so luxuriant that sport fish cannot find food. To control the grass, carp are introduced to eat the plants. To prevent the waterway from being overrun by the carp, they are bred to be sterile. Such sterile carp are called "triploid" in this model. The carp can overdo it as well—if they eat all of the plants, they will starve and the young sport fish become easy prey for the large fish. When the carp reduce the grass biomass to about 35–45 % of the unregulated biomass, on the average, the optimum level of control is reached. The problem becomes one of finding the appropriate number, size, and time of introduction of the carp into the waterway. If such a way can be found, then biological control can successfully displace chemical vegetation control.

Nursery-raised carp are commonly sold at 200 g, the minimum size for safe transfer to a new waterway. The time horizon for most waterways is assumed to be

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_36,

5 years, that is, for convenience the stocking should occur about every 5 years. Thus, to predict the full effect of the fish, a 10-year period is set as the modeling time: fish are introduced in the spring of the first and the fifth year. Spring is chosen to maximize the survival rate of the young fish. An area of 1,000 m^2 is chosen as the basic unit of waterway area. The targeted water area is that portion of the waterway up to 6 m in depth. The data used here are reported in Wiley et al. [1].

There are three basic parts to this model: the number of fish, the average size of the fish, and the plant biomass. Let us first turn to the model component that deals with the number of fish.

In the following figure, the input to the number of carp, which we here call STOCK Rate, specifies when and how many 200 g fish are added to the waterway. The output flows, which diminish the number of fish, are grouped under the control MORTALITY. They are:

- PRED MORT, the consumption of the carp by the larger sport fish;
- WINTER MORT, the death of fish due to the harshness of the winter months; and,
- STARVE MORT, the death of the carp due to lack of food.

Predator mortality is a function of the winter period, WINTER, a mortality coefficient, MORT COEF, and the number of fish. The mortality coefficient, in turn, is a function of the average age and size of the fish and a variable that causes the mortality rate to increase as the fish ages, MORT COEF AGE. Winter mortality is a function of the number of fish, the winter period, WINTER FLAG, and the above aging coefficient. The WINTER FLAG indicates when winter occurs. It is defined as

$$\text{WINTER FLAG} = \text{IF}\,(\text{WEEK} \leq 12)\,\text{OR}\,(\text{WEEK} \geq 52)\,\text{THEN}\,1\,\text{ELSE}\,0 \quad (36.1)$$

The starvation mortality is a function of the number of fish, a mortality coefficient based on the water temperature, and a determination of the food scarcity, HUNGER. The hunger variable is determined by comparing the desired and actual ingestion rates, as discussed below. The corresponding part of the STELLA model is shown in Fig. 36.1.

Figure 36.2 shows how the temperature is derived. Average weekly air temperatures (10 year sample) for Region 11 Illinois were fit with a fourth order polynomial. The resulting equation is

$$\begin{aligned}\text{AIR TEMP} = {} & -2.8474 - 1.025 * \text{WEEK} + .2114 * (\text{WEEK})^2 \\ & - .0066 * (\text{WEEK})^3 + 5.548\text{E} - 5 * (\text{WEEK})^4 + \text{NORMAL}(1,3)\end{aligned} \quad (36.2)$$

Winter is defined as that period when the average weekly temperature dropped below 8 °C.

The next module displays the procedure for calculating the average size of the fish in the waterway, in calories and in grams. The fish ingest vegetation measured

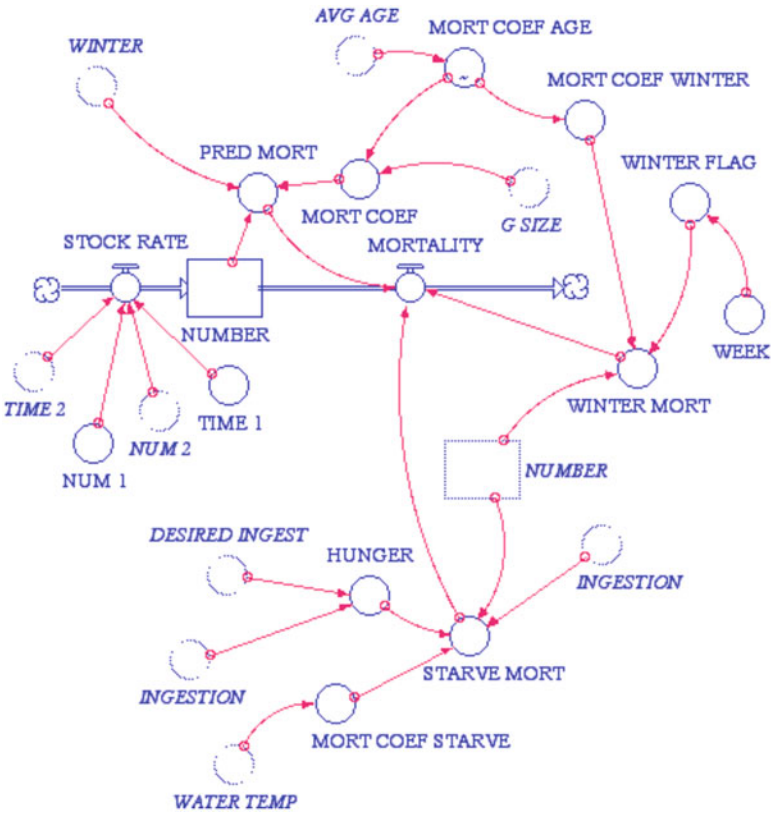

Fig. 36.1

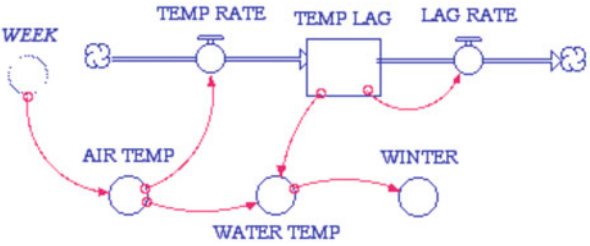

Fig. 36.2

in calories. They respire and excrete substances measured in calories as well. If the ingested calories exceed the calories excreted and respired, the fish will gain weight. If ingestion is smaller, the average size declines. The excess calories are converted to grams of flesh at different rates and the conversion efficiency declines as the fish grows larger. The conversion efficiencies for small and large fish, G SIZES and G SIZEL, are shown in Figs. 36.3 and 36.4.

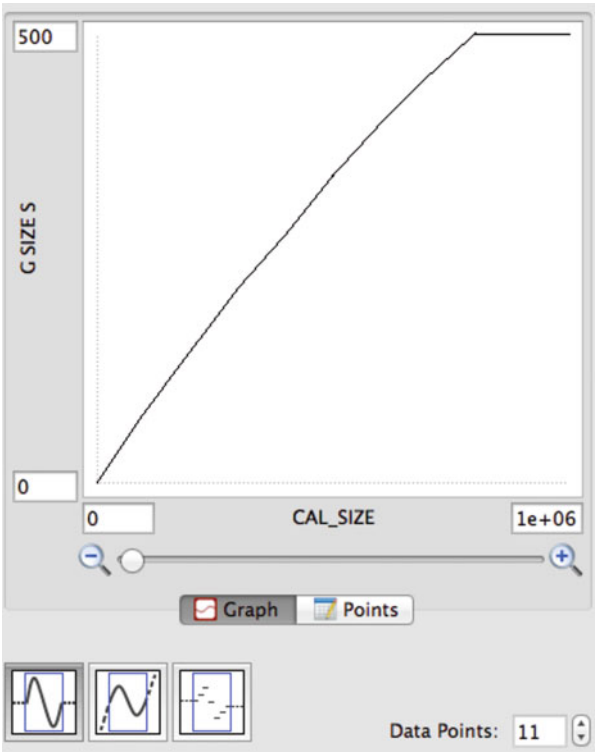

Fig. 36.3

The fish change their diet to one of declining caloric density as they grow older. These changes are captured by FCC1, FCC2, and FCC3:

$$\text{FCC1} = \text{IF } (\text{G SIZE} < 30) \text{ AND } (\text{G SIZE} \geq 0) \text{ THEN } 500 \text{ ELSE FCC2} \quad (36.3)$$

$$\text{FCC2} = \text{IF } (\text{G SIZE} \geq 30) \text{ AND } (\text{G SIZE} < 100) \text{ THEN } 1000 \text{ ELSE FCC3} \quad (36.4)$$

$$\text{FCC3} = \text{IF G SIZE} \geq 100 \text{ THEN } 430 \text{ ELSE } 0 \quad (36.5)$$

Actual ingestion is determined in the plant module (shown below) but the desired ingestion, DESIRED INGEST, is a function of the caloric density, the average eating rate, BWPD, a factor for comparing triploid eating rates to the natural carp eating rate, PLOIDY, the grams size, G SIZE, and a voraciousness factors TC and TC1 which depend on the water temperature:

$$\text{DESIRED INGEST} = 7 * \text{BWPD} * \text{G SIZE} * \text{TC} * \text{PLOIDY} * \text{FCC1} \quad (36.6)$$

$$\text{BWPD} = \text{IF G SIZE} < 15000 \text{ THEN } 0.52 \text{ ELSE } .21 \quad (36.7)$$

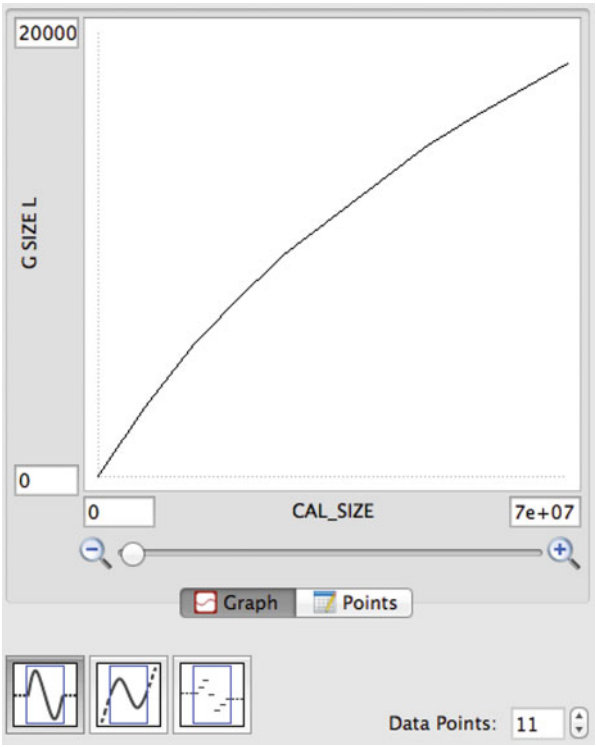

Fig. 36.4

$$\text{PLOIDY} = 0.91 \tag{36.8}$$

$$\begin{aligned}\text{TC} = {} & \text{IF (WATER TEMP} \geq 11\text{) AND (WATER TEMP} \leq 25\text{) THEN} \\ & -2.8591 + 1.19889 * \text{LOGN(WATER TEMP) ELSE TC1}\end{aligned} \tag{36.9}$$

$$\text{TC1} = \text{IF WATER TEMP} > 25 \text{ THEN } 1 \text{ ELSE } 0 \tag{36.10}$$

The respiration energy rate in Fig. 36.5 depends on the digestion effort, SDA, the activity level ACTIV FAC, and the standard metabolism, STANDARD METAB, the latter being dependent on the gram size and water temperature.

$$\begin{aligned}\text{RESPIRATION} = {} & \text{IF G\_SIZE} > 0 \text{ THEN} \\ & 7 * (\text{STANDARD METAB} * \text{ACTIV FAC}) \\ & + \text{SDA ELSE } 0\end{aligned} \tag{36.11}$$

$$\text{SDA} = .06 * \text{INGESTION} \tag{36.12}$$

$$\text{ACTIV FAC} = \text{TC}/(1.06 * \text{C}) + 1 \tag{36.13}$$

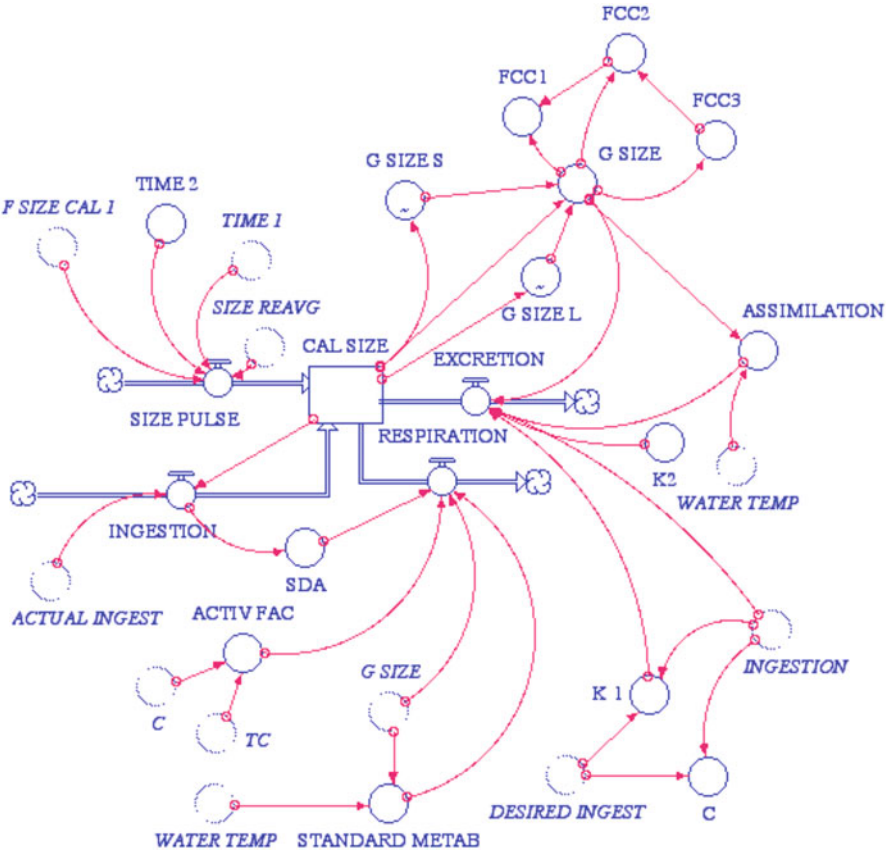

Fig. 36.5

$$\begin{aligned}\text{STANDARD METAB} = {} & \text{IF}(\text{WATER TEMP} \geq 0)\ \text{AND}\ (\text{G SIZE} > 1)\ \text{THEN} \\ & 82.2 * .026 * \left(\text{G SIZE}^{(.645)}\right) * \text{WATER TEMP}^{1.07} \\ & \text{ELSE}\ \ 0\end{aligned} \tag{36.14}$$

where conversions are made from milligrams of oxygen per fish-hour to standard calories per fish-day.

The excretion rate is a function of the amount of energy not assimilated and the fish size. The assimilation level depends in a complex way on the ratio of the desired to actual ingestion (Fig. 36.6).

$$\begin{aligned}\text{EXCRETION RATE} = {} & \text{IF G SIZE} > 0\ \text{THEN}\ (1 - \text{ASSIMILATION} * \text{K1}) \\ & * \text{K2} * \text{INGESTION ELSE}\ \ 0\end{aligned} \tag{36.15}$$

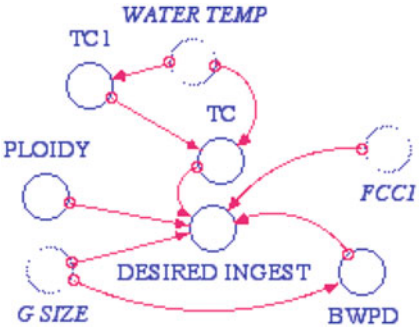

Fig. 36.6

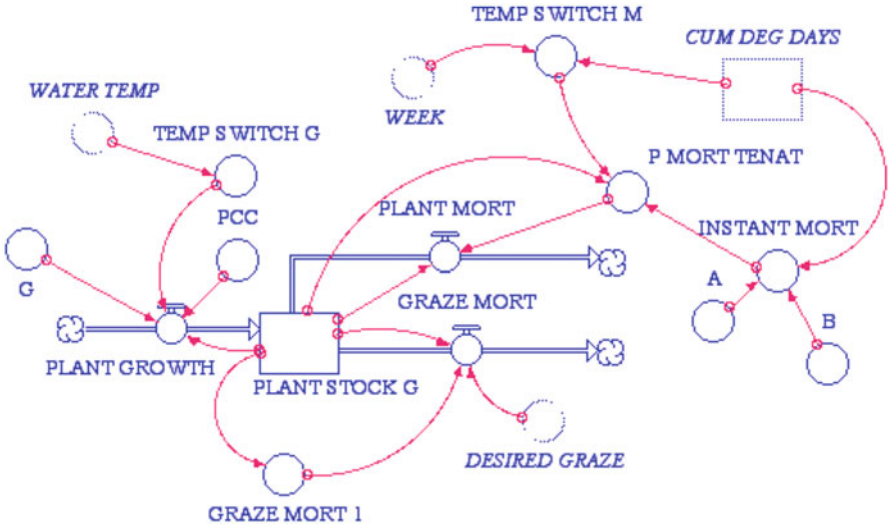

Fig. 36.7

Figure 36.7 shows the plant model. Growth of the plant is a simple logistic form with a specified upper limit on the maximum plant density, PCC = 500,000, and a specified growth rate G = 0.125. The growth rate is controlled by the temperature, TEMP SWITCH G:

$$\text{TEMP SWITCH G} = \text{IF WATER TEMP} > 15 \text{ THEN 1 ELSE 0} \qquad (36.16)$$

The temperature must be above a specified level for growth to occur. Plant mortality is controlled by the number of cumulative degree-days, which produces a mortality rate for the plant, INSTANT MORT

$$\text{INSTANT MORT} = \text{A} * \text{CUM DEG DAYS}^{\text{B}} \qquad (36.17)$$

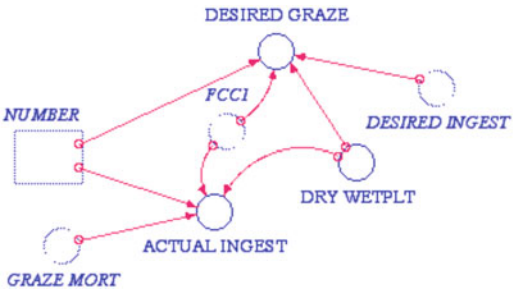

Fig. 36.8

A minimum level of vegetation (5,000 g) is preserved for regeneration and care must be taken to insure that plant mortality does not exceed this limit, P MORT TENAT

$$\text{P MORT TENAT} = 7 * \text{INSTANT MORT} * \text{PLANT STOCK G} * \text{TEMP SWITCH M}. \tag{36.18}$$

The most complicated part of the model is the removal of vegetation by fish grazing, GRAZE MORT (Figs. 36.7 and 36.8). Grazing mortality must not exceed the minimum level GRAZE MORT1 and the desired ingestion is consequently controlled. The desired grazing rate is the desired ingestion converted to grams of wet plant material from a dry caloric base. Actual ingestion is the allowed grazing rate converted back to dry calories.

$$\begin{aligned}\text{PLANT MORT} = {} & \text{IF P MORT TENAT} \geq \text{PLANT STOCK G} - 5000 \\ & \text{THEN (PLANT STOCK G} - 5000) \\ & \text{ELSE P MORT TENAT}\end{aligned} \tag{36.19}$$

$$\begin{aligned}\text{GRAZE MORT1} = {} & \text{IF PLANT STOCK G} - 5000 > 0 \\ & \text{THEN PLANT STOCK G} - 5000 \text{ ELSE } 0\end{aligned} \tag{36.20}$$

Figure 36.9 shows the two ways to find the average percentage plant biomass consumed by the fish for the 10-year period. The first module records the collective actual peak biomass levels for the ten periods and divides by ten and by the annual undisturbed peak in biomass. The second module simply integrates the area under the plant biomass–time curve and divides the sum by ten times the area under the undisturbed annual biomass curve. These are the two similar measures of the success of the stocking program being tested.

Figure 36.10 shows how the average yearly age AVG AGE of the current stock of fish at the second stocking time T2 is calculated. Before and after this time the average age of the fish is proportional to the TIME variable. The average age is used to change the winter mortality rate (see above).

Figure 36.11 gives the size-averaging module. In this module, the average size of the fish is thought to be sufficiently accurate. The alternative is to model each of the stockings independently.

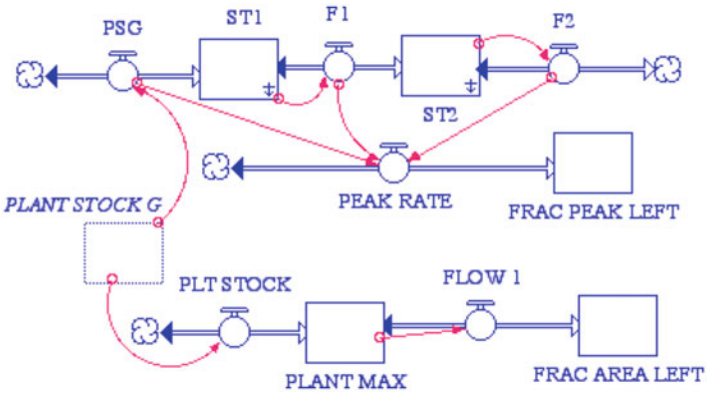

Fig. 36.9

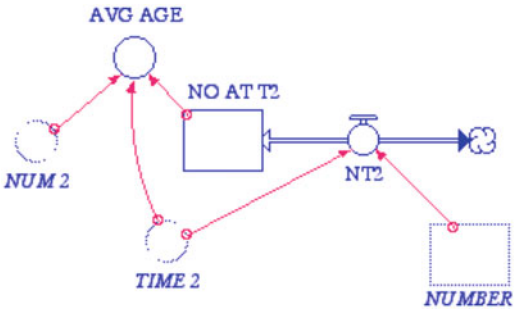

Fig. 36.10

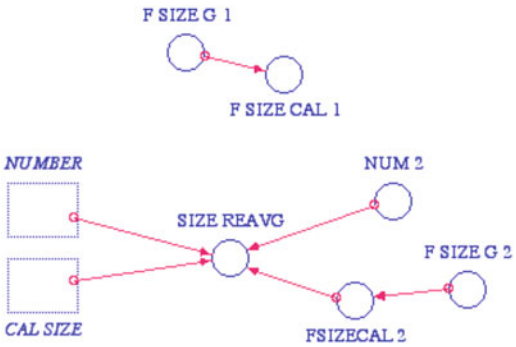

Fig. 36.11

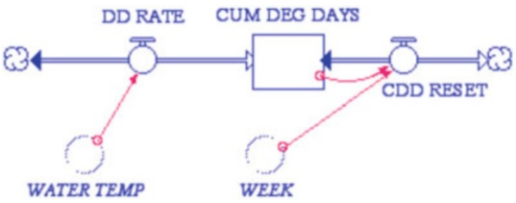

Fig. 36.12

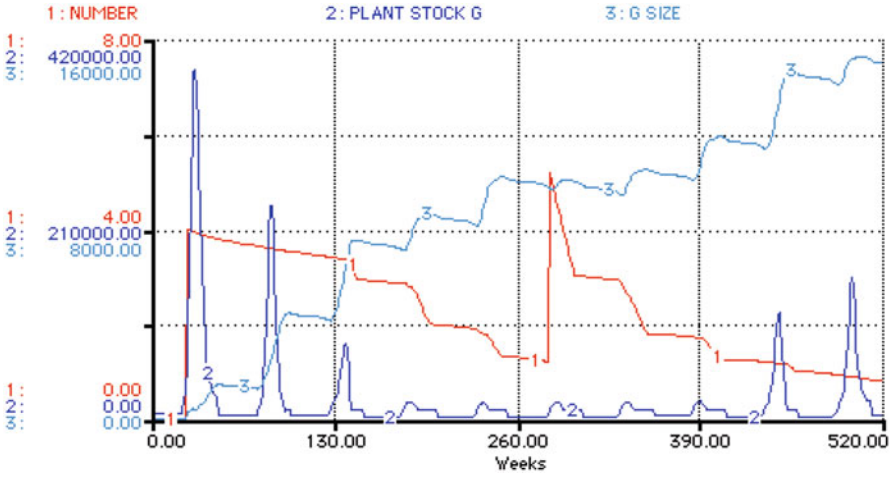

Fig. 36.13

The last module is set up for the degree-day calculation (Fig. 36.12). The base here is 0 °C.

Now all the necessary components of the model are laid out. Run the model as suggested above. It will yield the results shown in Fig. 36.13.

Now, experiment with the model, for example by choosing different stocking numbers. Try using a smaller number in the second stocking period to smooth out the vegetative peak variation. You will find that the weakest part of the model is the part where known air temperatures are converted into corresponding water temperatures. You will find that we have experimented with this connection, using random variations on the average weekly temperatures and a variety of lag times between the air and water temperatures.

36.2 Grass Carp Model Equations

ST1(t) = ST1(t − dt) + (PSG − F1) ∗ dt
INIT ST1 = 0
INFLOWS:
PSG = PLANT_STOCK_G {This section computes the max plant peak in each year, sums those peaks over the ten-year period of the run and then divides this sum by the sum of ten years of plant peaks which are undisturbed by grazing.}
OUTFLOWS:
F1 = ST1
ST2(t) = ST2(t − dt) + (F1 − F2) ∗ dt
INIT ST2 = 0
INFLOWS:
F1 = ST1
OUTFLOWS:
F2 = ST2
CAL_SIZE(t) = CAL_SIZE(t − dt) + (INGESTION + SIZE_PULSE − EXCRETION − RESPIRATION) ∗ dt
INIT CAL_SIZE = 0
INFLOWS:
INGESTION = IF CAL_SIZE > 0 THEN ACTUAL_INGEST ELSE 0 {Calories per Fish-Week}
SIZE_PULSE = PULSE(F_SIZE_CAL_1, TIME_1, 1000) + PULSE(SIZE_REAVG, TIME_2, 1000) {This is the pulsing of the two control sizes in std. cals, into the average calorie size state variable box.}
OUTFLOWS:
EXCRETION = IF G_SIZE > 0 THEN (1-ASSIMILATION∗K_1) ∗K2∗INGESTION ELSE 0 {Calories per Fish–Week}
RESPIRATION = IF G_SIZE > 0 THEN 7∗(STANDARD_METAB ∗ACTIV_FAC)+SDA ELSE 0 {Standard Calories per Fish–Week}
CUM_DEG_DAYS(t) = CUM_DEG_DAYS(t − dt) + (DD_RATE − CDD_RESET) ∗ dt
INIT CUM_DEG_DAYS = 0
INFLOWS:
DD_RATE = IF WATER_TEMP >= 0 THEN (WATER_TEMP)∗7 ELSE 0
OUTFLOWS:
CDD_RESET = IF WEEK = 52 THEN CUM_DEG_DAYS ELSE 0 {This control dumps the Cummulative Degree Days on Jan. 1st so that another accumulation can begin. }
FRAC_AREA_LEFT(t) = FRAC_AREA_LEFT(t − dt) + (FLOW_1) ∗ dt
INIT FRAC_AREA_LEFT = 0
INFLOWS:
FLOW_1 = IF TIME = 519 THEN PLANT_MAX/45300000 ELSE 0 {Divides the cumulative area under the plant stock (Grams) vs. time curve by the area under the standard curve (10 years), no variation in the avg. temp. curve}

FRAC_PEAK_LEFT(t) = FRAC_PEAK_LEFT(t − dt) + (PEAK_RATE) ∗ dt
INIT FRAC_PEAK_LEFT = 0
INFLOWS:
PEAK_RATE = IF (F1 > PSG) AND (F1 > F2) AND (F1 > 50000) THEN F1/4250000 ELSE 0
NO_AT_T2(t) = NO_AT_T2(t − dt) + (NT2) ∗ dt
INIT NO_AT_T2 = 0 {This section calculates the average age of the fish in years. It averages the fish of the second pulse with the age of the remaining fish from the first pulse.}
INFLOWS:
NT2 = IF TIME = TIME_2 THEN NUMBER ELSE 0
NUMBER(t) = NUMBER(t − dt) + (STOCK_RATE − MORTALITY) ∗ dt
INIT NUMBER = 0
INFLOWS:
STOCK_RATE = PULSE(NUM_1,TIME_1,1000) + PULSE(NUM_2, TIME_2,1000) {These are the Pulse functions. They only work with Euler integration and dt =1.00. See Specs Menu.}
OUTFLOWS:
MORTALITY = PRED_MORT + WINTER_MORT + STARVE_MORT
PLANT_MAX(t) = PLANT_MAX(t − dt) + (PLT_STOCK − FLOW_1) ∗ dt
INIT PLANT_MAX = 0
INFLOWS:
PLT_STOCK = PLANT_STOCK_G {This section computes the total area under the plant curve for the ten-year test run and then divides it by the total area under the undisturbed (by grazing) curve of plant growth.}
OUTFLOWS:
FLOW_1 = IF TIME = 519 THEN PLANT_MAX/45300000 ELSE 0 {Divides the cumulative area under the plant stock (Grams) vs. time curve by the area under the standard curve (10 years), no variation in the avg. temp. curve}
PLANT_STOCK_G(t) = PLANT_STOCK_G(t − dt) + (PLANT_GROWTH − PLANT_MORT − GRAZE_MORT) ∗ dt
INIT PLANT_STOCK_G = 5000 {Grams Dry Weight per 1000 Square Meters. To change veg. type, change variables: A, B, FCC3, G, PC, PCC, TEMP SWITCHG, and DRY WETPLT}
INFLOWS:
PLANT_GROWTH = IF PLANT_STOCK_G > 0 THEN 7∗TEMP_SWITCH_G∗G∗PLANT_STOCK_G∗(1-PLANT_STOCK_G/PCC) ELSE 0 {Grams dry weight per 1000 Square Meters–Week}
OUTFLOWS:
PLANT_MORT = IF P_MORT_TENAT >= PLANT_STOCK_G − 5000 THEN (PLANT_STOCK_G −5000) ELSE P_MORT_TENAT
GRAZE_MORT = IF DESIRED_GRAZE <= PLANT_STOCK_G − 5000 THEN DESIRED_GRAZE ELSE GRAZE_MORT_1 {Dry Weight Grams per Week. A 5000 Gram/Square Meter-Week reserve is maintained.}

TEMP_LAG(t) = TEMP_LAG(t − dt) + (TEMP_RATE − LAG_RATE) ∗ dt
INIT TEMP_LAG = 0 {Degrees C}
INFLOWS:
TEMP_RATE = AIR_TEMP {Degrees C per Time Period}
OUTFLOWS:
LAG_RATE = TEMP_LAG/3 {Degrees C per Time Period}
A = .11E-12 {from fig 3−11, page 3−27, part 3}
ACTIV_FAC = TC/(1.06∗C)+1
ACTUAL_INGEST = IF NUMBER > 0 THEN GRAZE_MORT∗FCC1/DRY_WETPLT/NUMBER ELSE 0 {Converting the actually allowed ingestion back to Wet Standard Calories per Fish-Week from Dry Vegetation Grams per Week.}
AIR_TEMP = −2.8474-1.025∗WEEK+.2114∗(WEEK)^2-.0066∗(WEEK)^3+5.548E−5∗(WEEK)^4 + NORMAL(1,3) {Degrees C}
ASSIMILATION = IF (WATER_TEMP > 1) AND (G_SIZE > 1) THEN −.026-.058∗LN(G_SIZE) + .213∗LN(WATER_TEMP) ELSE 0 {see eqn 8, page 859, Wiley & Wike, AFS, 1986.}
AVG_AGE = IF TIME >= TIME_2 THEN (NO_AT_T2∗TIME_2/52)/(NO_AT_T2 + NUM_2) + (TIME - TIME_2)/52 ELSE TIME/52
B = 3.45
BWPD = IF G_SIZE <15000 THEN 0.52 ELSE .21 {Average consumption rate for Elodea; Grams Wet Vegetation per Gram Fresh Fish–Day}
C = IF DESIRED_INGEST > INGESTION THEN 2 ELSE 1 {Reduces assimilation rate during starvation}
DESIRED_GRAZE = NUMBER∗DESIRED_INGEST∗DRY_WETPLT/FCC1 {Dry Vegetation Grams per Week. Conversion: from Standard Calories by / FCC1; from Wet Grams to Dry Grams by ∗DRY WETPLT}
DESIRED_INGEST = 7∗BWPD∗G_SIZE∗TC∗PLOIDY∗FCC1 {Regular Calories per Fish–Week}
DRY_WETPLT = .24 {This is the dry to wet weight ratio for Elodea.}
FCC1 = IF (G_SIZE < 30) AND (G_SIZE >= 0) THEN 500 ELSE FCC2 {Calories per gram of wet weight of Elodea c.}
FCC2 = IF (G_SIZE >= 30) AND (G_SIZE < 100) THEN 1000 ELSE FCC3 {Calories per gram of wet weight of Elodea c.}
FCC3 = IF G_SIZE >= 100 THEN 430 ELSE 0 {430 Calories per gram of wet weight of Elodea c.}
FSIZECAL_2 = 827.18∗F_SIZE_G_2^1.0968 + .0115∗F_SIZE_G_2^2.1936
F_SIZE_CAL_1 = 827.18∗F_SIZE_G_1^1.0968 + .0115∗F_SIZE_G_1^2.1936 {See Ref. under FSizeCal_2; eqn is multiplied by FSize_g_1 to get total Standard Calories per Fish}
F_SIZE_G_1 = 200 {Grams fresh weight, per fish. This variable is set by the user.}
F_SIZE_G_2 = 200 {The size of the average fish in the second pulse, in g fresh weight. This variable is set by the user.}
G = .125 {Instantaneous growth rate of Elodea c., Grams per Gram-Day}

GRAZE_MORT_1 = IF PLANT_STOCK_G − 5000 > 0 THEN PLANT_STOCK_G − 5000 ELSE 0

G_SIZE = IF CAL_SIZE < 700000 THEN G_SIZE_S ELSE G_SIZE_L {I broke the Wiley/Wike function into 2 parts for better accuracy. This relation controls the conversion of net cal. to fresh g of fish.}

G_SIZE_L = GRAPH(CAL_SIZE)

(0.00, 5.67e−317), (7e+06, 3100), (1.4e+07, 5800), (2.1e+07, 8000), (2.8e+07, 10000), (3.5e+07, 11600), (4.2e+07, 13200), (4.9e+07, 14800), (5.6e+07, 16100), (6.3e+07, 17300), (7e+07, 18500)

G_SIZE_S = GRAPH(CAL_SIZE)

(0.00, 6.05e-317), (100000, 78.0), (200000, 148), (300000, 218), (400000, 277), (500000, 342), (600000, 400), (700000, 452), (800000, 500), (900000, 500), (1e+06, 500)

HUNGER = 1− INGESTION/(DESIRED_INGEST+1.0) { The 1.0 keeps the ratio from becoming indefinite.}

INSTANT_MORT = A∗CUM_DEG_DAYS^B

K2 = .97 {calibration coefficient}

K_1 = 1 + (.2−.2∗(INGESTION/(DESIRED_INGEST+1.0))) {eqn 5, page 3−8, part 3. The 1.0 keeps 0/0 from being an indefinite number.}

MORT_COEF = IF (G_SIZE > 1.0) AND (G_SIZE < 100) THEN .04645−.00705∗LN (G_SIZE) ELSE MORT_COEF_AGE

MORT_COEF_AGE = GRAPH(AVG_AGE)

(0.00, 0.000495), (1.20, 0.0002), (2.40, 0.0002), (3.60, 0.00015), (4.80, 0.000395), (6.00, 0.001), (7.20, 0.001), (8.40, 0.001), (9.60, 0.001), (10.8, 0.001), (12.0, 0.001)

MORT_COEF_STARVE = IF WATER_TEMP < 20 THEN 0 ELSE .005479

MORT_COEF_WINTER = MORT_COEF_AGE

NUM_1 = 4 {Number of fish per 1000 sq. m. in the first pulse. This variable is set by the user.}

NUM_2 = 4 {Number of fish per 1000 sq. m. in the second pulse. This variable is set by the user.}

PCC = 500000 {Carrying capacity for Elodea; Dry Grams per 1000 Square Meters.}

PLOIDY = .91 {This is the factor for comparing triploid eating rates to the natural carp eating rate.}

PRED_MORT = IF NUMBER > 0 THEN 7∗MORT_COEF∗NUMBER∗(1-WINTER) ELSE 0 {number per week}

P_MORT_TENAT = 7∗INSTANT_MORT∗PLANT_STOCK_G∗TEMP_SWITCH_M {Grams per 1000 Square Meters}

SDA = .06∗INGESTION {Grams Dry Vegetation Equivalent per Fish–Week}

SIZE_REAVG = (NUMBER∗CAL_SIZE + NUM_2∗FSIZECAL_2)/ (NUM_2 + NUMBER) − CAL_SIZE {Reaverages the caloric size when the second pulse occurs.}

STANDARD_METAB = IF (WATER_TEMP >= 0) AND (G_SIZE > 1) THEN 82.2∗.026∗(G_SIZE^(.645))∗WATER_TEMP^1.07 ELSE 0 {82.2 converts from Milligram Oxygen per Fish-Hour to Standard Calories per Fish-Day}

STARVE_MORT = IF (NUMBER > 0) AND (INGESTION > 0) THEN 7*HUNGER*MORT_COEF_STARVE*NUMBER ELSE 0 {number per week, see eqn on page 3-14, part 3.}
TC = IF (WATER_TEMP >= 11) AND (WATER_TEMP <= 25) THEN −2.8591+1.19889*LN(WATER_TEMP) ELSE TC1
TC1 = IF WATER_TEMP > 25 THEN 1 ELSE 0
TEMP_SWITCH_G = IF WATER_TEMP > 15 THEN 1 ELSE 0 {Temp. growth threshold for Elodea C., Degrees C}
TEMP_SWITCH_M = IF (CUM_DEG_DAYS >= 800) AND (CUM_DEG_DAYS <= 10000) AND (WEEK > 1) AND (WEEK < 51) THEN 1 ELSE 0
TIME_1 = 21 {Time in weeks to the first pulse. Usually the best time is April or 17 weeks into the year. This variable set by user.}
TIME_2 = 281 {Time of the second pulse, in weeks. This variable set by user.}
WATER_TEMP = 3.06 + 0.32*TEMP_LAG + 0*AIR_TEMP {Degrees C}
WEEK = (TIME MOD 52) + 1 {Determines the number (1 to 52) of the week of the year}
WINTER = IF WATER_TEMP < 8 THEN 1 ELSE 0
WINTER_FLAG = IF (WEEK <=12) OR (WEEK >= 52) THEN 1 ELSE 0 {Indicates when winter occurs: December 21 thru March 21}
WINTER_MORT = IF NUMBER > 0 THEN 7*MORT_COEF_WINTER* WINTER_FLAG*NUMBER ELSE 0 {Number/Week}

Reference

1. Wiley MJ, Tazik PP, Sobaski ST (1987) Controlling aquatic vegetation with triploid grass CARP. Illinois Natural History Survey, Champaign (Circular 57)

Chapter 37
Recruitment and Trophic Dynamics of Gizzard Shad

The exquisite manipulation of the master gives to each atom of the multitude its own character and expression.

(Ruskin, 1843)

37.1 Gizzard Shad Model

The previous chapter provides one example of ways in which ecosystems can be managed via the deliberate manipulation of food webs. In this chapter, we develop a model of the addition of piscivorous (fish-eating) predators to a system that may enhance water quality by reducing algal biomass. These effects are obtained when predators diminish for example planktivore biomass, which in turn release zooplankton production. Increased numbers of zooplankton then result in lower numbers of algae, which increases water quality for human uses.

Unfortunately, numerous exceptions have been found within this simple "cascading" mechanism. For instance, due to rapid turnover of primary production, highly eutrophic systems are not easily limited by top-down regulation. Systems with many littoral plants also resist biomanipulation, since these plants serve as a reservoir of production and nutrients apart from the limnetic community.

The model developed in this chapter examines the recruitment and trophic dynamics of a freshwater clupeid, gizzard shad, in a flood control reservoir. Due to rapid growth, omnivorous food habits, and a high fecundity, populations of this fish species may escape both regulation by predators, and competition for food resources. As a result they may often impact predator and zooplankton populations more than they are impacted themselves. Such "middle-out" effects in the trophic

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_37,

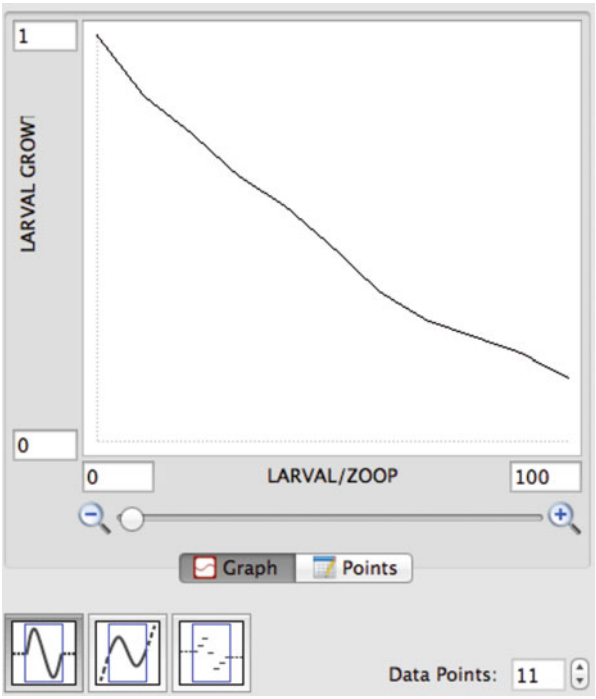

Fig. 37.1

cascade may thwart any attempt to improve water quality by adding predators to a system containing a substantial population of gizzard shad. Thus the question arises: Under what conditions could a lake manager hope to control gizzard shad populations and improve water quality through biomanipulation? In this chapter we examine the effect of various biomanipulation regimes, the effect of predator death rates, and gizzard shad egg survival.

Let us model a community of primary producers, zooplankton, planktivores (gizzard shad), and a predator species that forages optimal size classes of gizzard shad. The model is set up to run for approximately 10 years of simulated time. Gizzard shad growth is dependent on zooplankton density for the first two size classes (larval 5–20 mm, and early juvenile 20–40 mm). The larval and juvenile growth rates are graphically specified in Figs. 37.1 and 37.2.

Predation rates remain constant for a given abundance of predators. Thus survival of gizzard shad depends on the ability of the fish to outgrow predation. Growth is especially important during the first three life stages (larval, early juvenile, and late juvenile), but fish will transfer at set time intervals to higher size classes from the late juvenile stage onward. This maturation pattern reflects the foraging shift from zooplankton to detritus during the first year of life, and the diminishing importance of density dependent events for adult detritivores. Egg survival determined under the control variable SURVIVAL and WINTERKILL is

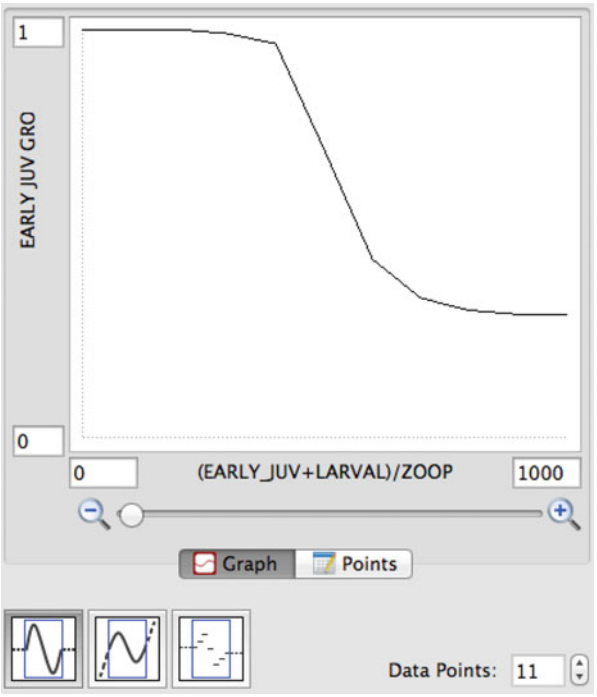

Fig. 37.2

controlled by SEVERITY. SURVIVAL is set to 0.008 and reflects year to year variation in egg survival due to flooding. SEVERITY is set equal to 0.1 and reflects year to year variation in winter severity and subsequent winter kill:

$$\text{WINTERKILL} = \text{PULSE}(\text{LARVAL} * \text{SEVERITY}, 50, 52) \tag{37.1}$$

Diminishing predation on late juvenile fish as the summer progresses is indicated by the controller variable SEASONAL SHIFT, which is a pulse that diminishes over time:

$$\text{SEASONAL SHIFT} = 1 - \text{PULSE}(1, 8, 52) \tag{37.2}$$

Here, the built-in PULSE function takes on the value of 1, which happens for the first time in period 8 and repeats itself every 52 periods.

Predation on larvae and small fish, i.e. early and late juveniles, is specified as shown in Figs. 37.3 and 37.4.

Predators are divided into three categories; juveniles (young of year or YOY), small predators, and adults. The STELLA diagram that captures the corresponding age cohort dynamic is shown in Figs. 37.5 and 37.6.

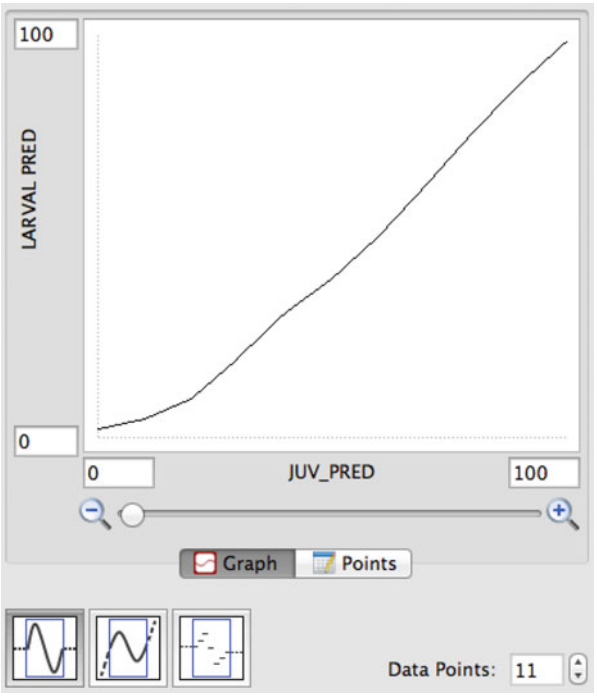

Fig. 37.3

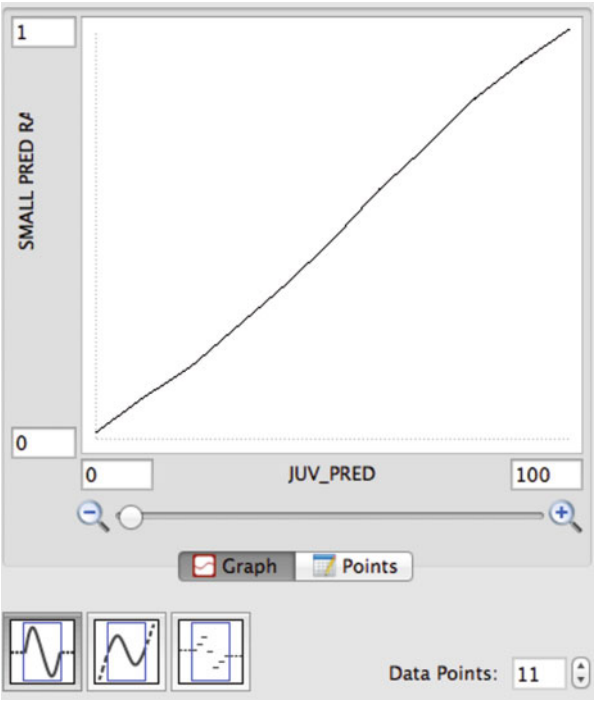

Fig. 37.4

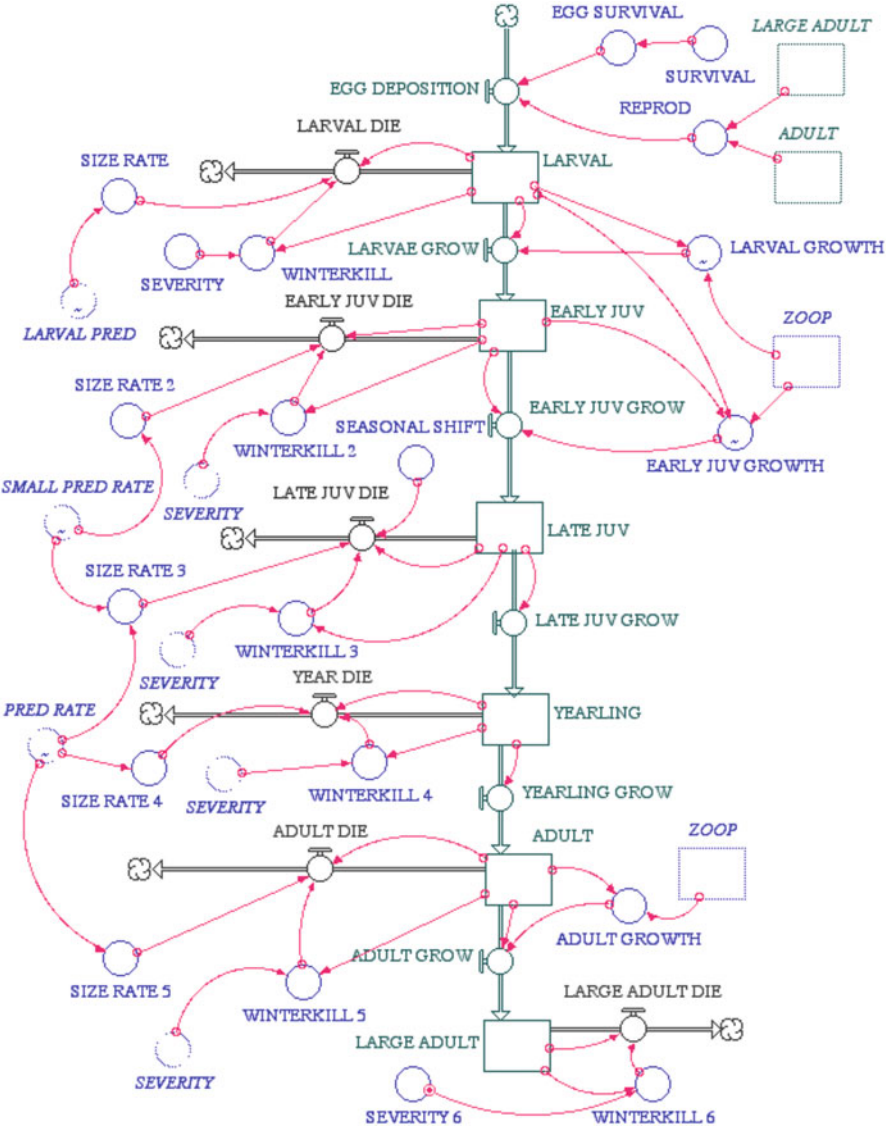

Fig. 37.5

Death of gizzard shad is determined by the density of predators of various size classes. The predation rate is given graphically in Fig. 37.7.

Large adult gizzard shad are not removed by predation at all. Regulation strength of predators is modified at the variable REGULATION STRENGTH. This control variable affects death rates of the predator and its ability to control gizzard shad. It may be said that the control is directly related to the spatial separation of the two

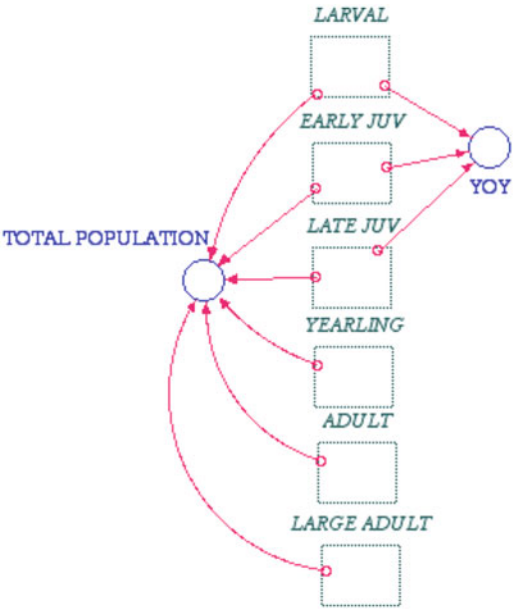

Fig. 37.6

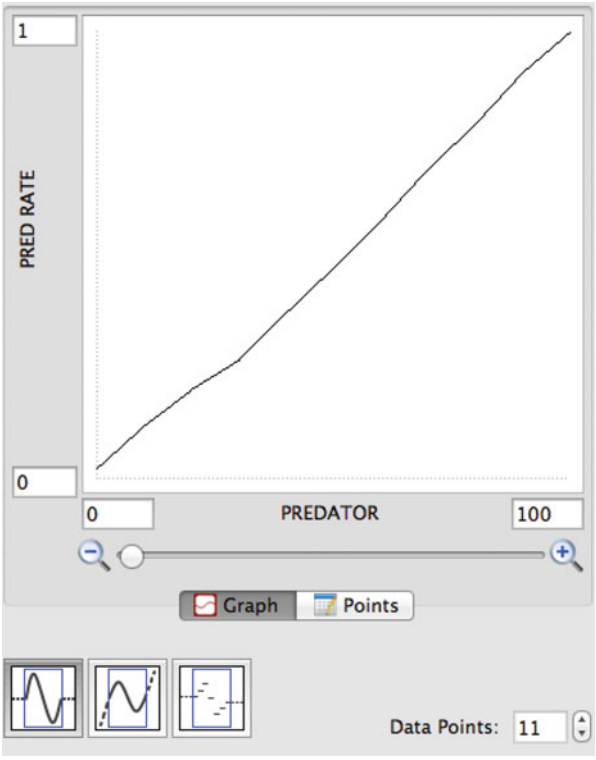

Fig. 37.7

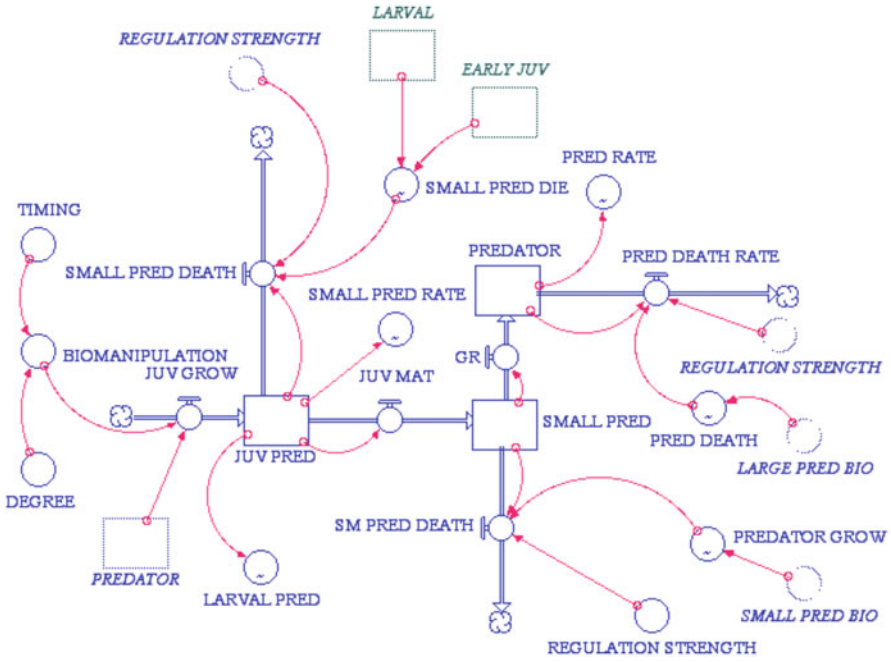

Fig. 37.8

species (gizzard shad occur in both the limnetic and littoral zone, and largemouth bass occur primarily in the littoral zone).

The population dynamics of the predator are given in the module of Fig. 37.8. Here, the death of small predators depends on the availability of gizzard shad larvae and early juveniles (Fig. 37.9).

Similarly, the effects of biomass of prey for small and adult predators are specified through control variables set up as graphs, shown in Figs. 37.10 and 37.11.

The modules capturing algal and zooplankton growth are set up as shown in Figs. 37.12 and 37.13.

The growth rates for algae and zooplankton are specified, respectively, as in Figs. 37.14 and 37.15.

Consumption rates of algae by zooplankton and of zooplankton by gizzard shad are also specified graphically (Figs. 37.16 and 37.17).

Now all the pieces of the model are in place and we are ready to investigate the impacts of biomanipulation. Our models show that the top-down manipulations are effective in controlling populations of gizzard shad (Figs. 37.18, 37.19, and 37.20).

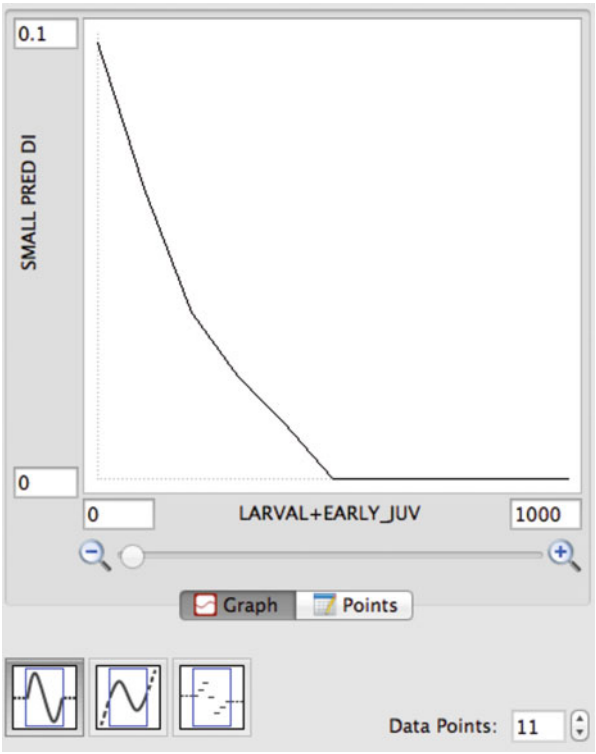

Fig. 37.9

Conduct sensitivity analyses for the impact of the number of predators and the regulation strength on algal bloom. You should find that by increasing the number of predators which are introduced into a system, the peak of algal blooms (which occur without these predators) is reduced and the number of algal blooms is also reduced. Regulation strength is effective in controlling algal biomass and gizzard shad numbers as well. By reducing the regulation strength, the amount of algae in the system is substantially enhanced, and increasing the amount of regulation strength diminishes the amount of algae in the system.

In a separate set of sensitivity analyses, assess the impacts of egg survival rates and winterkill on the system's dynamics. In the case of winterkill, reduction of the

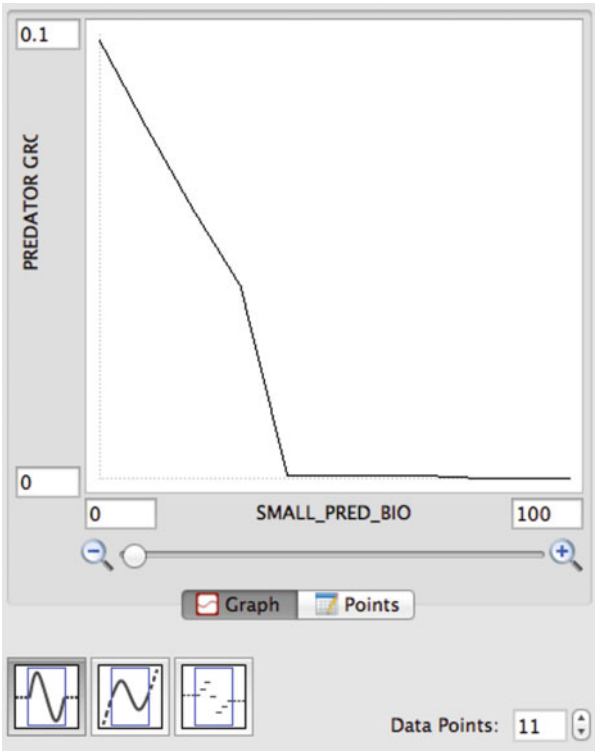

Fig. 37.10

population does not matter, because reproduction of gizzard shad is so great that a few adults could fill the system. Thus the peak numbers of larval gizzard shad are slightly reduced, but the overall population patterns remained unchanged. Egg survival also does not greatly effect the population or algal biomass.

From the results of sensitivity testing, it appears that manipulations of predator populations may still have success in controlling gizzard shad populations despite the fact of rapid gizzard shad growth and high fecundity. Lake managers should take care to conserve and possibly enhance predator stocks (taking care not to impact local stocks or environments) in lakes where water quality is important.

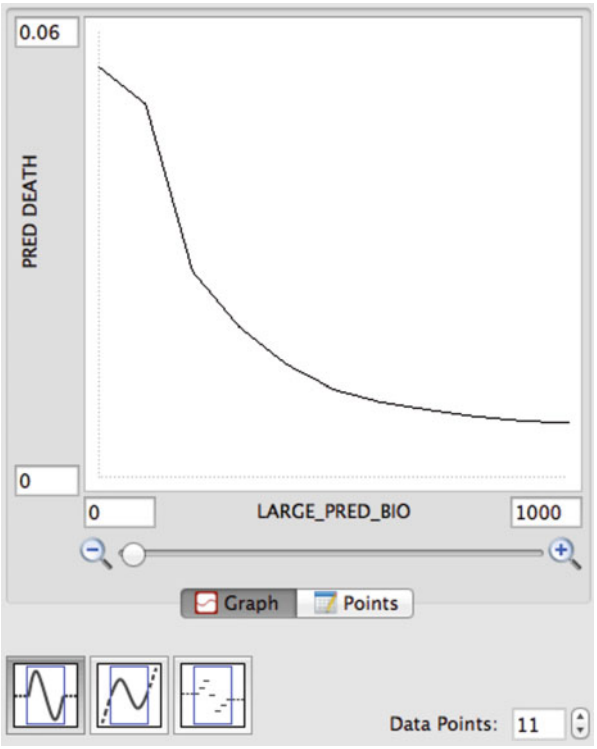

Fig. 37.11

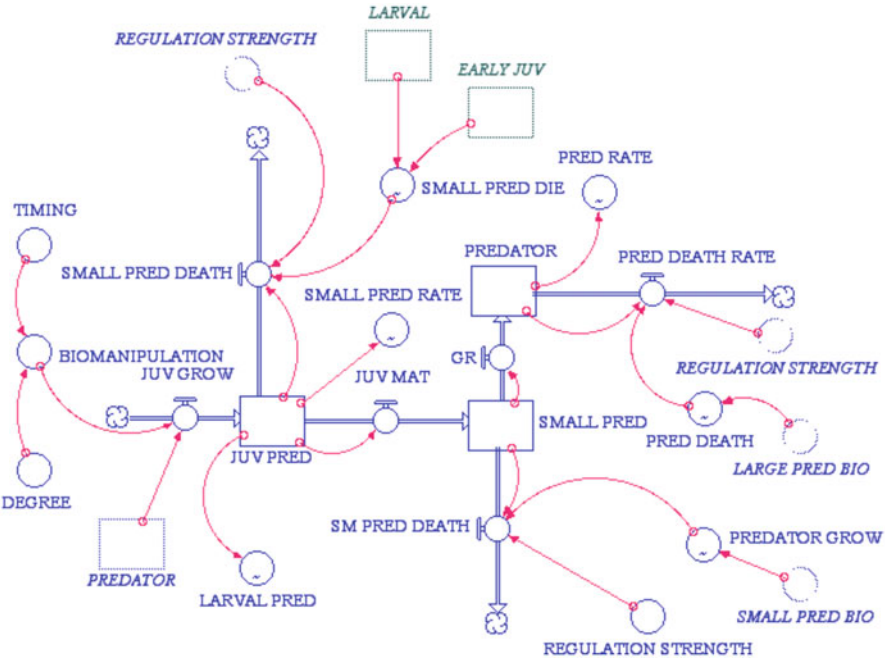

Fig. 37.12

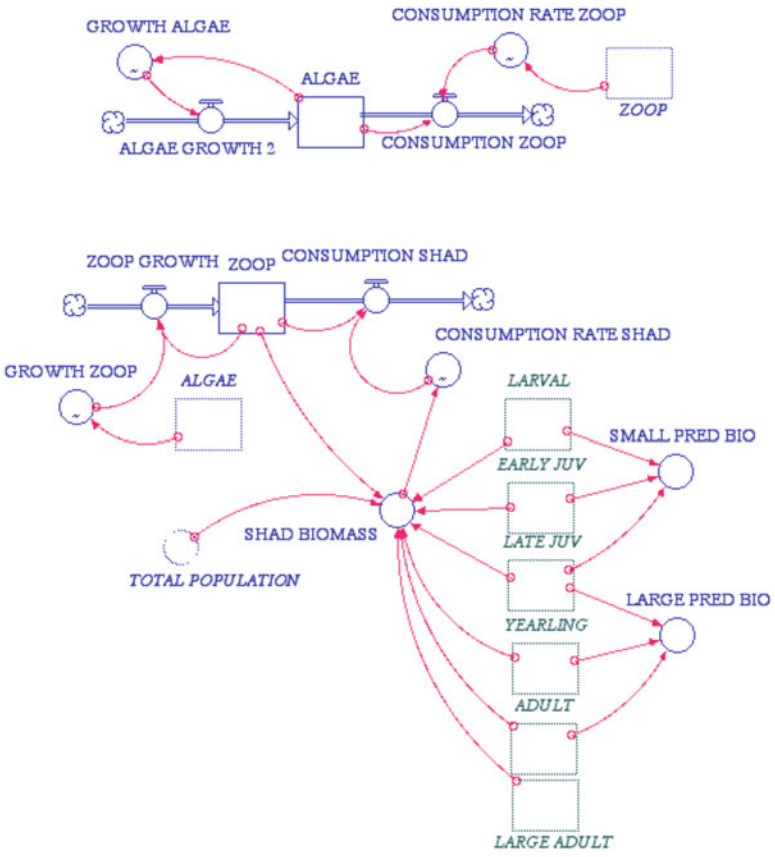
GROWTH ALGAE
CONSUMPTION RATE ZOOP
ALGAE
ZOOP
ALGAE GROWTH 2
CONSUMPTION ZOOP
ZOOP GROWTH
ZOOP
CONSUMPTION SHAD
CONSUMPTION RATE SHAD
GROWTH ZOOP
ALGAE
LARVAL
SMALL PRED BIO
EARLY JUV
SHAD BIOMASS
LATE JUV
TOTAL POPULATION
LARGE PRED BIO
YEARLING
ADULT
LARGE ADULT

Fig. 37.13

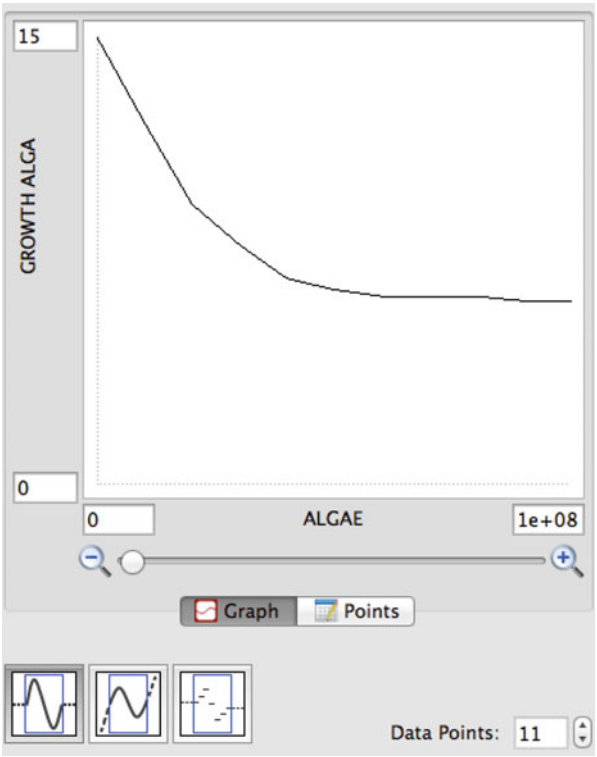

Fig. 37.14

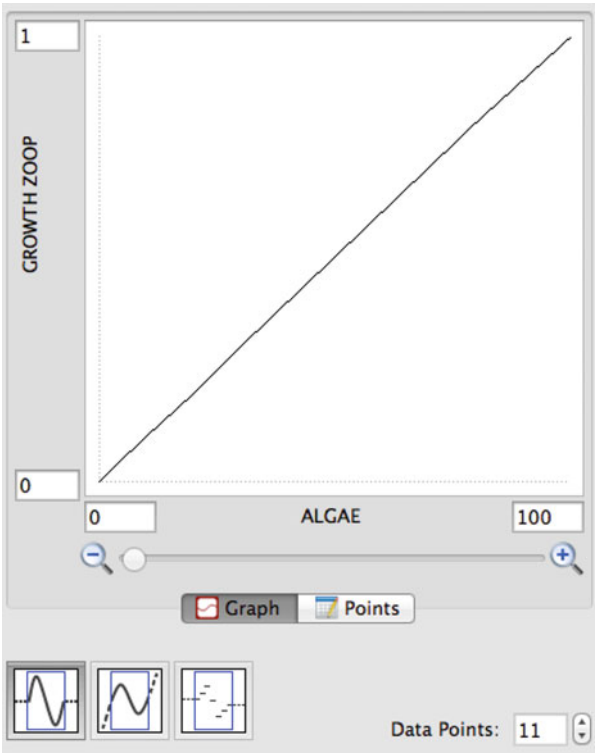

Fig. 37.15

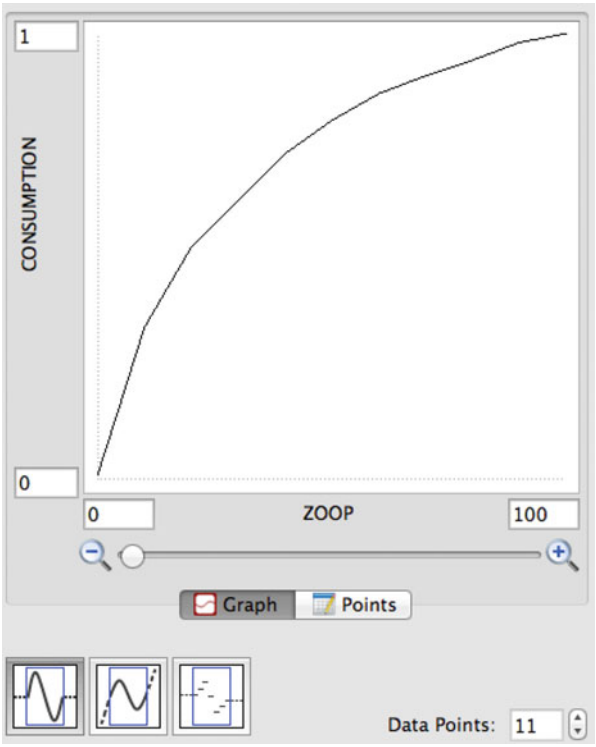

Fig. 37.16

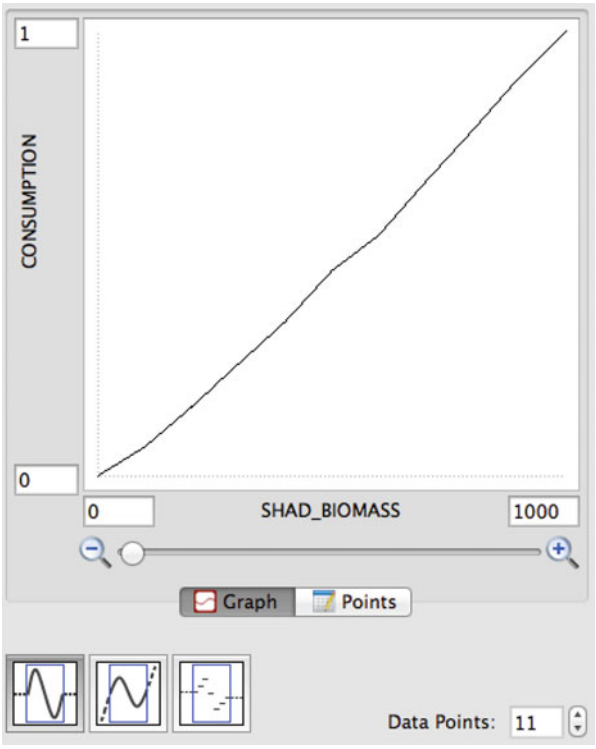
1
CONSUMPTION
0
0
SHAD_BIOMASS
1000
Graph
Points
Data Points:
11

Fig. 37.17

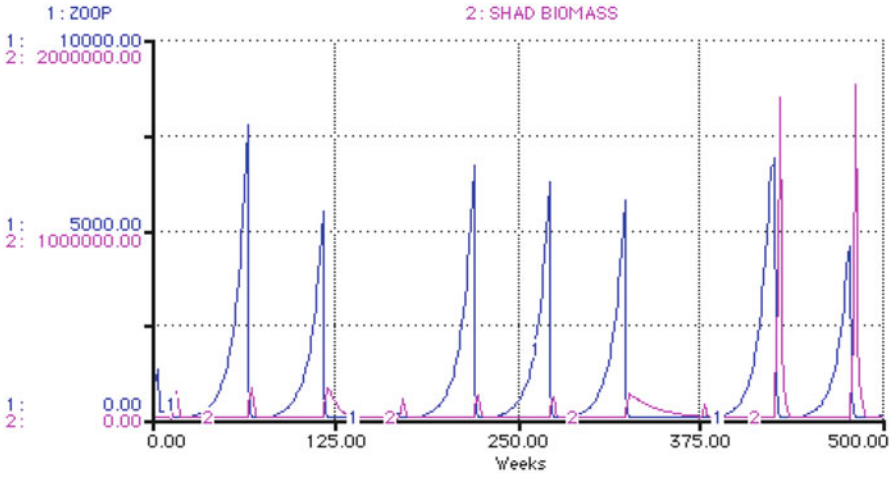

Fig. 37.18

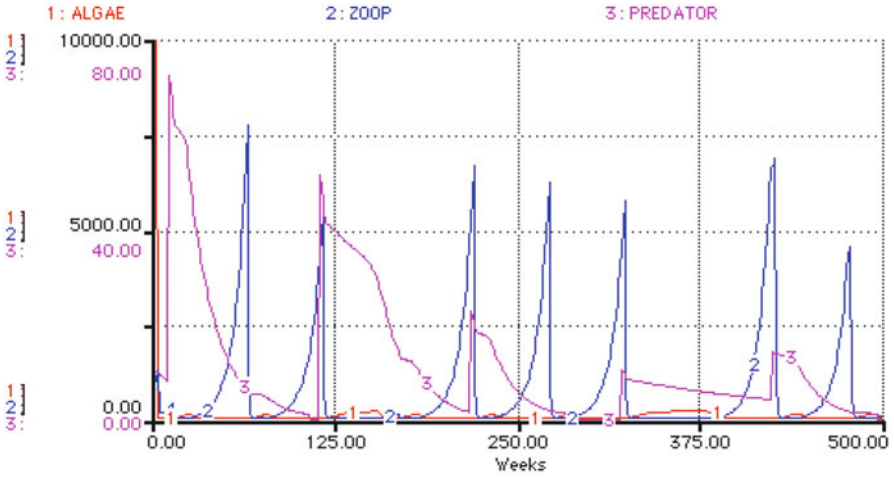

Fig. 37.19

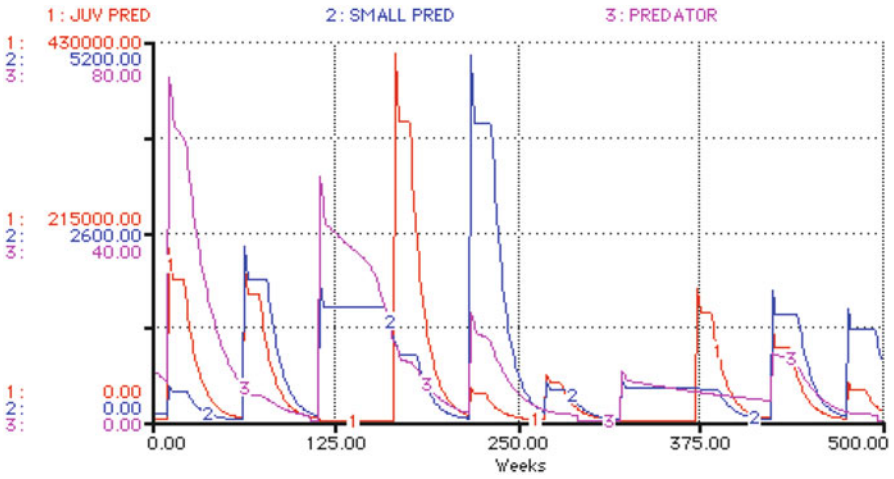

Fig. 37.20

37.2 Gizzard Shad Model Equations

ADULT(t) = ADULT(t − dt) + (YEARLING_GROW − ADULT_GROW − ADULT_DIE) ∗ dt
INIT ADULT = 100 {Individuals}
INFLOWS:
YEARLING_GROW = PULSE(YEARLING,7,52) {Individuals per Week}
OUTFLOWS:
ADULT_GROW = ADULT∗ADULT_GROWTH {Individuals per Week}
ADULT_DIE = IF ADULT <1 THEN 0 ELSE (ADULT∗SIZE_RATE_5) +WINTERKILL_5 {Individuals per Week}
ALGAE(t) = ALGAE(t − dt) + (ALGAE_GROWTH_2 − CONSUMPTION_ZOOP) ∗ dt
INIT ALGAE = 10000 {Units of Biomass}
INFLOWS:
ALGAE_GROWTH_2 = GROWTH_ALGAE {Units of Biomass per Week}
OUTFLOWS:
CONSUMPTION_ZOOP = (ALGAE∗CONSUMPTION_RATE_ZOOP) {Units of Biomass per Week}
EARLY_JUV(t) = EARLY_JUV(t − dt) + (LARVAE_GROW − EARLY_JUV_GROW − EARLY_JUV_DIE) ∗ dt
INIT EARLY_JUV = 100 {Individuals}
INFLOWS:
LARVAE_GROW = LARVAL∗LARVAL_GROWTH {Individuals per Week}
OUTFLOWS:

EARLY_JUV_GROW = EARLY_JUV*EARLY_JUV_GROWTH {Individuals per Week}
EARLY_JUV_DIE = IF EARLY_JUV < 1 THEN 0 ELSE (EARLY_JUV*SIZE_RATE_2)+WINTERKILL_2 {Individuals per Week}
JUV_PRED(t) = JUV_PRED(t − dt) + (JUV_GROW − SMALL_PRED_DEATH − JUV_MAT) * dt
INIT JUV_PRED = 1000 {Individuals}
INFLOWS:
JUV_GROW = PULSE(PREDATOR*25000,8,52)+BIOMANIPULATION {Individuals per Week}
OUTFLOWS:
SMALL_PRED_DEATH = (SMALL_PRED_DIE*REGULATION_STRENGTH)*JUV_PRED {Individuals per Week}
JUV_MAT = PULSE(JUV_PRED,8,52) {Individuals per Week}
LARGE_ADULT(t) = LARGE_ADULT(t − dt) + (ADULT_GROW − LARGE_ADULT_DIE) * dt
INIT LARGE_ADULT = 40 {Individuals}
INFLOWS:
ADULT_GROW = ADULT*ADULT_GROWTH {Individuals per Week}
OUTFLOWS:
LARGE_ADULT_DIE = IF LARGE_ADULT <1 THEN 0 ELSE (.001*(LARGE_ADULT))+WINTERKILL_6 {Individuals per Week}
LARVAL(t) = LARVAL(t − dt) + (EGG_DEPOSITION − LARVAE_GROW − LARVAL_DIE) * dt
INIT LARVAL = 100 {Individuals}
INFLOWS:
EGG_DEPOSITION = (PULSE(REPROD*20000*EGG_SURVIVAL,10,52))+(PULSE(REPROD*160000*EGG_SURVIVAL,12,52))+(PULSE(REPROD*2000*EGG_SURVIVAL,16,52)) {Individuals per Week}
OUTFLOWS:
LARVAE_GROW = LARVAL*LARVAL_GROWTH {Individuals per Week}
LARVAL_DIE = IF LARVAL <1 THEN 0 ELSE (LARVAL*SIZE_RATE)+WINTERKILL {Individuals per Week}
LATE_JUV(t) = LATE_JUV(t − dt) + (EARLY_JUV_GROW − LATE_JUV_GROW − LATE_JUV_DIE) * dt
INIT LATE_JUV = 75 {Individuals}
INFLOWS:
EARLY_JUV_GROW = EARLY_JUV*EARLY_JUV_GROWTH {Individuals per Week}
OUTFLOWS:
LATE_JUV_GROW = PULSE(LATE_JUV,9,52) {Individuals per Week}
LATE_JUV_DIE = IF LATE_JUV <1 THEN 0 ELSE ((LATE_JUV*SIZE_RATE_3)*SEASONAL_SHIFT)+WINTERKILL_3 {Individuals per Week}
PREDATOR(t) = PREDATOR(t − dt) + (GR − PRED_DEATH_RATE) * dt
INIT PREDATOR = 10 {Individuals}

INFLOWS:
GR = PULSE(SMALL_PRED,8,104) {Individuals per Week}
OUTFLOWS:
PRED_DEATH_RATE = IF PREDATOR <1 THEN 1 ELSE (PRED_DEATH*REGULATION_STRENGTH)*PREDATOR {Individuals per Week}
SMALL_PRED(t) = SMALL_PRED(t − dt) + (JUV_MAT − GR − SM_PRED_DEATH) * dt
INIT SMALL_PRED = 100 {Individuals}
INFLOWS:
JUV_MAT = PULSE(JUV_PRED,8,52) {Individuals per Week}
OUTFLOWS:
GR = PULSE(SMALL_PRED,8,104) {Individuals per Week}
SM_PRED_DEATH = (PREDATOR_GROW*REGULATION_STRENGTH) *SMALL_PRED {Individuals per Week}
YEARLING(t) = YEARLING(t − dt) + (LATE_JUV_GROW − YEARLING_GROW − YEAR_DIE) * dt
INIT YEARLING = 60 {Individuals}
INFLOWS:
LATE_JUV_GROW = PULSE(LATE_JUV,9,52) {Individuals per Week}
OUTFLOWS:
YEARLING_GROW = PULSE(YEARLING,7,52) {Individuals per Week}
YEAR_DIE = IF YEARLING <1 THEN 0 ELSE (YEARLING*SIZE_RATE_4) +WINTERKILL_4 {Individuals per Week}
ZOOP(t) = ZOOP(t − dt) + (ZOOP_GROWTH − CONSUMPTION_SHAD) * dt
INIT ZOOP = 1000 {Units of Biomass}
INFLOWS:
ZOOP_GROWTH = ZOOP*GROWTH_ZOOP {Units of Biomass per Week}
OUTFLOWS:
CONSUMPTION_SHAD = IF ZOOP <1 THEN 0 ELSE (ZOOP*CONSUMPTION_RATE_SHAD) {Units of Biomass per Week}
ADULT_GROWTH = ADULT/ZOOP*.0009
BIOMANIPULATION = PULSE(DEGREE,50,TIMING) {Individuals per Week}
CONSUMPTION_RATE_SHAD = GRAPH(SHAD_BIOMASS)
(0.00, 0.00), (100, 0.065), (200, 0.155), (300, 0.25), (400, 0.345), (500, 0.46), (600, 0.54), (700, 0.66), (800, 0.775), (900, 0.895), (1000, 1.00)
CONSUMPTION_RATE_ZOOP = GRAPH(ZOOP)
(0.00, 0.01), (10.0, 0.34), (20.0, 0.52), (30.0, 0.625), (40.0, 0.73), (50.0, 0.805), (60.0, 0.865), (70.0, 0.905), (80.0, 0.94), (90.0, 0.98), (100, 1.00)
DEGREE = 0 {Week}
EARLY_JUV_GROWTH = GRAPH((EARLY_JUV+LARVAL)/ZOOP)
(0.00, 1.00), (100, 1.00), (200, 1.00), (300, 0.99), (400, 0.965), (500, 0.705), (600, 0.435), (700, 0.34), (800, 0.31), (900, 0.3), (1000, 0.3)
EGG_SURVIVAL = ABS(RANDOM(SURVIVAL,.009))
GROWTH_ALGAE = GRAPH(ALGAE)
(0.00, 14.8), (1e+07, 12.0), (2e+07, 9.30), (3e+07, 7.95), (4e+07, 6.83), (5e+07, 6.45), (6e+07, 6.22), (7e+07, 6.22), (8e+07, 6.22), (9e+07, 6.08), (1e+08, 6.08)

GROWTH_ZOOP = GRAPH(ALGAE)
(0.00, 0.00), (10.0, 0.1), (20.0, 0.195), (30.0, 0.3), (40.0, 0.4), (50.0, 0.5), (60.0, 0.6), (70.0, 0.7), (80.0, 0.8), (90.0, 0.895), (100, 0.99)
LARGE_PRED_BIO = ADULT+LATE_JUV+YEARLING {prey for adults}
LARVAL_GROWTH = GRAPH(LARVAL/ZOOP)
(0.00, 0.995), (10.0, 0.845), (20.0, 0.755), (30.0, 0.65), (40.0, 0.575), (50.0, 0.475), (60.0, 0.365), (70.0, 0.295), (80.0, 0.255), (90.0, 0.215), (100, 0.155)
LARVAL_PRED = GRAPH(JUV_PRED)
(0.00, 2.00), (10.0, 4.50), (20.0, 9.50), (30.0, 19.5), (40.0, 30.5), (50.0, 39.0), (60.0, 49.5), (70.0, 62.0), (80.0, 75.0), (90.0, 87.0), (100, 97.5)
PREDATOR_GROW = GRAPH(SMALL_PRED_BIO)
(0.00, 0.098), (10.0, 0.0785), (20.0, 0.06), (30.0, 0.043), (40.0, 0.0005), (50.0, 0.0005), (60.0, 0.0005), (70.0, 0.0005), (80.0, 0.00), (90.0, 0.00), (100, 0.00)
PRED_DEATH = GRAPH(LARGE_PRED_BIO)
(0.00, 0.055), (100, 0.05), (200, 0.0275), (300, 0.02), (400, 0.015), (500, 0.0117), (600, 0.01), (700, 0.009), (800, 0.0081), (900, 0.0075), (1000, 0.0072)
PRED_RATE = GRAPH(PREDATOR)
(0.00, 0.02), (10.0, 0.115), (20.0, 0.195), (30.0, 0.26), (40.0, 0.365), (50.0, 0.465), (60.0, 0.57), (70.0, 0.685), (80.0, 0.785), (90.0, 0.9), (100, 0.99)
REGULATION_STRENGTH = 1
REPROD = ADULT+(LARGE_ADULT*1.2)
SEASONAL_SHIFT = 1−PULSE(1,8,52)
SEVERITY = .1 {reflects year to year variation in winter severity and subsequent winter kill}
SEVERITY_6 = RANDOM(.1,.01)
SHAD_BIOMASS = IF ZOOP > 2*TOTAL_POPULATION THEN ((LARVAL*.05)+(EARLY_JUV*3)+(LATE_JUV*10)+((YEARLING*30)+(ADULT*100)+(LARGE_ADULT*200))*.001)*.7 ELSE (LARVAL*.05)+(EARLY_JUV*3)+(LATE_JUV*3)
SIZE_RATE = LARVAL_PRED*.5
SIZE_RATE_2 = SMALL_PRED_RATE*.5
SIZE_RATE_3 = (SMALL_PRED_RATE*.5)+(PRED_RATE*.5)
SIZE_RATE_4 = PRED_RATE*.125
SIZE_RATE_5 = PRED_RATE*.01
SMALL_PRED_BIO = LARVAL+EARLY_JUV+(LATE_JUV*.5){prey for small predators}
SMALL_PRED_DIE = GRAPH(LARVAL+EARLY_JUV)
(0.00, 0.0975), (100, 0.0645), (200, 0.037), (300, 0.0225), (400, 0.012), (500, 0.00), (600, 0.00), (700, 0.00), (800, 0.00), (900, 0.00), (1000, 0.00)
SMALL_PRED_RATE = GRAPH(JUV_PRED)
(0.00, 0.015), (10.0, 0.1), (20.0, 0.175), (30.0, 0.275), (40.0, 0.375), (50.0, 0.485), (60.0, 0.61), (70.0, 0.715), (80.0, 0.83), (90.0, 0.92), (100, 1.00)
SURVIVAL = .008 {reflects year to year variation in egg survival due to flooding}
TIMING = 52 {Weeks}

TOTAL_POPULATION = ADULT+LARVAL+LARGE_ADULT+LATE_JUV +EARLY_JUV+YEARLING {Individuals}
WINTERKILL = PULSE(LARVAL∗SEVERITY,50,52) {Individuals per Week}
WINTERKILL_2 = PULSE(EARLY_JUV∗SEVERITY,50,52) {Individuals per Week}
WINTERKILL_3 = PULSE(LATE_JUV∗SEVERITY,50,52) {Individuals per Week}
WINTERKILL_4 = PULSE(YEARLING∗SEVERITY,50,52) {Individuals per Week}
WINTERKILL_5 = PULSE(ADULT∗SEVERITY,50,52) {Individuals per Week}
WINTERKILL_6 = PULSE(LARGE_ADULTSEVERITY_6,50,52) {Individuals per Week}
YOY = EARLY_JUV+LARVAL+LATE_JUV {Individuals}

Chapter 38
Salamander Dispersal

The salamanders, like tiny birds, locked into formation, fly down into the endless mysteries of the transforming water, and how could anyone believe that anything in this world is only what it appears to be—that anything is ever final—that anything, in spite of its absence, ever dies a perfect death? (from the poem 'What Is It?')".

(Mary Oliver, House of Light)

38.1 Salamander Dispersal Model

The models of the two previous chapters concentrated on biomanipulation of populations within a given habitat. In this chapter we turn to the colonization of new habitat by a species. Our example is for the spotted salamander (*Ambystoma maculatum*), which is a pond breeding species found throughout most of the eastern United States and southern Canada, west to eastern Iowa, and eastern Texas. Adults breed in ephemeral woodland pools once a year in the spring. Females lay an average of 125 eggs per clutch. Eggs hatch into an aquatic larvae after 1–2 months. Larvae remain in the pond until they metamorphose into the terrestrial juvenile stage within 2–4 months. Juveniles then disperse from their natal pond to surrounding forested areas remaining under leaf litter and dead logs or utilizing small mammal burrows until they mature to a reproductive stage. Females are considered mature after 7 years, and males are considered mature after 2–6 years. Adult salamanders also travel to terrestrial areas away from the pond during the nonbreeding season, yet they return to the same breeding pond every year, so dispersal should occur primarily during the juvenile stage.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_38,

Given these movement patterns throughout the life of salamanders, we explore the time it would take for salamanders to colonize habitat that becomes available to them, such as ponds that are created nearby their current location. To do so, we begin with a population of spotted salamanders inhabiting a single woodland pond, but with access to a new pond—Pond 2—in the same woodland area as Pond 1. We also assume a third pond—Pond 3—in a nearby woodlot, accessible only through an open grassy field.

We assumed that dispersal only occurs during the juvenile stage since the eggs/larvae are aquatic and adults are known to have a high level of site fidelity. Initially, we assumed that each life stage has its own death rate, but that these death rates would be the same across all three ponds.

Following work by Rothermel and Semlitsch [1] we assume that only 14 % of juveniles disperse farther than 100 m, so we used this number to represent the total percentage of juveniles moving far enough from their natal pond to colonize adjacent ponds. Several studies show that juveniles show a preference to disperse through forest versus grass, and so we assume that 10 % disperse through the forest and 4 % disperse through grass. Recapture rates from different habitats are used to represent the survival rates of the juvenile salamanders as they moved through grass (0.167) and through forest (0.333) Rothermel and Semlitsch [1]. And because it takes 7 years for the females to mature and reach reproductive age, we simply account for this aging delay by dividing the juvenile population of each pond by seven, so that 1/7 of each juvenile cohort would enter the adult population.

Dispersal rates, death rates, hatching success, and average clutch size all are highly dependent on environmental factors, such as predation, water characteristics, and weather. The parameters used here have been chosen to reproduce an observational 4-year study monitoring the adult population size of a single pond Blackwell et al. [2]. There are 236 adults and 365 juveniles in Pond 1 at a carrying capacity of 274.

Figure 38.1 shows the STELLA model for the three ponds. Here, we use a converter to capture the field observations (Observed Pop) on population in a pond—data against which we will compare our model results. For simplicity, we assume that the final population size in that pond corresponds to the steady-state population. The hatching success rate has been calibrated (through sensitivity analysis) to yield population sizes that are, in the long run, close to that steady state (Fig. 38.2). The particular value we chose is 0.27, and the larval, juvenile, and adult death rates that correspond with that hatching success rate to generate acceptable results are, respectively, 0.5, 0.43, and 0.17.

The adults in Pond 2 and 3 rise to an initial surge after the juveniles disperse from Pond 1 and colonize the other two ponds (Fig. 38.2). Adults reach a peak in Pond 2 before they do in Pond 3 because juvenile dispersal through forest to Pond 2 is easier than dispersal through grass to Pond 3. Initial adult populations peak after juveniles because the juveniles are the ones dispersing from the natal pond (Pond 1) to the other two ponds (Fig. 38.3). Upon maturation, they grow into adults and begin reproducing in each of the two new ponds.

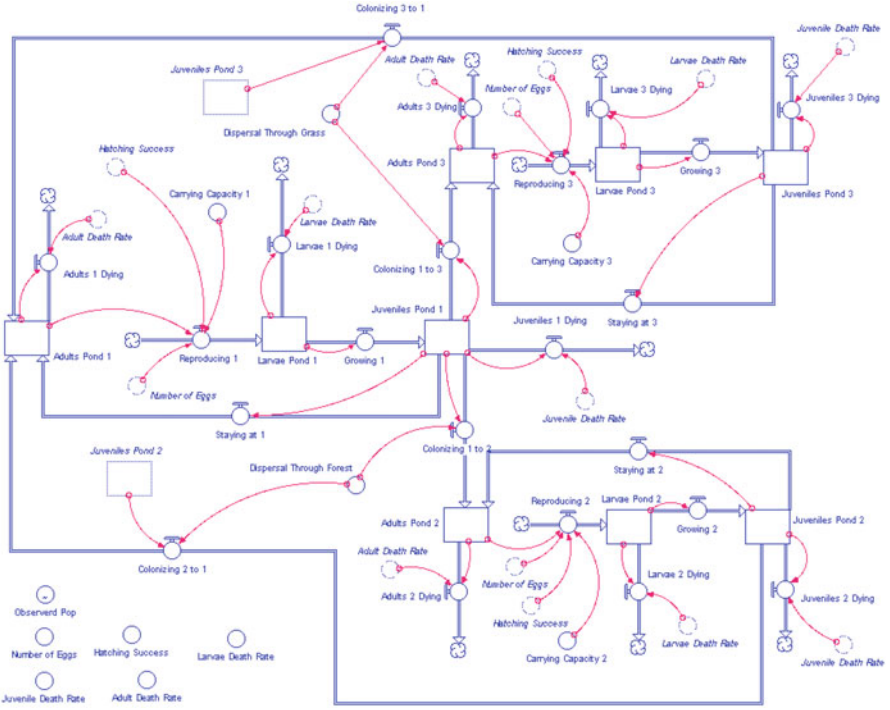

Fig. 38.1

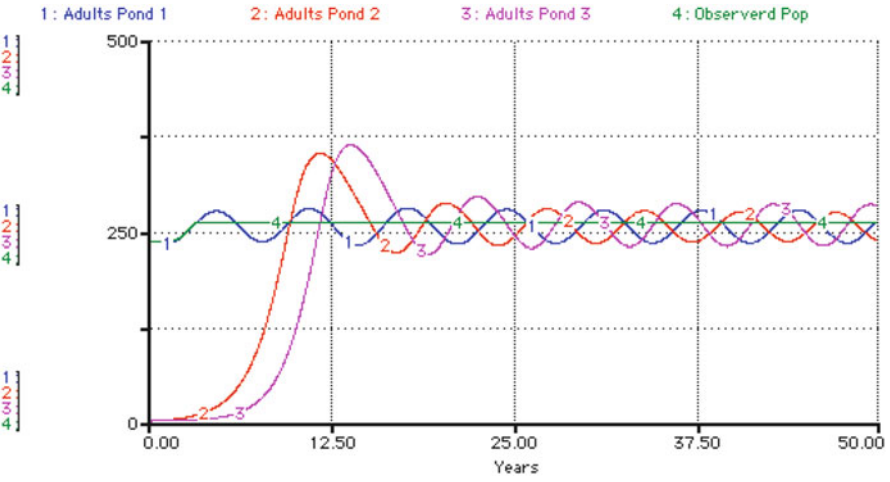

Fig. 38.2

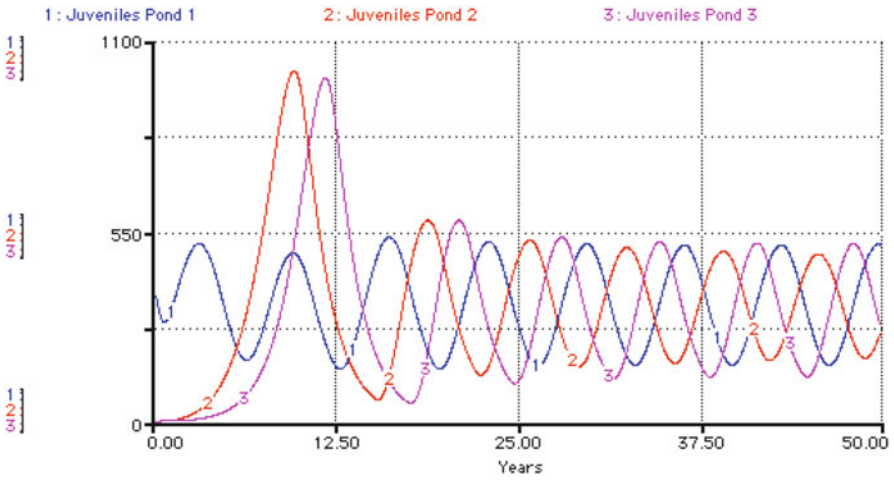

Fig. 38.3

Let us now experiment with this set-up and assume that the carrying capacity in Pond 2 is 180 instead of the 27 assumed in the previous run. Although it is preferable for the salamander to disperse through forest at a higher survival rate of dispersal than dispersing through grass, the population of Pond 2 decreases compared to Ponds 1 and 3 if the habitat is less suitable and has a lower carrying capacity. This decrease demonstrates a lower cost-benefit ratio of choosing to disperse through grass rather than forest—even though survival of dispersing through grass may be lower, it is more beneficial for the overall population in the long run if juveniles survive because the habitat is more suitable and can maintain a larger population. These are the results captured in Fig. 38.4.

What would happen in the case of equal carrying capacities of 274 in each of the three ponds, but Pond 1 suffering suddenly from local extinction? We model the extinction with conditional statements so that the death rate for each life stage is 100 % for 5 years between time periods 10 and 15. In addition, during those 5 years, the carrying capacity is set to 1 for the years between 10 and 15, but the usual 274 otherwise. These changes cause the population to crash starting at year 10 and reaching an adult population low of five individuals and a juvenile population low of two individuals at year 15 since the other populations are still dispersing to Pond 1 (Fig. 38.5). Once the extinction reaches a low, both the juvenile and adult populations began increasing almost immediately, leading to a temporary population surge after the local extinction overshoots the carrying capacity. Juvenile and adult populations then decline and finally stabilize, oscillating around the carrying capacity much as before.

How would the model results change if all of the ponds have different carrying capacities? How would the populations vary if each pond had different death rates and hatching success based upon different environmental factors such as predation, pH levels, hydrologic period of the ponds, food availability, and temperature?

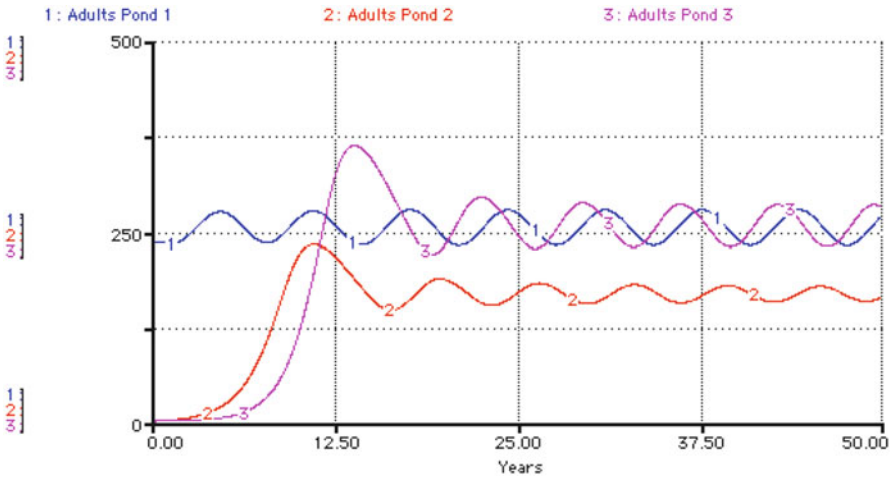

Fig. 38.4

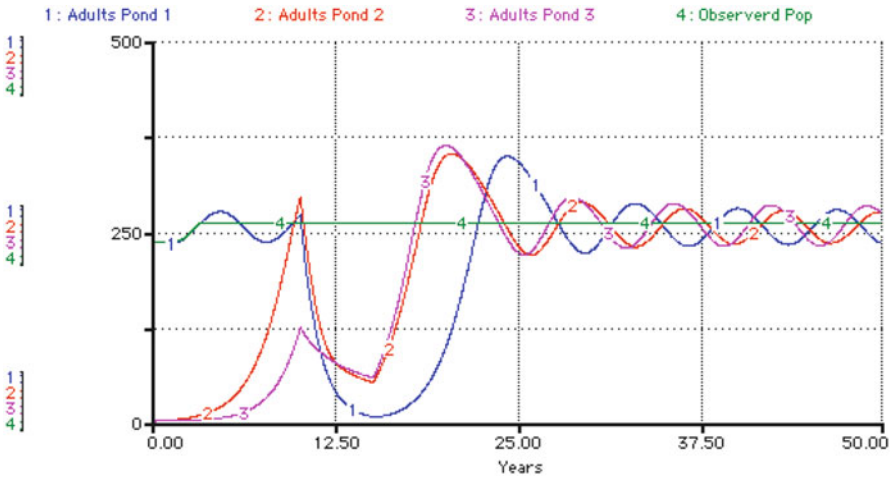

Fig. 38.5

38.2 Salamander Dispersal Model Equations

Adults_Pond_1(t) = Adults_Pond_1(t − dt) + (Colonizing_3_to_1 + Staying_at_1 + Colonizing_2_to_1 − Adults_1_Dying) ∗ dt
INIT Adults_Pond_1 = 236
INFLOWS:
Colonizing_3_to_1 = (Juveniles_Pond_3/7)∗Dispersal_Through_Grass∗0.04
Staying_at_1 = (Juveniles_Pond_1/7)∗0.86

Colonizing_2_to_1 = (Juveniles_Pond_2/7)*Dispersal_Through_Forest*0.10
OUTFLOWS:
Adults_1_Dying = Adults_Pond_1*Adult_Death_Rate
Adults_Pond_2(t) = Adults_Pond_2(t − dt) + (Colonizing_1_to_2 + Staying_at_2 − Adults_2_Dying) * dt
INIT Adults_Pond_2 = 0
INFLOWS:
Colonizing_1_to_2 = (Juveniles_Pond_1/7)*Dispersal_Through_Forest*0.10
Staying_at_2 = (Juveniles_Pond_2/7)*0.86
OUTFLOWS:
Adults_2_Dying = Adults_Pond_2*Adult_Death_Rate*
Adults_Pond_3(t) = Adults_Pond_3(t − dt) + (Colonizing_1_to_3 + Staying_at_3 − Adults_3_Dying) * dt
INIT Adults_Pond_3 = 0
INFLOWS:
Colonizing_1_to_3 = (Juveniles_Pond_1/7)*Dispersal_Through_Grass*0.04
Staying_at_3 = (Juveniles_Pond_3/7)*0.96
OUTFLOWS:
Adults_3_Dying = Adults_Pond_3*Adult_Death_Rate
Juveniles_Pond_1(t) = Juveniles_Pond_1(t − dt) + (Growing_1 − Colonizing_1_to_2 − Colonizing_1_to_3 − Staying_at_1 − Juveniles_1_Dying) * dt
INIT Juveniles_Pond_1 = 365
INFLOWS:
Growing_1 = DELAY(Larvae_Pond_1,0.25)
OUTFLOWS:
Colonizing_1_to_2 = (Juveniles_Pond_1/7)*Dispersal_Through_Forest*0.10
Colonizing_1_to_3 = (Juveniles_Pond_1/7)*Dispersal_Through_Grass*0.04
Staying_at_1 = (Juveniles_Pond_1/7)*0.86
Juveniles_1_Dying = Juveniles_Pond_1*Juvenile_Death_Rate
Juveniles_Pond_2(t) = Juveniles_Pond_2(t − dt) + (Growing_2 − Staying_at_2 − Juveniles_2_Dying − Colonizing_2_to_1) * dt
INIT Juveniles_Pond_2 = 0
INFLOWS:
Growing_2 = DELAY(Larvae_Pond_2,0.25)
OUTFLOWS:
Staying_at_2 = (Juveniles_Pond_2/7)*0.86
Juveniles_2_Dying = Juveniles_Pond_2*Juvenile_Death_Rate
Colonizing_2_to_1 = (Juveniles_Pond_2/7)*Dispersal_Through_Forest*0.10
Juveniles_Pond_3(t) = Juveniles_Pond_3(t − dt) + (Growing_3 − Colonizing_3_to_1 − Staying_at_3 − Juveniles_3_Dying) * dt
INIT Juveniles_Pond_3 = 0
INFLOWS:
Growing_3 = DELAY(Larvae_Pond_3,0.25)
OUTFLOWS:
Colonizing_3_to_1 = (Juveniles_Pond_3/7)*Dispersal_Through_Grass*0.04

Staying_at_3 = (Juveniles_Pond_3/7)∗0.96
Juveniles_3_Dying = Juveniles_Pond_3∗Juvenile_Death_Rate
Larvae_Pond_1(t) = Larvae_Pond_1(t − dt) + (Reproducing_1 − Growing_1 − Larvae_1_Dying) ∗ dt
INIT Larvae_Pond_1 = 0
INFLOWS:
Reproducing_1 = (.5∗Adults_Pond_1∗Number_of_Eggs∗Hatching_Success)∗(1−(Adults_Pond_1/Carrying_Capacity_1))
OUTFLOWS:
Growing_1 = DELAY(Larvae_Pond_1,0.25)
Larvae_1_Dying = Larvae_Pond_1∗Larvae_Death_Rate
Larvae_Pond_2(t) = Larvae_Pond_2(t − dt) + (Reproducing_2 − Growing_2 − Larvae_2_Dying) ∗ dt
INIT Larvae_Pond_2 = 0
INFLOWS:
Reproducing_2 = (.5∗Adults_Pond_2∗Number_of_Eggs∗Hatching_Success)∗(1−(Adults_Pond_2/Carrying_Capacity_2))
OUTFLOWS:
Growing_2 = DELAY(Larvae_Pond_2,0.25)
Larvae_2_Dying = Larvae_Pond_2∗Larvae_Death_Rate
Larvae_Pond_3(t) = Larvae_Pond_3(t − dt) + (Reproducing_3 − Growing_3 − Larvae_3_Dying) ∗ dt
INIT Larvae_Pond_3 = 0
INFLOWS:
Reproducing_3 = (.5∗Adults_Pond_3∗Number_of_Eggs∗Hatching_Success)∗(1−(Adults_Pond_3/Carrying_Capacity_3))
OUTFLOWS:
Growing_3 = DELAY(Larvae_Pond_3,0.25)
Larvae_3_Dying = Larvae_Pond_3∗Larvae_Death_Rate
Adult_Death_Rate = .17 {IF TIME >= 10 and TIME <= 15 THEN 1 ELSE 0.17}
Carrying_Capacity_1 = 274 {IF TIME >= 10 and TIME <= 15 THEN 1 ELSE 274}
Carrying_Capacity_2 = 180∗0+274
Carrying_Capacity_3 = 274
Dispersal_Through_Forest = 0.333
Dispersal_Through_Grass = 0.167
Hatching_Success = 0.27
Juvenile_Death_Rate = .43 {IF TIME >= 10 and TIME <= 15 THEN 1 ELSE 0.43}
Larvae_Death_Rate = .5 {IF TIME >= 10 and TIME <= 15 THEN 1 ELSE 0.5}
Number_of_Eggs = 125
Observerd_Pop = GRAPH(TIME)
(0.00, 236), (1.00, 236), (2.00, 240), (3.00, 260)

References

1. Rothermel BB, Semlitsch RD (2002) An experimental investigation of landscape resistance of forest versus old-field habitats to emigrating juvenile amphibians. Conserv Biol 16 (5):1324–1332
2. Blackwell EA, Cline GE, Marion KR (2004) Annual variation in population estimators for a southern population of ambystoma maculatum. Herpetologica 60(3):304–311

Chapter 39
Quail Movement

The Blue Jay's Lullaby Spiders and sowbugs and beetles and crickets, Slugs from the roses and ticks from the thickets, Grasshoppers, snails, and a quail's egg or two—All to be regurgitated for you. Lullaby, lullaby, swindles and schemes, Flying's not near as much fun as it seems.

(Peter S. Beagle, The Last Unicorn)

39.1 Quail Model

As in the previous chapter, we develop here a spatial model of population movement. The species of interest are northern bobwhite quail, which move between prairie patches that have been burned and exhibit different stages of succession. These patches are characterized by varying vegetative species and density. While bobwhite utilize different habitat types, they prefer open areas for most of their life activities like brood rearing, nesting, foraging, and roosting [1]. Sites that had been burned within a year or two produce greater amounts of food for bobwhite than older burns [2], thus bobwhite migrate to areas where the vegetation is the least dense (i.e. 0 years post burn). As a consequence of burning increases in bobwhite abundance can be observed, and these increases diminish after the vegetation recovered and bare ground decreased [3]. Regular disturbance, such as prescribed burning, is essential for a continued positive impact on northern bobwhite quail [1]. Effective burning tends to double the density of birds from 2.5 to 5 birds per ha [4].

Our model describes quail migration between four stocks labeled A, B, C, and D. Each stock represents a region in a different stage of prairie succession. Births of quail, named Birthing A, Birthing B, etc. in the model (Fig. 39.1) are the only

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_39,

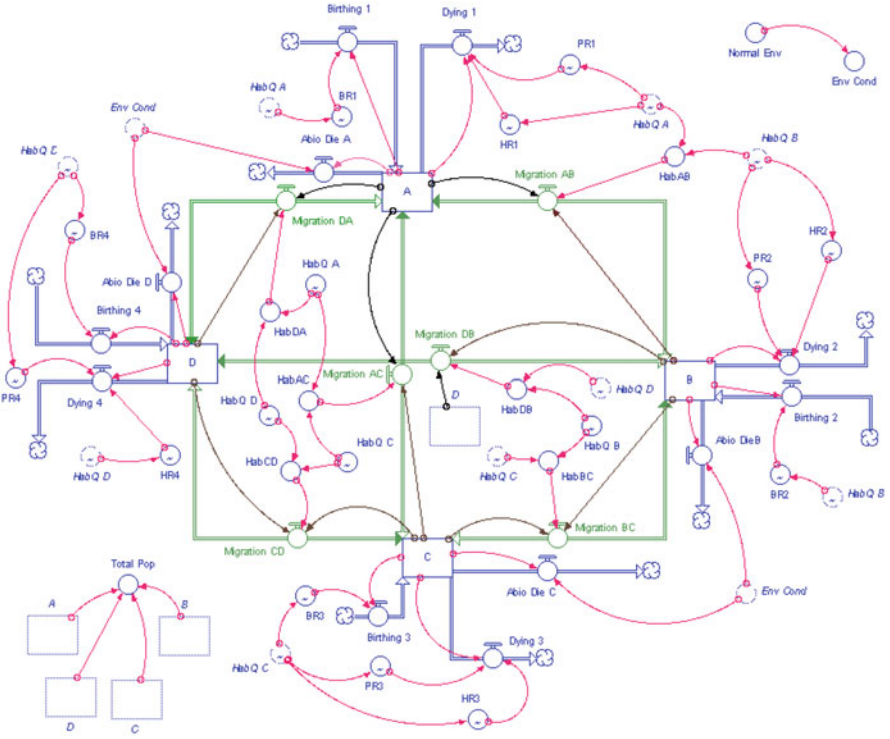

Fig. 39.1

population influx; immigration and emigration from outside the four regions are not included. Average clutch size is 14 eggs, with 40 % of the total number of females having at least one nest during the year. On average, 43.7 % of the nests survive incubation [5].

The physical pressures on quail that come from predation, hunting, and weather are incorporated into the model as dying outflows. Underlying the model structure are the habitat quality converters called Hab Q A, Hab Q B, etc. Converters for each stock start at a different habitat quality level (4 is the highest and 1 the lowest) and cycle through the four levels over 4 years using STELLA's built-in count function (simulating the succession conditions), illustrated in Fig. 39.2.

Habitat burning happens at the lowest point in the cycle, which quickly brings the habitat to optimal quality. In nature, predation is habitat dependent. In the best habitat (4), predation is highest, because there are fewest hiding spots and birds are generally more abundant, attracting more predators. Drawing on data from Burger et al. [6], we use graphical functions of the kind shown in Figs. 39.3 and 39.4 to specify, respectively, predation rates and hunting rates dependent on habitat quality. Similarly, birth rates are specified graphically, assuming that better habitat quality implies more food for larger, more successful clutches (Fig. 39.5). Here, however, the rate is delayed by half a time period to represent the time delay needed for juveniles to become reproducing adults.

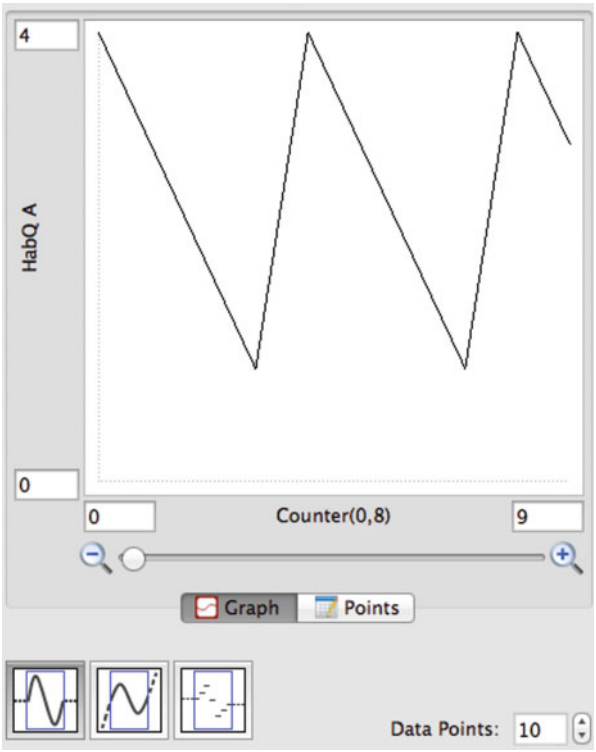

Fig. 39.2

Also included is death rate associated with abiotic factors. This is a stochastic weather function based on a normal distribution with a mean of 0 and an SD of 0.67:

$$\text{Normal Env} = \text{Normal}(0, .67) \tag{39.1}$$

When incorporated into the outflow that specifies deaths from abiotic influences, we assume that between 0 and 24 % of the population can be lost to weather each year in the standard model conditions:

$$\begin{aligned}\text{Env Cond} = \text{If Normal\_Env} > 0 \text{ AND Normal\_Env} \\ < .95 \text{ THEN Normal\_Env} * .25 \text{ ELSE } 0\end{aligned} \tag{39.2}$$

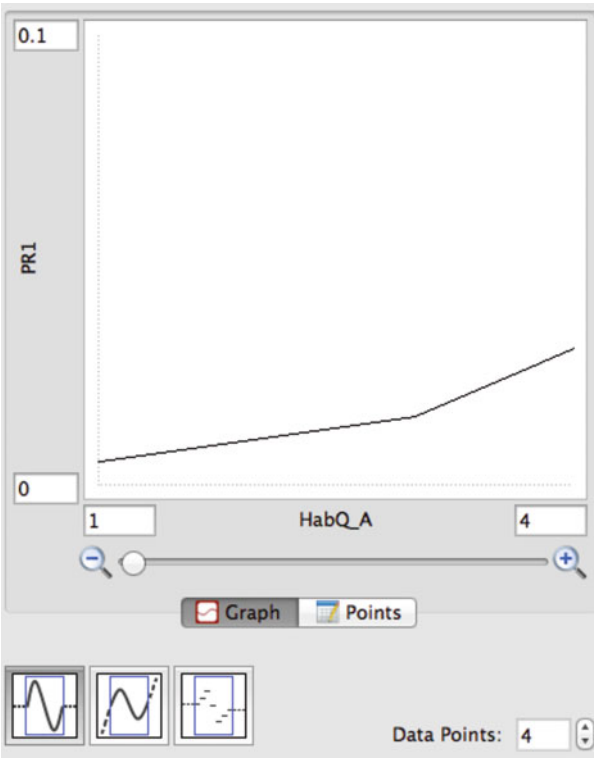

Fig. 39.3

Given our choice of initial stock sizes and parameter values, stable cyclic dynamics develop with population size fluctuating around 1,000 individuals (Fig. 39.6). These fluctuations are due to the stochastic weather function and vary between runs (Fig. 39.7). Changing the initial population in each of the stocks does not change the timing of that cycle, only its magnitude. The cycle is in delayed synchronization with the fluctuation of the habitat quality: peaks in stock population size are delayed by about 1.5 years following a peak in habitat quality (Fig. 39.8).

Slightly modify the parameters for hunting and predation-induced death rates and you will find that even small changes can have large impact on the population. Similarly, small modifications of weather-induced mortality can have devastating

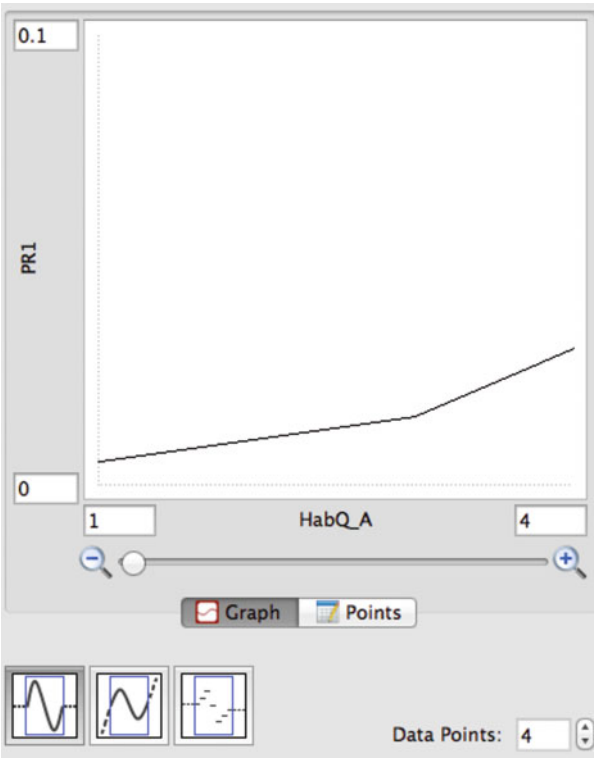

Fig. 39.4

consequences for bobwhite. In reality, with increasing human encroachment on their habitat and with climate change, both factors tend to coincide and move in the same direction.

Northern bobwhites breed twice a year. You may want to explicitly introduce into the model multiple age classes within a year and see the implications of their inclusion for the cycles we observed above, and for the long-term viability of our quail population under different biotic and abiotic pressures.

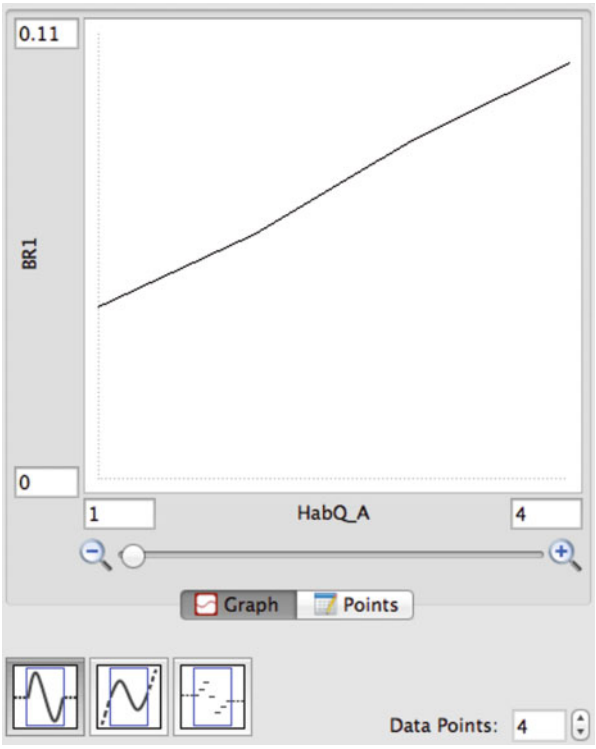

Fig. 39.5

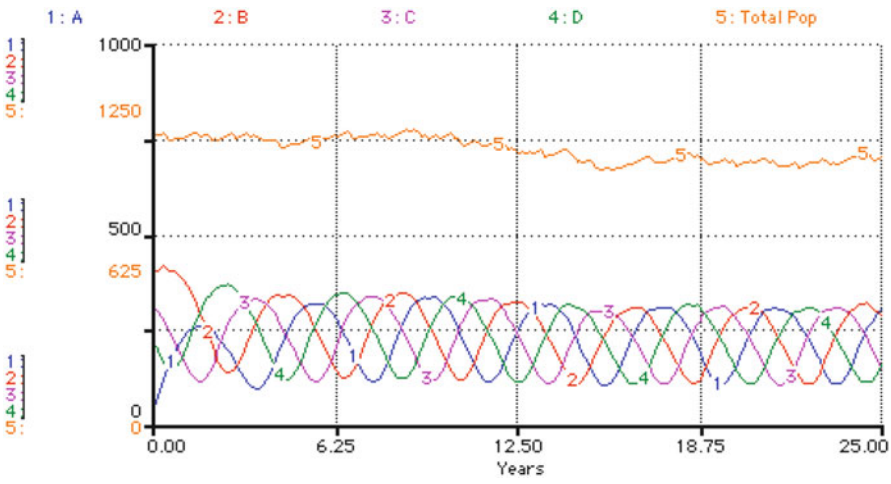

Fig. 39.6

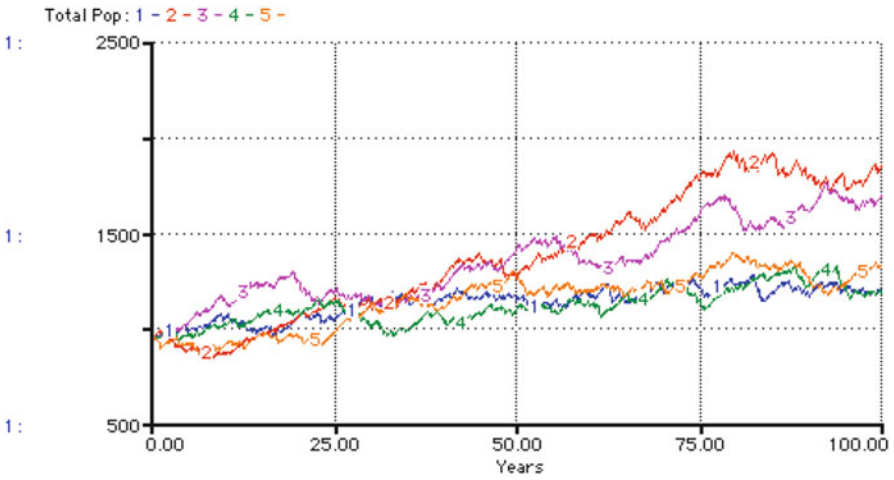

Fig. 39.7

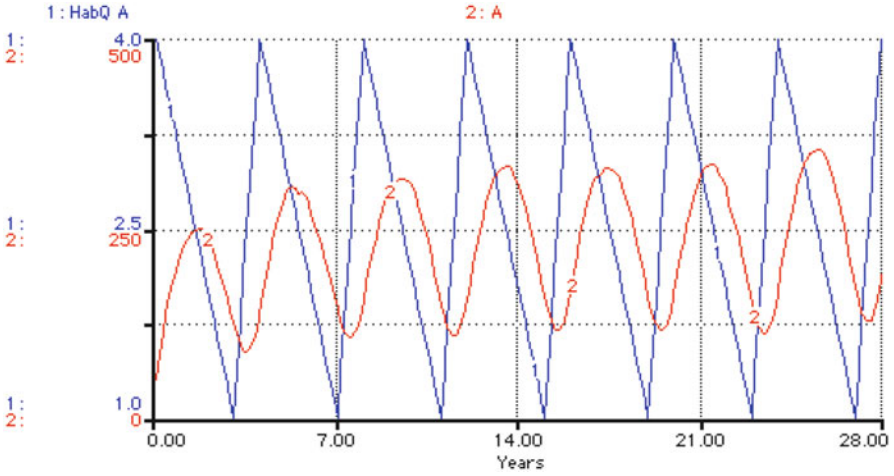

Fig. 39.8

39.2 Quail Model Equations

A(t) = A(t − dt) + (Birthing_1 + Migration_DA − Migration_AB − Dying_1 − Migration_AC − Abio_Die_A) ∗ dt
INIT A = 50
INFLOWS:
Birthing_1 = A∗Delay(BR1,.5)
Migration_DA = IF HabDA> 0 THEN −HabDA∗A∗.167 ELSE −HabDA∗D∗.167

OUTFLOWS:
Migration_AB = IF HabAB >0 THEN −HabAB∗B∗.167 ELSE −HabAB∗A∗.167
Dying_1 = A∗(HR1+PR1)
Migration_AC = IF HabAC> 0 THEN −HabAC∗C∗.167 ELSE −HabAC∗A∗.167
Abio_Die_A = A∗Env_Cond
B(t) = B(t − dt) + (Migration_AB + Migration_DB + Birthing_2 − Migration_BC − Dying_2 − Abio_DieB) ∗ dt
INIT B = 400
INFLOWS:
Migration_AB = IF HabAB >0 THEN −HabAB∗B∗.167 ELSE −HabAB∗A∗.167
Migration_DB = IF HabDB> 0 THEN −HabDB∗B∗.167 ELSE −HabDB∗D∗.167
Birthing_2 = Delay(BR2,.5)∗B
OUTFLOWS:
Migration_BC = IF HabBC > 0 THEN -HabBC∗C∗.167 ELSE −HabBC∗B∗.167
Dying_2 = B∗(HR2+PR2)
Abio_DieB = B∗Env_Cond
C(t) = C(t − dt) + (Migration_BC + Birthing_3 + Migration_AC − Migration_CD − Dying_3 − Abio_Die_C) ∗ dt
INIT C = 300
INFLOWS:
Migration_BC = IF HabBC > 0 THEN −HabBC∗C∗.167 ELSE −HabBC∗B∗.167
Birthing_3 = C∗Delay(BR3,.5)
Migration_AC = IF HabAC> 0 THEN −HabAC∗C∗.167 ELSE −HabAC∗A∗.167
OUTFLOWS:
Migration_CD = IF HabCD> 0 THEN −HabCD∗D∗.167 ELSE −HabCD∗C∗.167
Dying_3 = C∗(HR3+PR3)
Abio_Die_C = C∗Env_Cond
D(t) = D(t − dt) + (Migration_CD + Birthing_4 − Migration_DA − Migration_DB − Dying_4 − Abio_Die_D) ∗ dt
INIT D = 200
INFLOWS:
Migration_CD = IF HabCD> 0 THEN −HabCD∗D∗.167 ELSE −HabCD∗C∗.167
Birthing_4 = D∗Delay(BR4,.5)
OUTFLOWS:
Migration_DA = IF HabDA> 0 THEN −HabDA∗A∗.167 ELSE -HabDA∗D∗.167
Migration_DB = IF HabDB> 0 THEN −HabDB∗B∗.167 ELSE −HabDB∗D∗.167
Dying_4 = D∗(HR4+PR4)
Abio_Die_D = D∗Env_Cond
BR1 = GRAPH(HabQ_A)
(1.00, 0.042), (2.00, 0.06), (3.00, 0.083), (4.00, 0.102)
BR2 = GRAPH(HabQ_B)
(1.00, 0.042), (2.00, 0.06), (3.00, 0.083), (4.00, 0.102)

BR3 = GRAPH(HabQ_C)
(1.00, 0.042), (2.00, 0.06), (3.00, 0.083), (4.00, 0.102)
BR4 = GRAPH(HabQ_D)
(1.00, 0.042), (2.00, 0.06), (3.00, 0.083), (4.00, 0.102)
Env_Cond = If Normal_Env >0 AND Normal_Env <.95 THEN Normal_Env∗.25 ELSE 0
HabAB = HabQ_A−HabQ_B
HabAC = HabQ_A−HabQ_C
HabBC = HabQ_B−HabQ_C
HabCD = HabQ_C−HabQ_D
HabDA = HabQ_D−HabQ_A
HabDB = HabQ_D−HabQ_B
HabQ_A = GRAPH(Counter(0,8))
(0.00, 4.00), (1.00, 3.00), (2.00, 2.00), (3.00, 1.00), (4.00, 4.00), (5.00, 3.00), (6.00, 2.00), (7.00, 1.00), (8.00, 4.00), (9.00, 3.00)
HabQ_B = GRAPH(Counter(0,8))
(0.00, 3.00), (1.00, 2.00), (2.00, 1.00), (3.00, 4.00), (4.00, 3.00), (5.00, 2.00), (6.00, 1.00), (7.00, 4.00), (8.00, 3.00), (9.00, 2.00)
HabQ_C = GRAPH(Counter(0,8))
(0.00, 2.00), (1.00, 1.00), (2.00, 4.00), (3.00, 3.00), (4.00, 2.00), (5.00, 1.00), (6.00, 4.00), (7.00, 3.00), (8.00, 2.00), (9.00, 1.00)
HabQ_D = GRAPH(Counter(0,8))
(0.00, 1.00), (1.00, 4.00), (2.00, 3.00), (3.00, 2.00), (4.00, 1.00), (5.00, 4.00), (6.00, 3.00), (7.00, 2.00), (8.00, 1.00), (9.00, 4.00)
HR1 = GRAPH(HabQ_A)
(1.00, 0.003), (2.00, 0.01), (3.00, 0.02), (4.00, 0.04)
HR2 = GRAPH(HabQ_B)
(1.00, 0.003), (2.00, 0.01), (3.00, 0.02), (4.00, 0.04)
HR3 = GRAPH(HabQ_C)
(1.00, 0.003), (2.00, 0.01), (3.00, 0.02), (4.00, 0.04)
HR4 = GRAPH(HabQ_D)
(1.00, 0.003), (2.00, 0.01), (3.00, 0.02), (4.00, 0.04)
Normal_Env = Normal(0,.67)
PR1 = GRAPH(HabQ_A)
(1.00, 0.005), (2.00, 0.01), (3.00, 0.015), (4.00, 0.03)
PR2 = GRAPH(HabQ_B)
(1.00, 0.005), (2.00, 0.01), (3.00, 0.015), (4.00, 0.03)
PR3 = GRAPH(HabQ_C)
(1.00, 0.005), (2.00, 0.01), (3.00, 0.015), (4.00, 0.03)
PR4 = GRAPH(HabQ_D)
(1.00, 0.005), (2.00, 0.01), (3.00, 0.015), (4.00, 0.03)
Total_Pop = A+B+C+D

References

1. Greenfield KC, Chamberlain MJ, Burger LW Jr, Kurzejeski EW (2003) Effects of burning and discing conservation reserve program fields to improve habitat quality for northern bobwhite (Colinus virginianus). Am Midland Nat 149(2):344–353
2. Reid VH (1953) Multiple land use: timber, cattle, and bobwhite quail. North Am Wildl Conf 18:412–420
3. Wilson MM, Crawford JA (1979) Response of bobwhites to controlled burning in south Texas. Wildl Soc Bull 7(1):53–56
4. Stoddard HL (1931) The bobwhite quail: its habits, preservation, and increase. Scribner's, New York, p 559
5. Burger L, Ryan M, Dailey T, Kurzweski E (1995) Reproductive strategies, success, and mating systems of northern bobwhite in Missouri. J Wildl Manag 59(3):417–426
6. Burger L, Dailey T, Kurzejeski E, Ryan M (1995) Survival and cause−specific mortality of northern bobwhite. J Wildl Manag 59(2):401–410

Chapter 40
Modeling Spatial Dynamics of Predator–Prey Interactions in a Changing Environment

It is a sad fact that several of our most noble birds of prey can no longer be studied in what were once their native haunts.

(D. A. Bannerman Birds Brit. Isles, 1956)

40.1 Spatial Predator–Prey Model

In the previous chapter we have modeled the spatial dynamics of species that move across the landscape to take advantage of differences in habitat characteristics. Let us now model the case of spatial predator–prey interactions in a changing environment. To model the spatial aspect of these interactions, we define four subdivisions of the landscape as laid out in Fig. 40.1.

Cell 1 is occupied with 1,000 predators and 10,000 prey. There are no predators or prey in the other three cells. The birth rates for predators and prey are given exogenously, yet the number of births depends on predator–prey interaction and the carrying capacity of their ecosystem. Similarly, the deaths of predators and prey depend on their interaction—the prey consumed by predators. For example, the births and deaths of predators and prey in cell 1 are defined as

$$\text{BIRTH PRED 1} = \text{BR PRED} * (\text{PRED 1} - \text{DEATH PRED 1}) \tag{40.1}$$

$$\text{DEATH PRED 1} = \text{PRED 1} - \text{CONSUME 1/CONSUME RATE} \tag{40.2}$$

$$\text{BIRTH PREY 1} = (\text{PREY 1} - \text{DEATH PREY 1}) * \text{BR PREY 1} * (1 - \text{PREY 1/CC PREY 1}) \tag{40.3}$$

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_40,

1	2
3	4

Fig. 40.1

$$\text{DEATH PREY 1} = \text{CONSUME 1} \tag{40.4}$$

where PREY 1 and PRED 1 are the stock of the two populations in cell 1, BR PREY 1 is the birth rate of prey in cell 1, CONSUME 1 is the number of prey consumed by predators in cell 1. The number of prey consumed in cell 1 cannot exceed the number of prey in that cell and is at least zero and at most the consumption rate times the number of predators in that cell. The amount of prey consumed by predators in cell 1 is defined as

$$\text{CONSUME 1} = \text{MIN}(\text{PREY 1}, \text{CONSUME RATE} * \text{PRED 1}) \tag{40.5}$$

CC PREY 1 is the carrying capacity for prey in cell 1. The latter is assumed to change along a sinewave with an exogenously defined CYCLE TIME:

$$\text{CC PREY 1} = 2250 * \text{SIN}\,(2 * \text{PI} * \text{TIME}/\text{CYCLE TIME}) + 2500 \tag{40.6}$$

In addition, we assumed that the carrying capacity for each individual patch was out of phase by 90° with the "previous" patch, i.e.

$$\text{CC PREY 2} = 2250 * \text{SIN}\,(2 * \text{PI} * \text{TIME}/\text{CYCLE\_TIME} - \text{PI}/2) + 2500 \tag{40.7}$$

$$\text{CC PREY 3} = 2250 * \text{SIN}\,(2 * \text{PI} * \text{TIME}/\text{CYCLE\_TIME} - 3 * \text{PI}/2) + 2500 \tag{40.8}$$

$$\text{CC PREY 4} = 2250 * \text{SIN}\,(2 * \text{PI} * \text{TIME}/\text{CYCLE\_TIME} - \text{PI}) + 2500 \tag{40.9}$$

Prey migrate routinely regardless of their population in the starting or the receiving cells, and the predators migrate to a new cell when they begin to starve in their current cell. Once the migration quantity is established, a random process determines its distribution to adjacent cells. This simple idea enables the prey to "escape" to a neighboring cell where the predator population may be at a relatively low level.

The model for cell 1 of the 4-cell predator–prey model is shown in Fig. 40.2. Predators and prey can move to the right (RT) and down (DN). The fraction of predators and prey that move is a random number between zero and one. For example, the migration of predators from cell one to any of the other cells is defined as

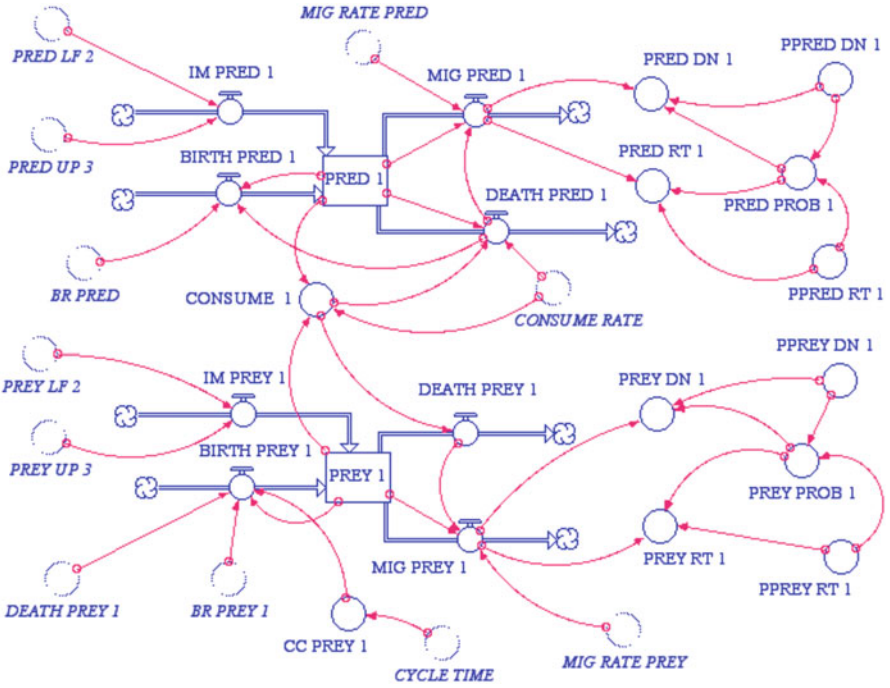

Fig. 40.2

$$\text{MIG PRED 1} = \text{MIN}\big(\text{PRED 1} - \text{DEATH PRED 1}, \text{MIG RATE PRED} * \text{DEATH PRED 1}\big) \tag{40.10}$$

The migration rate of predators, MIG RATE PRED, depends on the availability of prey in the cell. For simplicity, we assume that the number of predators that migrate is the product of the number of starvation deaths and the migration rate of predators. In the model, we define this number as a constant.

How do the predator and prey populations vary as the cycle time is varied on the carrying capacity? For what values of the carrying capacity will the predator and prey populations have the lowest standard deviation?

Run several trials with cycle time being varied. To evaluate which cycle time produces the most stable populations, you must look for the least value of the standard deviation. For the predator population, a cycle time of 8 produces a low standard deviation of approximately 630. Similarly, for the prey population a cycle time of 1 produces a low standard deviation of about 2,242. Are these the lowest standard deviations one can find? Run the model many times for the same cycle time and observe the results.

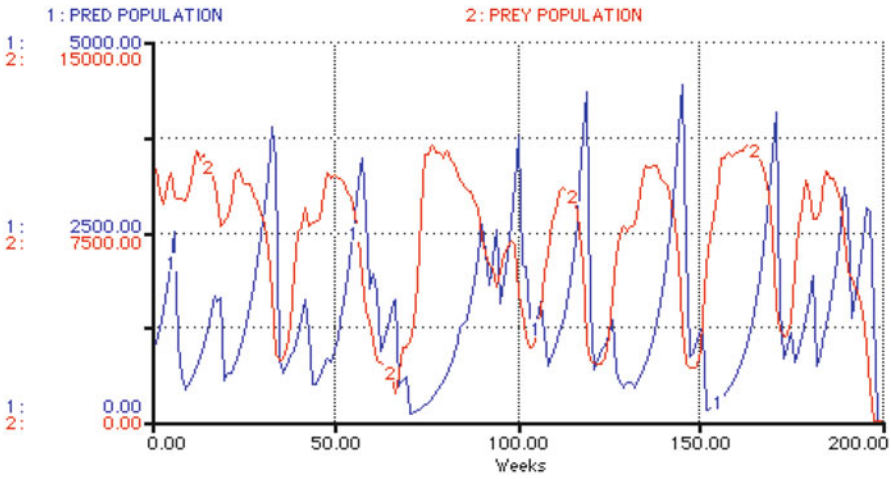

Fig. 40.3

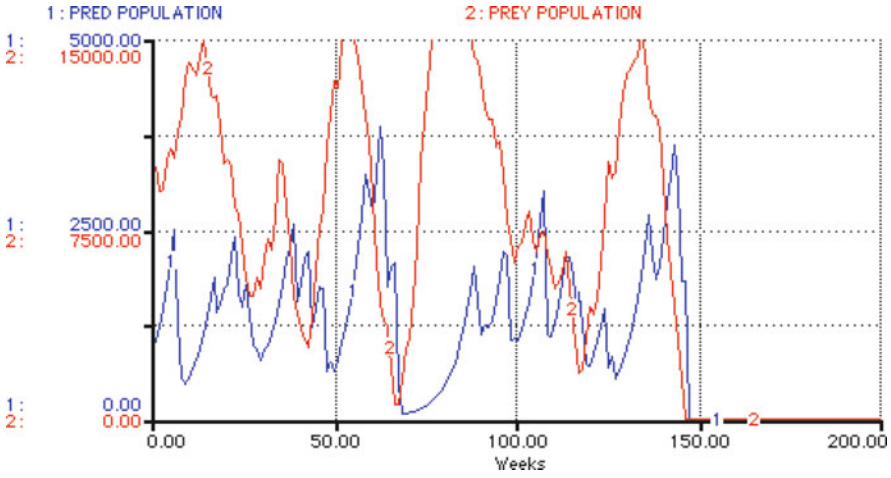

Fig. 40.4

The two cases of CYCLE TIME = 1 and CYCLE TIME = 8 are shown, respectively, in Figs. 40.3 and 40.4. Admittedly, even in the best cases, the population still varies significantly over time. This variation can best be minimized by increasing the number of patches that describe the ecosystem.

By holding the phase difference of the sinusoidal carrying capacity constant between patches, will a variation of frequency between patches create a more stable ecosystem? Can a similar phase difference that was used in this model be applied to a spatial model with nine or more patches? Can variations in other parameters, including birth rate, migration rate, and consumption rate, create the desired effects of this spatial predator–prey model? Try to find answers to these questions.

40.2 Spatial Predator–Prey Model Equations

PRED_1(t) = PRED_1(t − dt) + (BIRTH_PRED_1 + IM_PRED_1 − MIG_PRED_1 − DEATH_PRED_1) ∗ dt
INIT PRED_1 = 100.
INFLOWS:
BIRTH_PRED_1 = BR_PRED∗(PRED_1−DEATH_PRED_1)
IM_PRED_1 = PRED_LF_2+PRED_UP_3
OUTFLOWS:
MIG_PRED_1 = MIN(PRED_1−DEATH_PRED_1,MIG_RATE_PRED∗DEATH_PRED_1)
DEATH_PRED_1 = PRED_1−CONSUME__1/CONSUME_RATE
PRED_2(t) = PRED_2(t − dt) + (BIRTH_PRED_2 + IM_PRED_2 − MIG_PRED_2 − DEATH_PRED_2) ∗ dt
INIT PRED_2 = 0
INFLOWS:
BIRTH_PRED_2 = BR_PRED∗(PRED_2−DEATH_PRED_2)
IM_PRED_2 = PRED_RT_1+PRED_UP_2
OUTFLOWS:
MIG_PRED_2 = MIN(PRED_2−DEATH_PRED_2,MIG_RATE_PRED∗DEATH_PRED_2)
DEATH_PRED_2 = PRED_2−CONSUME_2/CONSUME_RATE
PRED_3(t) = PRED_3(t − dt) + (BIRTH_PRED_7 + IM_PRED_7 − MIG_PRED_3 − DEATH_PRED_3) ∗ dt
INIT PRED_3 = 0
INFLOWS:
BIRTH_PRED_7 = BR_PRED∗(PRED_3−DEATH_PRED_3)
IM_PRED_7 = PRED_DN_1+PRED_LF_4
OUTFLOWS:
MIG_PRED_3 = MIN(PRED_3−DEATH_PRED_3,DEATH_PRED_3∗MIG_RATE_PRED)
DEATH_PRED_3 = PRED_3-CONSUME_3/CONSUME_RATE
PRED_4(t) = PRED_4(t − dt) + (BIRTH_PRED_4 + IM_PRED_4 − MIG_PRED_4 − DEATH_PRED_4) ∗ dt
INIT PRED_4 = 0
INFLOWS:
BIRTH_PRED_4 = BR_PRED∗(PRED_4−DEATH_PRED_4)
IM_PRED_4 = PRED_DN_2+PRED_RT_3
OUTFLOWS:
MIG_PRED_4 = MIN(PRED_4−DEATH_PRED_4,DEATH_PRED_4∗MIG_RATE_PRED)
DEATH_PRED_4 = PRED_4−CONSUME_4/CONSUME_RATE
PREY_1(t) = PREY_1(t − dt) + (BIRTH_PREY_1 + IM_PREY_1 − DEATH_PREY_1 − MIG_PREY_1) ∗ dt
INIT PREY_1 = 1000.

INFLOWS:
BIRTH_PREY_1 = (PREY_1−DEATH_PREY_1)∗BR_PREY_1∗(1−PREY_1/CC_PREY_1)
IM_PREY_1 = PREY_LF_2+PREY_UP_3
OUTFLOWS:
DEATH_PREY_1 = CONSUME__1
MIG_PREY_1 = MIG_RATE_PREY∗(PREY_1−DEATH_PREY_1)
PREY_2(t) = PREY_2(t − dt) + (BIRTH_PREY_2 + IM_PREY_2 − DEATH_PREY_2 − MIG_PREY_2) ∗ dt
INIT PREY_2 = 0
INFLOWS:
BIRTH_PREY_2 = (PREY_2−DEATH_PREY_2)∗BR_PREY_1∗(1−PREY_2/CC_PREY_2)
IM_PREY_2 = PREY_RT_1+PREY_UP_4
OUTFLOWS:
DEATH_PREY_2 = CONSUME_2
MIG_PREY_2 = MIG_RATE_PREY∗(PREY_2−DEATH_PREY_2)
PREY_3(t) = PREY_3(t − dt) + (BIRTH_PREY_3 + IM_PREY_3 − DEATH_PREY_3 − MIG_PREY_3) ∗ dt
INIT PREY_3 = 0
INFLOWS:
BIRTH_PREY_3 = (PREY_3−DEATH_PREY_3)∗BR_PREY_1∗(1−PREY_3/CC_PREY_3)
IM_PREY_3 = PREY_DN_1+PREY_LF_4
OUTFLOWS:
DEATH_PREY_3 = CONSUME_3
MIG_PREY_3 = MIG_RATE_PREY∗(PREY_3−DEATH_PREY_3)
PREY_4(t) = PREY_4(t − dt) + (IM_PREY_4 + BIRTH_PREY_4 − DEATH_PREY_4 − MIG_PREY_4) ∗ dt
INIT PREY_4 = 0
INFLOWS:
IM_PREY_4 = PREY_DN_2+PREY_RT_3
BIRTH_PREY_4 = (PREY_4−DEATH_PREY_4)∗BR_PREY_1∗(1−PREY_4/CC_PREY_4)
OUTFLOWS:
DEATH_PREY_4 = CONSUME_4
MIG_PREY_4 = MIG_RATE_PREY∗(PREY_4−DEATH_PREY_4)
BR_PRED = 0.2
BR_PREY_1 = 2
CC_PREY_1 = 2250∗SIN(2∗PI∗TIME/CYCLE_TIME)+2500
CC_PREY_2 = 2250∗SIN(2∗PI∗TIME/CYCLE_TIME−PI/2)+2500
CC_PREY_3 = 2250∗SIN(2∗PI∗TIME/CYCLE_TIME−3∗PI/2)+2500
CC_PREY_4 = 2250∗SIN(2∗PI∗TIME/CYCLE_TIME−PI)+2500
CONSUME_2 = MIN(PREY_2,CONSUME_RATE∗PRED_2)

CONSUME_3 = MIN(PREY_3,CONSUME_RATE*PRED_3)
CONSUME_4 = MIN(PREY_4,CONSUME_RATE*PRED_4)
CONSUME_RATE = 1
CONSUME__1 = MIN(PREY_1,CONSUME_RATE*PRED_1)
CYCLE_TIME = 8
MIG_RATE_PRED = .05 {this times the number of starvation deaths is the number that migrate}
MIG_RATE_PREY = .1 {this is the proportion of the prey that migrate}
PPRED_DN_1 = RANDOM(0,1)
PPRED_DN_2 = RANDOM(0,1)
PPRED_LF_2 = RANDOM(0,1)
PPRED_LF_4 = RANDOM(0,1)
PPRED_RT_1 = RANDOM(0,1)
PPRED_RT_3 = RANDOM(0,1)
PPRED_UP_3 = RANDOM(0,1)
PPRED_UP_4 = RANDOM(0,1)
PPREY_DN_1 = RANDOM(0,1)
PPREY_DN_2 = RANDOM(0,1)
PPREY_LF_2 = RANDOM(0,1)
PPREY_LF_4 = RANDOM(0,1)
PPREY_RT_1 = RANDOM(0,1)
PPREY_RT_3 = RANDOM(0,1)
PPREY_UP_3 = RANDOM(0,1)
PPREY_UP_4 = RANDOM(0,1)
PRED_DN_1 = MIG_PRED_1*PPRED_DN_1/PRED_PROB_1
PRED_DN_2 = MIG_PRED_2*PPRED_DN_2/PRED_PROB_2
PRED_LF_2 = MIG_PRED_2*PPRED_LF_2/PRED_PROB_2
PRED_LF_4 = MIG_PRED_4*PPRED_LF_4/PRED_PROB_4
PRED_POPULATION = PRED_1+PRED_2+PRED_3+PRED_4
PRED_PROB_1 = PPRED_RT_1+PPRED_DN_1
PRED_PROB_2 = PPRED_LF_2+PPRED_DN_2
PRED_PROB_3 = PPRED_RT_3+PPRED_UP_3
PRED_PROB_4 = PPRED_LF_4+PPRED_UP_4
PRED_RT_1 = MIG_PRED_1*PPRED_RT_1/PRED_PROB_1
PRED_RT_3 = MIG_PRED_3*PPRED_RT_3/PRED_PROB_3
PRED_UP_2 = MIG_PRED_4*PPRED_UP_4/PRED_PROB_4
PRED_UP_3 = MIG_PRED_3*PPRED_UP_3/PRED_PROB_3
PREY_DN_1 = MIG_PREY_1*PPREY_DN_1/PREY_PROB_1
PREY_DN_2 = MIG_PREY_2*PPREY_DN_2/PREY_PROB_2
PREY_LF_2 = MIG_PREY_2*PPREY_LF_2/PREY_PROB_2
PREY_LF_4 = MIG_PREY_4*PPREY_LF_4/PREY_PROB_4
PREY_POPULATION = PREY_1+PREY_3+PREY_2+PREY_4
PREY_PROB_1 = PPREY_RT_1+PPREY_DN_1
PREY_PROB_2 = PPREY_LF_2+PPREY_DN_2

PREY_PROB_3 = PPREY_RT_3+PPREY_UP_3
PREY_PROB_4 = PPREY_LF_4+PPREY_UP_4
PREY_RT_1 = MIG_PREY_1*PPREY_RT_1/PREY_PROB_1
PREY_RT_3 = MIG_PREY_3*PPREY_RT_3/PREY_PROB_3
PREY_UP_3 = MIG_PREY_3*PPREY_UP_3/PREY_PROB_3
PREY_UP_4 = MIG_PREY_4*PPREY_UP_4/PREY_PROB_4

Part VII
Catastrophe and Self-Organization

Chapter 41
Catastrophe

The choice of the name, catastrophe theory, is unfortunate as it denotes abnormal nasty events. What we have come to realize is that such events are normal and necessary for the continued smooth functioning of many systems.

(E.D. Schneider and J.J. Kay. 1994. Complexity and Thermodynamics, Futures, Vol. 26, p. 641)

41.1 Catastrophe Model

If a large number of real systems exhibit dynamics that bear the potential for chaos, why do we not see more chaos in real-world processes? Fortunately, the domains over which stability of the system occurs can be relatively large. But once in a while, systems may move "towards the edge of stability" and little nudges to the system may move it from stability to instability—that is, into a catastrophe. Subsequently, reorganization of system components may occur to bring the system back into a stable domain—a kind of evolutionary process. This stable domain, however, may not be the same as the one prior to the disturbance.

The system undergoes a catastrophic event in the sense that it is moved from an initial state of stability through a dramatic phase of reorganization and back to some degree of stability. Examples for such catastrophic events include landslides, avalanches, earthquakes, and pest outbreaks in ecosystems. In each case, small changes in the system occur that individually may not be critical to the system's behavior. Collectively, however, they lead to the evolution of the system towards a critical state. This is apparent, for example, in the case of avalanches.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_41,

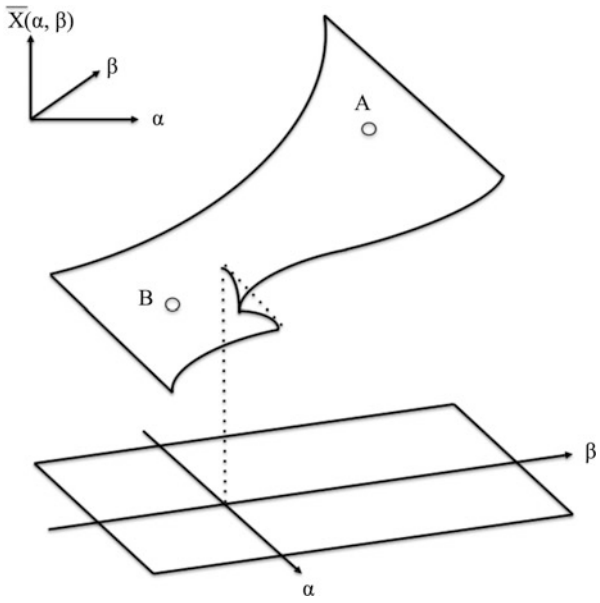

Fig. 41.1

Each individual snow flake potentially adds to the instability of the system. When a critical point is reached, the next snow flake may trigger an avalanche that affects a large part of the system. Temporary stability is quickly reached if the avalanche was not too dramatic. Even if not of a large scale, the avalanche adds to the "stress" of the system downhill, making it more susceptible to further avalanches as more snow falls at those regions or as additional small avalanches are received from higher on the hill. Ultimately, a large-scale, catastrophic event may occur which affects the entire system, not just individual regions. The system components re-group and finally enter a phase of new, temporary stability.

So, evolutionary processes are at work making the system more "efficient" in some sense. This is evolution toward catastrophe. A system in such a state can remerge to a stable state by another process of evolution, likely faster than the first kind, and this new stable state may not be very efficient. Large living natural systems are likely constrained from operating at or near peak efficiency by random intervention of uncoordinated external processes at the regional levels.

In this chapter we develop a simple model of catastrophe. Before we develop the models, consider Fig. 41.1 that illustrates the surface defined by the following equation [1]:

$$X^3 - \text{ALPHA} * X - \text{BETA} = 0 \tag{41.1}$$

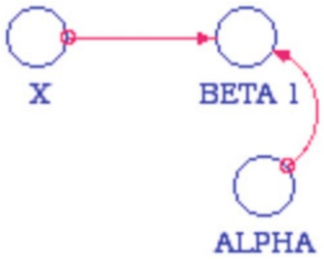

Fig. 41.2

Imagine a ball lying at the top of this surface, such as at point A. The ball may lie still, and very small nudges away from its equilibrium point A lead to a new equilibrium. After a series of such small perturbations, however, the ball will roll off the top part of the surface, and a priori, it is difficult for us to determine exactly where it will end up. All we know for certain is that the new equilibrium position is somewhere at the bottom of the surface, say point B.

Small nudges to the ball in B will again move it slightly away from B. And if we kick it hard enough, we can propel the ball through the fold, or "cusp," to the upper part of the surface again. Where exactly will it end up? To give a precise answer requires exact knowledge of the shape of the surface, the properties of the ball, and the magnitude and direction of the force exerted on the ball. In more complicated, real-life systems, not all the variables to describe the system and the forces incident upon them are well enough known. As a result, we may only know stability domains rather than specific locations.

The STELLA model for Eq. (41.1) above is given in Fig. 41.2. (Note that we named our variable BETA 1 in the diagram because STELLA already uses the name BETA for a built-in function.) We slightly vary X with each simulation time step. Solve Eq. (41.1) for BETA. Set DT = 0.0025 and define X, for example, as

$$\mathrm{X} = \mathrm{TIME} - 2, \tag{41.2}$$

then run the model.

For positive ALPHA, the cusp or fold appears in the X vs. BETA plot. For ALPHA = 0 and negative ALPHA, the "S" curve appears. Figure 41.3 shows the results for BETA = −3, BETA = 0, and BETA = 3, respectively.

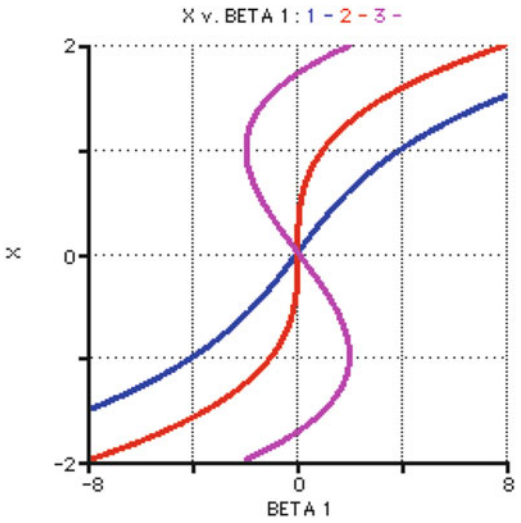

Fig. 41.3

41.2 Catastrophe Model Equations

ALPHA $= -1$
BETA_1 $=$ X^3 $-$ ALPHA$*$X
X $=$ TIME $-$ 2

Reference

1. Beltrami E (1987) Mathematics for dynamic modeling. Academic Press, Boston

Chapter 42
Spruce Budworm Dynamics

Books are subject among other Chances to fire, and the Worme.

(Whitlock Zootomia, 1654)

42.1 Spruce Budworm Model

A classical example for the implications of catastrophes for ecosystem management is the spruce budworm dynamics. Spruce budworm is a caterpillar that feeds on spruce and fir forests in the northeastern United States and eastern Canada. For many years, population sizes of spruce budworm are low and have little impact on trees. When forest stands reach maturity, however, spruce budworm populations explode, seriously affecting the forest by defoliating the trees. As a result of defoliation, trees are weakened and ultimately may die. With the death of trees comes a loss of the food source for spruce budworm and a consequent population crash.

The cycle of low spruce budworm population densities, followed by population explosions and catastrophic collapse tend to repeat themselves over the course of years. The resulting damage and death of trees affects negatively the timber and paper pulp industries of the region. Frequently, forest managers decided to spray forest stands to control budworm populations. The dynamics inherent in the system, however, lead it to follow its own path, making ever more extensive pest control necessary. If those controls fail, outbreaks are more sever and devastating than if the system had been left to control itself, as recent experiences in the US and Canada show.

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_42,

One of the natural system controls not now used in forestry relates to the idea of patch size. Natural systems no doubt avoid large catastrophes because they operate in patches, where the degree of maturity of adjacent patches is nearly always different. Consequently, pests and fires find difficulty in spreading beyond a patch and the size of the catastrophe is kept small. Current forestry practice seems to be disconnected from such natural system behavior. Our model is not a regional one and so such interpatch dynamics are not captured. However, we have modeled such patch dynamics in Chap. 40, and you may want to combine the approach of that chapter with the one developed here.

To model the spruce budworm catastrophe, we closely Beltrami [1] and denote B as the budworm population size, K as the carrying capacity, S as habitat size, and GR as the budworm's natural rate of increase. Thus,

$$\frac{dB}{dt} = GR * B * \left(1 - \frac{B}{K * S}\right) \tag{42.1}$$

would describe the population dynamics for a fixed carrying capacity and no predatory influences on population growth. This is the logistic growth equation that we have used in this book many times before. Let us introduce the effects of predation with a maximum predation rate C, which is assumed to be constant. At small population densities, predation has only little effects on the budworm population because they are well-hidden in a relatively dense canopy. As population densities increase, however, predators may increasingly feed on budworm that partially or totally defoliated the trees and are then easy prey. A predation term that captures such interactions is

$$\frac{C * B^2}{A^2 + B^2}, \tag{42.2}$$

with A as a scalar that captures the effectiveness of the predators to spot and prey on spruce budworm. In an immature forest, predation is easier than in a mature forest with diverse and dense canopy. Thus, A may be assumed to increase with increased maturity of the forest, i.e. habitat size S

$$A = K1 * S \tag{42.3}$$

and thus

$$\frac{C * B^2}{A^2 + B^2} = \frac{C * B^2}{(K1 + S)^2 + B^2}, \tag{42.4}$$

with K1 as a constant.

Combining predation with the logistic growth function yields

$$\Delta B = \frac{dB}{dt} = GR * B * \left(1 - \frac{B}{K * S}\right) - \frac{C * B^2}{(K1 + S)^2 + B^2}, \tag{42.5}$$

which is the equation used in the model to drive spruce budworm population changes, ΔB.

Changes in habitat size are assumed to also follow the logistic growth curve, with RS as the natural rates of increase and KS as carrying capacity:

$$\Delta S = \frac{dS}{dt} = RS * S * \left(1 - \frac{BS}{KS * E}\right) \tag{42.6}$$

E is the percentage of foliage on trees. The more healthy the forest, the higher is E. The percentage of foliage on trees is assumed to decrease as the average budworm density per habitat size B/S increases. To model diminishing stress as budworm populations decrease we multiply B/S by E^2. The combined effect of logistic growth in foliage and budworm-induced foliage losses is

$$\Delta E = \frac{dE}{dt} = RE * E * (1 - E) - \frac{P * B * E^2}{S}, \tag{42.7}$$

with RE the rate of foliage increase and P a proportionality factor.

Let us consider the case of $B \neq 0$ and introduce the following notation:

$$R = \frac{GR * K1 * S}{C} \tag{42.8}$$

$$Q = \frac{K}{K1}, \tag{42.9}$$

and rewrite

$$B = K1 * S * X. \tag{42.10}$$

It can be shown [1] that the nontrivial equilibria of Eq. (42.5) fulfill

$$R\left(1 - \frac{X}{Q}\right) = F(X) = G(X) \tag{42.11}$$

with

$$G(X) = \frac{X}{1 + X^2}. \tag{42.12}$$

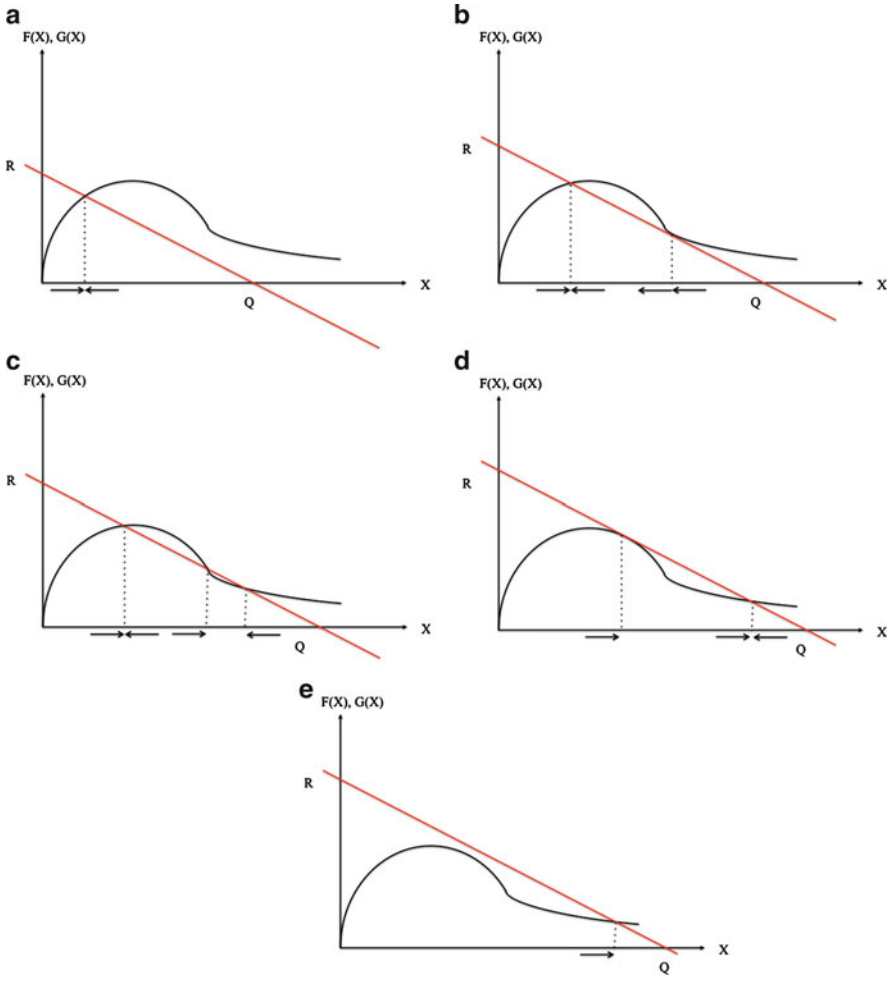

Fig. 42.1

The left side of Eq. (42.11) is a straight line F(X) with slope $-R'/Q$. Equilibria occur where this line intersects with G(X).

S and R increase with increases in Q. At first, there is a single equilibrium, corresponding to the situation shown in the first graphs of Fig. 42.1a–e. After some time, the line becomes tangent to the curve, as in Graph b. With further increases in the slope, two points that "attract" system behavior emerge (Graph c). From left to right, these intersection points in Graph c are, respectively, stable, unstable, and stable.

Note that in each of the graphs of Fig. 42.1, X is the horizontal axis and F has the slope R/Q. As such, F will be equal to, or intersect, G at equilibrium points [per Eqs. (42.11) and (42.12)] of varying stability. If F is less than G, to the right of the

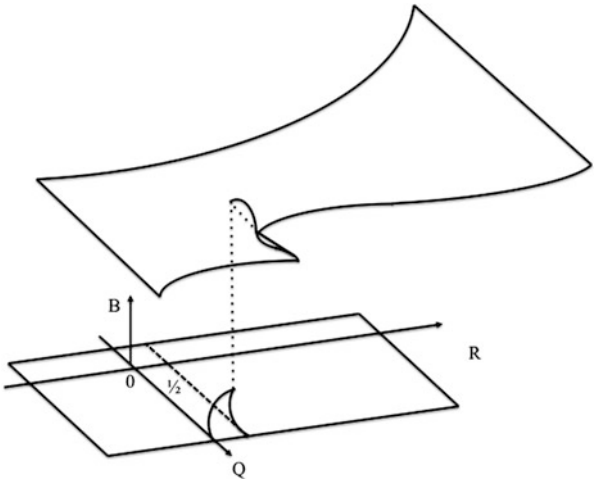

Fig. 42.2

intersection, the point is stable. Thus, between Graph a and Graph d of Fig. 42.1, the left intersection switches suddenly from stable to unstable, the only stable point becoming the right intersection. This sudden shift in stable points is the reason behind the sudden shift, or explosion, in the budworm population. Graph e in Fig. 42.1 shows the case in which only one point of intersection exists. This point is a stable attractor.

The cusp of the spruce budworm dynamics is shown schematically in Fig. 42.2 in the R–Q plane. The upper part of the surface corresponds to an outbreak level while the lower part corresponds to a subsistence level.

The modules to solve for the dynamics of the spruce budworm population are shown in Fig. 42.3. We drive changes in the model by setting X = TIME. The functions G(X) and F(X) are shown in Figs. 42.4 and 42.5, respectively.

For KS = 2.5, the corresponding values of X are 0.69, 2.0, and 7.32. These correspond to B = 0.173, 0.500, and 1.83, respectively [see Eq. (42.10)]. These B values represent the steady states of B. However, only two of these extrema are stable: the middle value represents an unstable extremum. All initial values of B within a given range will lead to the same, single steady state for B, one of the two. There are two such given ranges for initial Bs, given the way that the main model is set up—initial values of B greater and less than 1/2. Figure 42.6 shows, for a range of initial Bs close to 1/2 the approach to the two equilibria. Figures 42.7 and 42.8, in contrast, are for an initial value of B = 3.0.

Recognize that the choice of KS is a crucial one. So is the choice of Q. They determine whether there are one, two, or three extreme solutions for B. Run a series of sensitivity tests on KS and Q to explore their impact on the system's dynamics.

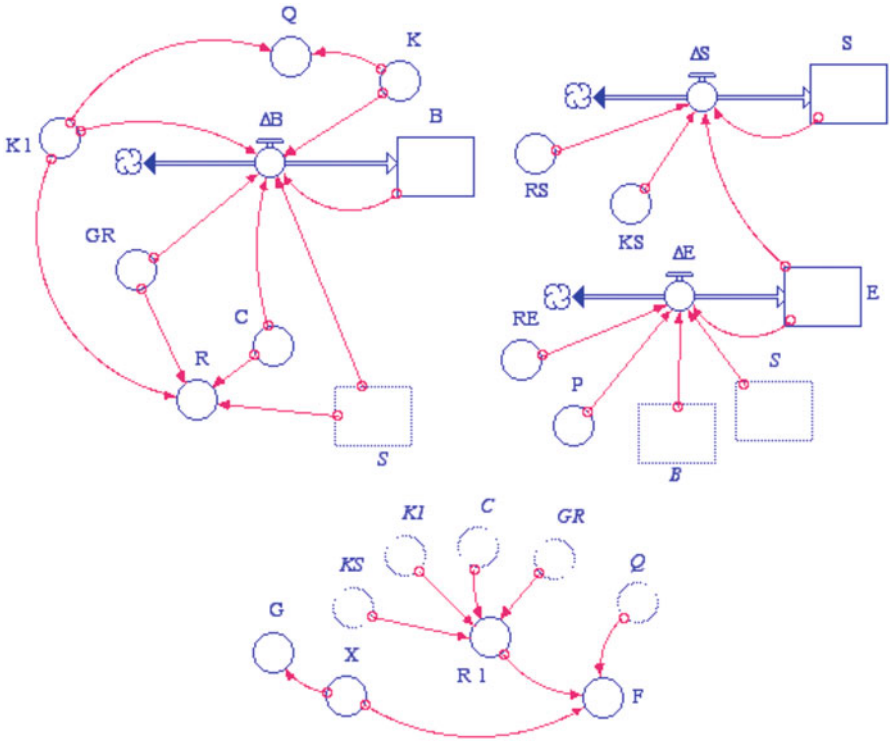

Fig. 42.3

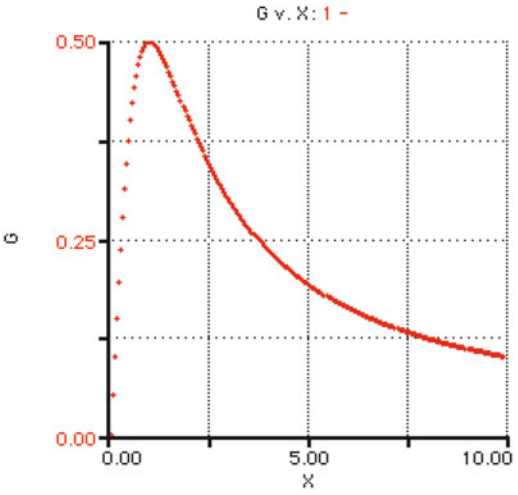

Fig. 42.4

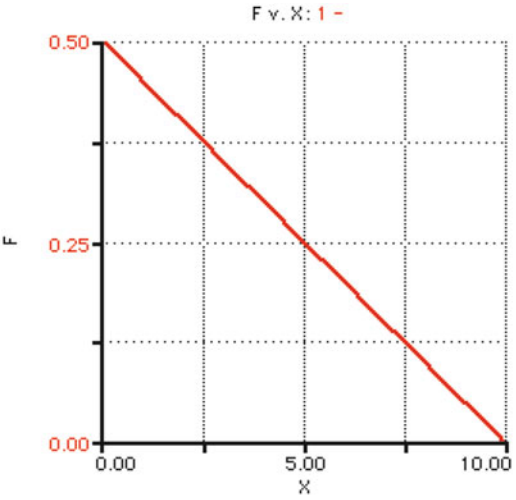

Fig. 42.5

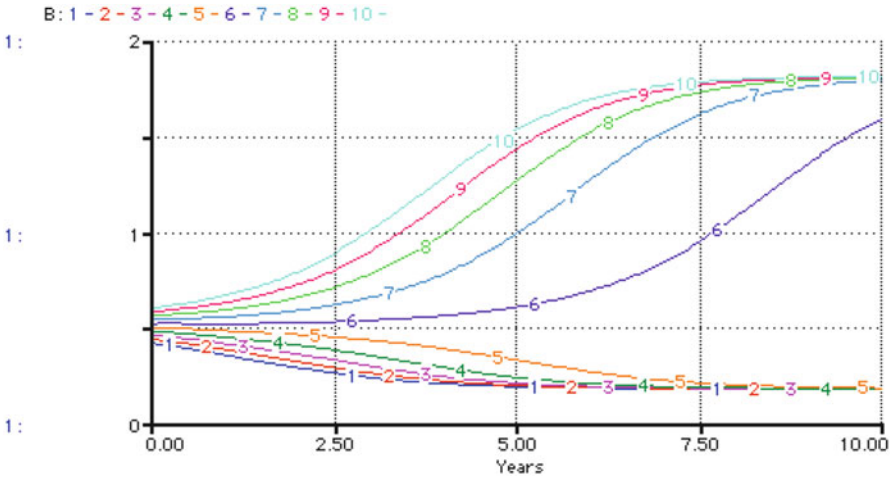

Fig. 42.6

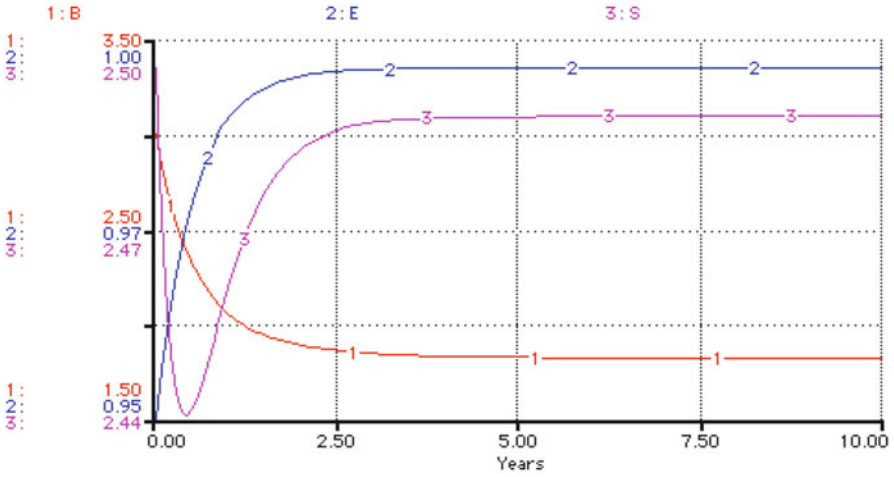

Fig. 42.7

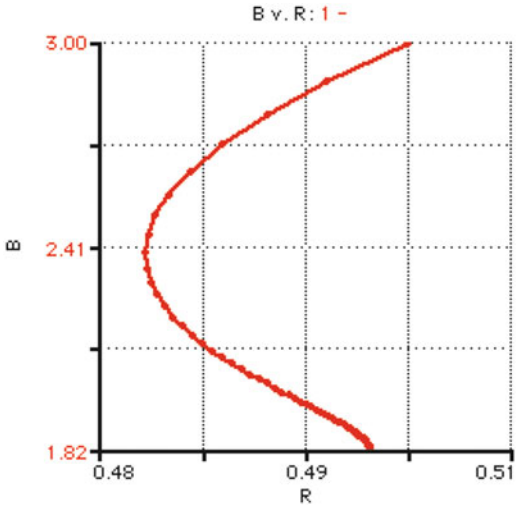

Fig. 42.8

42.2 Spruce Budworm Model Equations

$B(t) = B(t - dt) + (\Delta B) * dt$
INIT B = 3 {Spruce Budworm per Unit Area}
INFLOWS:
$\Delta B = GR*B*(1-B/K/S) - C*B\text{^}2/((K1*S)\text{^}2 + B\text{^}2)$ {Spruce Budworm per Unit Area per Time Period}

E(t) = E(t − dt) + (ΔE) ∗ dt
INIT E = .95 {Percentage of Foilage Cover}
INFLOWS:
ΔE = RE∗E∗(1−E) − P∗B∗E^2/S {Change in Percentage of Foilage Cover per Time Period}
S(t) = S(t − dt) + (ΔS) ∗ dt
INIT S = 2.5 {Habitat Density}
INFLOWS:
ΔS = RS∗S∗(1 − S/KS/E) {Habitat Density Change per Time Period}
C = 1
F = R_1∗(1−X/Q)
G = X/(1+X^2)
GR = 2 {Spruce Budworm per Unit Area per Spruce Budworm per Unit Area per Time Period}
K = 1
K1 = 0.1
KS = 2.5
P = 0.01
Q = K/K1
R = GR∗K1∗S/C
RE = 2 {Change in Percentage of Foilage Cover per Percentage Foilage Cover Time Period}
RS = 3 {Habitat Density Change per Habitat Density Change per Time Period}
R_1 = GR∗K1∗KS/C
X = TIME

Reference

1. Beltrami E (1987) Mathematics for dynamic modeling. Academic Press, Boston

Chapter 43
Game of Life

What I need is the dandelion in the spring. The bright yellow that means rebirth instead of destruction. The promise that life can go on, no matter how bad our losses. That it can be good again.

(Suzanne Collins, Mockingjay)

43.1 Game of Life Model

Two of the most striking features of nature are the ability of these systems to evolve and reproduce. For example, gradients in material concentrations and temperature in the primordial soup enabled formation of simple structures that were not only distinguishable from their environment but eventually also possessed the ability to change that environment. The emergence of catalytic RNA in a heterogeneous environment, in turn, not only led to change in the environment of these simple structures but, with chance, to the change of these structures themselves [1]. Increasingly complex molecular structures evolved, reproduced, and continuously transformed the environment in which they lived.

The evolution and reproduction of life forms was long interpreted as the result of some "vital force." It is only since the beginning of this century that we have gained a better understanding of evolution and reproduction from a physical perspective [2]. In 1966, John von Neumann created a simple model of a reproducing machine [3] that sparked the development of numerous efforts to boil down rules for replication to simple logical statements that take on the form of "mechanistic" rules. Collectively, these models are known as cellular automata. In general, a cellular automaton is any set of individuals with a finite number of characteristics

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_43,

that change over time based on previous characteristics and the characteristics of individuals with which it interacts. Typically, individuals are represented as cells in a grid. The features of each cell are determined by its features in the past and those of the neighboring cells.

The model of this chapter is set up for 100 cells arranged in a square. It is an extremely simple cellular automaton which has been developed by the British mathematician John H. Conway. Conway [4] called this model "Life," perhaps because the displays of the model looks remarkably like a Petri dish full of microscopic organisms, moving back and forth, merging with each other, replicating themselves or generating new forms.

The rules for the game of life are as follows:

1. Each cell on the grid can only take on two characteristics, for example, the colors black or white.
2. If, for a given cell, the number of neighbors is exactly two, the value of the square does not change at the next time step.
3. If the number of black neighbors is exactly three, the square will be black in the next time step.
4. If the number of black neighbors is neither two nor three, the square will be white at the next time step.

We have named the cells in our model with letters A, B, ..., J for rows and numbers 1, 2, ..., 10 for columns. The state of each cell is captured by a state variable that either takes on a value of one if it is black or zero if it is white. Changes in the state of a cell are captured by bi-flows and calculated on the basis of its own state and the value its neighbors take on. If a cell is black (i.e., its state variable takes on a value of one) and if exactly three of its neighbors are black (the sum of the state variables of all neighbors is three), then—according to the third rule above—there is no change in the state of the cell and the bi-flow is zero. If the cell is white but exactly three of its neighbors are black, then—according to the third rule above—the bi-flow takes on a value of one, changing the state of the respective cell from zero to one. Rules two and four can be accommodated analogously in the model. An example for the application of these rules to one of the cells in the model is given in the module of Fig. 43.1.

The parameter BIRTH # is the number of neighbor cells that must be black to "give birth" to a cell. According to our rules we set BIRTH # = 3. Similarly, NEIGHBOR MAX is the number of cells being black above which the cell dies from crowding, and NEIGHBOR MIN is the minimum number of neighboring cells below which the cell is no longer viable. NEIGHBOR MAX and NEIGHBOR MIN in our model are set to 3 and 2, respectively. The rule for the change in cell B4 is shown in Eq. (43.1).

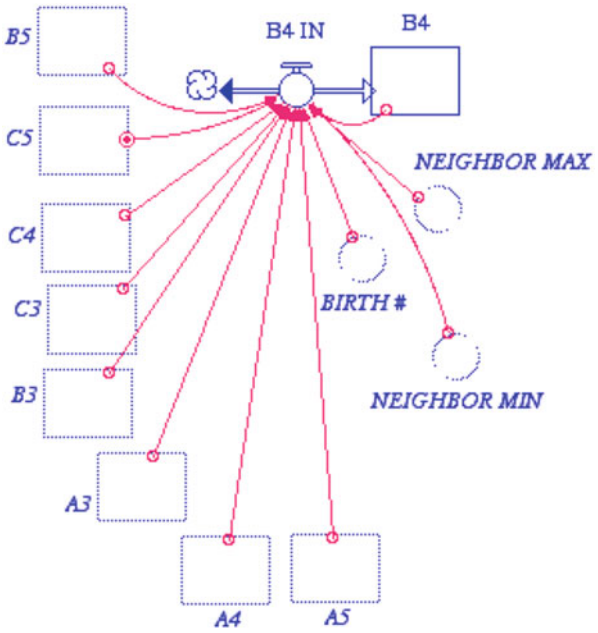

Fig. 43.1

$$
\begin{aligned}
\text{B4 IN} = \text{IF } & (\text{A3} + \text{A4} + \text{A5} + \text{B3} + \text{B5} + \text{C3} + \text{C4} + \text{C5}) = \text{BIRTH\#AND} \\
& \text{B4} = 0 \text{ THEN } 1 \\
& \text{ELSE IF } (\text{A3} + \text{A4} + \text{A5} + \text{B3} + \text{B5} + \text{C3} + \text{C4} + \text{C5}) < \text{NEIGHBOR MIN OR} \\
& (\text{A3} + \text{A4} + \text{A5} + \text{B3} + \text{B5} + \text{C3} + \text{C4} + \text{C5}) > \text{NEIGHBOR MAX AND} \\
& \text{B4} = 1 \text{ THEN} - 1 \\
& \text{ELSE } 0
\end{aligned}
\tag{43.1}
$$

Randomly initialize the value for each cell and observe the system's dynamics. You will find that some initial conditions lead to more interesting dynamics than others. Among the more interesting initial conditions for the model are the ones shown in the set-ups of Fig. 43.2, where a black cell is one with initial value of one, and a white cell has initial value of zero. Start the model with one of these initial conditions, and observe its spatio-temporal dynamics. Find your own constellations of initial conditions that lead to self-replicating structures. With this model you can construct "factories" which produce gliding swarms of black cells, or intricate structures of squares within squares.

We have animated the stocks in the STELLA model and placed ghosts of the stocks close together to set up a checkerboard-like pattern. The results of the model with initial conditions depicted in the upper left-hand panel of Fig. 43.2 yields the results shown in Fig. 43.3.

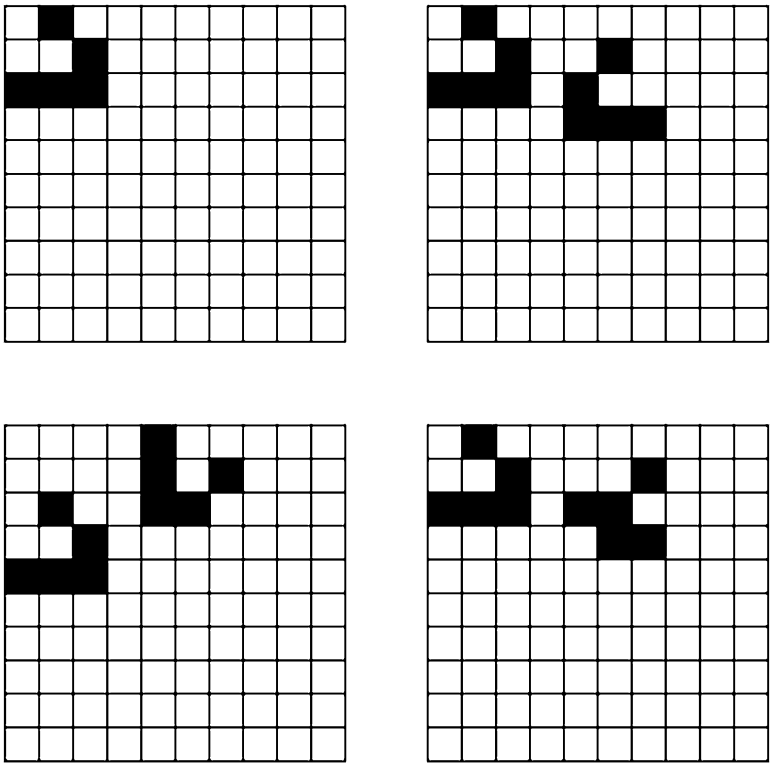

Fig. 43.2

The initial "glider" moves downward in the grid and to the right, repeatedly replicating itself until it reaches the lower right-hand corner of the model where it gets stuck. If you keep enlarging the number of cells, you can keep the slider going. Can you set up the decision rules such that the glider moves out of the cells, rather than gets stuck? The other initial conditions shown in Fig. 43.2 yield situations in which two gliders annihilate, and in which new, stable patterns emerge.

Cellular automata not only generate interesting-looking spatio-temporal patterns but have been proven to be quite informative in physics when modeling the behavior of fluids [5], and in landscape ecology where they have been used, for example, to model sediment transport on coastal landscapes [6] or the spread of forest fires [7].

In the model above, a "black" cell indicated the presence of an organism, "white" meant that there were no organisms in a particular cell. Can you set up the model to capture competition of two species with each other? When introducing a second type of organism, make only as few changes to the rules as possible to model that organism and its competitive behavior.

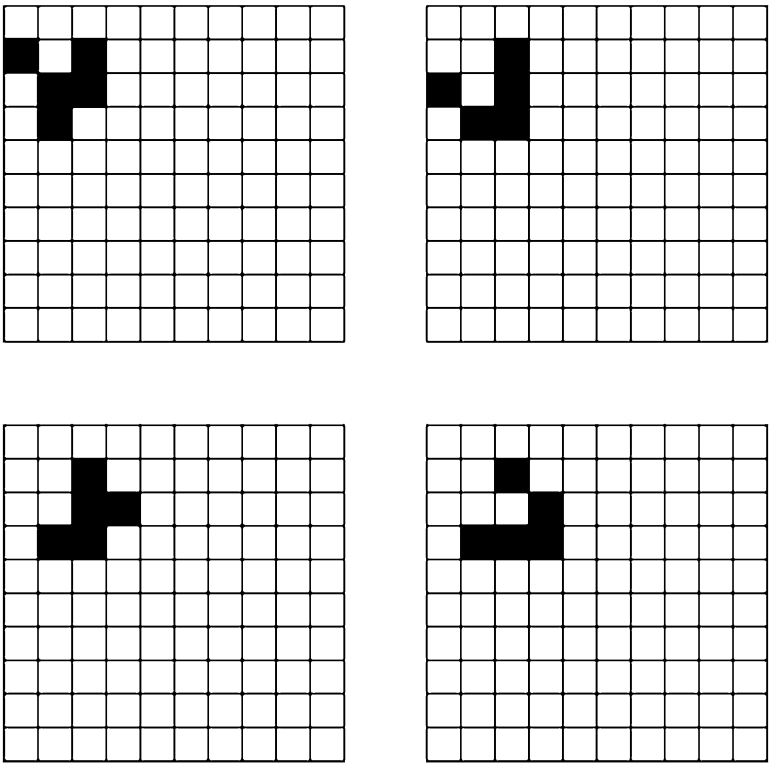

Fig. 43.3

43.2 Game of Life Model Equations

A1(t) = A1(t − dt) + (A1_IN) ∗ dt
INIT A1 = 0
INFLOWS:
A1_IN = IF (A2+B1+B2)=BIRTH_# and A1=0 THEN 1 ELSE IF (A2+B1+B2) <NEIGHBOR_MIN OR (A2+B1+B2)>NEIGHBOR_MAX AND A1=1 THEN −1 ELSE 0
A10(t) = A10(t − dt) + (A10_IN) ∗ dt
INIT A10 = 0
INFLOWS:
A10_IN = IF (A9+B10+B9)=BIRTH_# AND A10=0 THEN 1 ELSE IF (A9+B10 +B9)<NEIGHBOR_MIN OR (A9+B10+B9)>NEIGHBOR_MAX AND A10=1 THEN −1 ELSE 0
A2(t) = A2(t − dt) + (A2_IN) ∗ dt
INIT A2 = 1

INFLOWS:
A2_IN = IF (A1+A3+B1+B2+B3)=BIRTH_# AND A2=0 THEN 1 ELSE IF (A1+A3+B1+B2+B3)<NEIGHBOR_MIN OR (A1+A3+B1+B2+B3)> NEIGHBOR_MAX AND A2=1 THEN −1 ELSE 0
A3(t) = A3(t − dt) + (A3_IN) ∗ dt
INIT A3 = 0
INFLOWS:
A3_IN = IF (A2+A4+B2+B3+B4)=BIRTH_# AND A3=0 THEN 1 ELSE IF (A2+A4+B2+B3+B4)<NEIGHBOR_MIN OR (A2+A4+B2+B3+B4)> NEIGHBOR_MAX AND A3=1 THEN −1 ELSE 0
A4(t) = A4(t − dt) + (A4_IN) ∗ dt
INIT A4 = 0
INFLOWS:
A4_IN = IF (A3+A5+B3+B4+B5)=BIRTH_# AND A4=0 THEN 1 ELSE IF (A3+A5+B3+B4+B5)<NEIGHBOR_MIN OR (A3+A5+B3+B4+B5)> NEIGHBOR_MAX AND A4=1 THEN −1 ELSE 0
A5(t) = A5(t − dt) + (A5_IN) ∗ dt
INIT A5 = 0
INFLOWS:
A5_IN = IF (A6+B4+B5+B6+A4)=BIRTH_# AND A5=0 THEN 1 ELSE IF (A6+B4+B5+B6+A4)<NEIGHBOR_MIN OR (A6+B4+B5+B6+A4)> NEIGHBOR_MAX AND A5=1 THEN −1 ELSE 0
A6(t) = A6(t − dt) + (A6_IN) ∗ dt
INIT A6 = 0
INFLOWS:
A6_IN = IF (A7+B5+B6+B7+A5)=BIRTH_# AND A6=0 THEN 1 ELSE IF (A7+B5+B6+B7+A5)<NEIGHBOR_MIN OR (A7+B5+B6+B7+A5)> NEIGHBOR_MAX AND A6=1 THEN −1 ELSE 0
A7(t) = A7(t − dt) + (A7_IN) ∗ dt
INIT A7 = 0
INFLOWS:
A7_IN = IF (A6+A8+B6+B7+B8)=BIRTH_# AND A7=0 THEN 1 ELSE IF (A6+A8+B6+B7+B8)<NEIGHBOR_MIN OR (A6+A8+B6+B7+B8)> NEIGHBOR_MAX AND A7=1 THEN −1 ELSE 0
A8(t) = A8(t − dt) + (A8_IN) ∗ dt
INIT A8 = 0
INFLOWS:
A8_IN = IF (A7+A9+B7+B8+B9)=BIRTH_# AND A8=0 THEN 1 ELSE IF (A7+A9+B7+B8+B9)<NEIGHBOR_MIN OR (A7+A9+B7+B8+B9)> NEIGHBOR_MAX AND A8=1 THEN −1 ELSE 0
A9(t) = A9(t − dt) + (A9_IN) ∗ dt
INIT A9 = 0
INFLOWS:
A9_IN = IF (A10+A8+B10+B8+B9)=BIRTH_# AND A9=0 THEN 1 ELSE IF (A10+A8+B10+B8+B9)<NEIGHBOR_MIN OR (A10+A8+B10+B8+B9)> NEIGHBOR_MAX AND A9=1 THEN −1 ELSE 0

B1(t) = B1(t − dt) + (B1_IN) ∗ dt
INIT B1 = 0
INFLOWS:
B1_IN = IF (A1+A2+B2+C1+C2)=BIRTH_# AND B1=0 THEN 1 ELSE IF (A1+A2+B2+C1+C2)<NEIGHBOR_MIN OR (A1+A2+B2+C1+C2)> NEIGHBOR_MAX AND B1=1 THEN −1 ELSE 0
B10(t) = B10(t − dt) + (B10_IN) ∗ dt
INIT B10 = 0
INFLOWS:
B10_IN = IF (A10+A9+B9+C10+C9)=BIRTH_# AND B10=0 THEN 1 ELSE IF (A10+A9+B9+C10+C9)<NEIGHBOR_MIN OR (A10+A9+B9+C10+C9)> NEIGHBOR_MAX AND B10=1 THEN −1 ELSE 0
B2(t) = B2(t − dt) + (B2_IN) ∗ dt
INIT B2 = 0
INFLOWS:
B2_IN = IF (A1+A2+A3+B1+B3+C1+C2+C3)=BIRTH_# AND B2=0 THEN 1 ELSE IF (A1+A2+A3+B1+B3+C1+C2+C3)<NEIGHBOR_MIN OR (A1+A2+A3+B1+B3+C1+C2+C3)>NEIGHBOR_MAX AND B2=1 THEN −1 ELSE 0
B3(t) = B3(t − dt) + (B3_IN) ∗ dt
INIT B3 = 1
INFLOWS:
B3_IN = IF (A2+A3+A4+B2+B4+C2+C3+C4)=BIRTH_# AND B3=0 THEN 1 ELSE IF (A2+A3+A4+B2+B4+C2+C3+C4)<NEIGHBOR_MIN OR (A2+A3+A4+B2+B4+C2+C3+C4)>NEIGHBOR_MAX AND B3=1 THEN −1 ELSE 0
B4(t) = B4(t − dt) + (B4_IN) ∗ dt
INIT B4 = 0
INFLOWS:
B4_IN = IF (A3+A4+A5+B3+B5+C3+C4+C5)=BIRTH_# AND B4=0 THEN 1 ELSE IF (A3+A4+A5+B3+B5+C3+C4+C5)<NEIGHBOR_MIN OR (A3+A4+A5+B3+B5+C3+C4+C5)>NEIGHBOR_MAX AND B4=1 THEN −1 ELSE 0
B5(t) = B5(t − dt) + (B5_IN) ∗ dt
INIT B5 = 0
INFLOWS:
B5_IN = IF (A4+A5+A6+B4+B6+C4+C5+C6)=BIRTH_# AND B5=0 THEN 1 ELSE IF (A4+A5+A6+B4+B6+C4+C5+C6)<NEIGHBOR_MIN OR (A4+A5+A6+B4+B6+C4+C5+C6)>NEIGHBOR_MAX AND B5=1 THEN −1 ELSE 0
B6(t) = B6(t − dt) + (B6_IN) ∗ dt
INIT B6 = 0

INFLOWS:
B6_IN = IF (A5+A6+A7+B5+B7+C5+C6+C7)=BIRTH_# AND B6=0 THEN 1 ELSE IF (A5+A6+A7+B5+B7+C5+C6+C7)<NEIGHBOR_MIN OR (A5+A6+A7+B5+B7+C5+C6+C7)>NEIGHBOR_MAX AND B6=1 THEN −1 ELSE 0
B7(t) = B7(t − dt) + (B7_IN) ∗ dt
INIT B7 = 0
INFLOWS:
B7_IN = IF (A6+A7+A8+B6+B8+C6+C7+C8)=BIRTH_# AND B7=0 THEN 1 ELSE IF (A6+A7+A8+B6+B8+C6+C7+C8)<NEIGHBOR_MIN OR (A6+A7+A8+B6+B8+C6+C7+C8)>NEIGHBOR_MAX AND B7=1 THEN −1 ELSE 0
B8(t) = B8(t − dt) + (B8_IN) ∗ dt
INIT B8 = 0
INFLOWS:
B8_IN = IF (A7+A8+A9+B7+B9+C7+C8+C9)=BIRTH_# AND B8=0 THEN 1 ELSE IF (A7+A8+A9+B7+B9+C7+C8+C9)<NEIGHBOR_MIN OR (A7+A8+A9+B7+B9+C7+C8+C9)>NEIGHBOR_MAX AND B8=1 THEN −1 ELSE 0
B9(t) = B9(t − dt) + (B9_IN) ∗ dt
INIT B9 = 0
INFLOWS:
B9_IN = IF (A10+A8+A9+B10+B8+C10+C8+C9)=BIRTH_# AND B9=0 THEN 1 ELSE IF (A10+A8+A9+B10+B8+C10+C8+C9)<NEIGHBOR_MIN OR (A10+A8+A9+B10+B8+C10+C8+C9)>NEIGHBOR_MAX AND B9=1 THEN −1 ELSE 0
C1(t) = C1(t − dt) + (C1_IN) ∗ dt
INIT C1 = 1
INFLOWS:
C1_IN = IF (B1+B2+C2+D1+D2)=BIRTH_# AND C1=0 THEN 1 ELSE IF (B1+B2+C2+D1+D2)<NEIGHBOR_MIN OR (B1+B2+C2+D1+D2)>NEIGHBOR_MAX AND C1=1 THEN −1 ELSE 0
C10(t) = C10(t − dt) + (C10_IN) ∗ dt
INIT C10 = 0
INFLOWS:
C10_IN = IF (B10+B9+C9+D10+D9)=BIRTH_# AND C10=0 THEN 1 ELSE IF (B10+B9+C9+D10+D9)<NEIGHBOR_MIN OR (B10+B9+C9+D10+D9)>NEIGHBOR_MAX AND C10=1 THEN −1 ELSE 0
C2(t) = C2(t − dt) + (C2_IN) ∗ dt
INIT C2 = 1
INFLOWS:
C2_IN = IF (B1+B2+B3+C1+C3+D1+D2+D3)=BIRTH_# AND C1=0 THEN 1 ELSE IF (B1+B2+B3+C1+C3+D1+D2+D3)<NEIGHBOR_MIN OR (B1+B2+B3+C1+C3+D1+D2+D3)>NEIGHBOR_MAX AND C2=1 THEN −1 ELSE 0

C3(t) = C3(t − dt) + (C3_IN) ∗ dt
INIT C3 = 1
INFLOWS:
C3_IN = IF (B2+B3+B4+C2+C4+D2+D3+D4)=BIRTH_# AND C3=0 THEN 1 ELSE IF (B2+B3+B4+C2+C4+D2+D3+D4)<NEIGHBOR_MIN OR (B2+B3+B4+C2+C4+D2+D3+D4)>NEIGHBOR_MAX AND C3=1 THEN −1 ELSE 0
C4(t) = C4(t − dt) + (C4_IN) ∗ dt
INIT C4 = 0
INFLOWS:
C4_IN = IF (B3+B4+B5+C3+C5+D3+D4+D5)=BIRTH_# AND C4=0 THEN 1 ELSE IF (B3+B4+B5+C3+C5+D3+D4+D5)<NEIGHBOR_MIN OR (B3+B4+B5+C3+C5+D3+D4+D5)>NEIGHBOR_MAX AND C4=1 THEN −1 ELSE 0
C5(t) = C5(t − dt) + (C5_IN) ∗ dt
INIT C5 = 0
INFLOWS:
C5_IN = IF (B4+B5+B6+C4+C6+D4+D5+D6)=BIRTH_# AND C5=0 THEN 1 ELSE IF (B4+B5+B6+C4+C6+D4+D5+D6)<NEIGHBOR_MIN OR (B4+B5+B6+C4+C6+D4+D5+D6)>NEIGHBOR_MAX AND C5=1 THEN −1 ELSE 0
C6(t) = C6(t − dt) + (C6_IN) ∗ dt
INIT C6 = 0
INFLOWS:
C6_IN = IF (B5+B6+B7+C5+C7+D5+D6+D7)=BIRTH_# AND C6=0 THEN 1 ELSE IF (B5+B6+B7+C5+C7+D5+D6+D7)<NEIGHBOR_MIN OR (B5+B6+B7+C5+C7+D5+D6+D7)>NEIGHBOR_MAX AND C6=1 THEN −1 ELSE 0
C7(t) = C7(t − dt) + (C7_IN) ∗ dt
INIT C7 = 0
INFLOWS:
C7_IN = IF (B6+B7+B8+C6+C8+D6+D7+D8)=BIRTH_# AND C7=0 THEN 1 ELSE IF (B6+B7+B8+C6+C8+D6+D7+D8)<NEIGHBOR_MIN OR (B6+B7+B8+C6+C8+D6+D7+D8)>NEIGHBOR_MAX AND C7=1 THEN −1 ELSE 0
C8(t) = C8(t − dt) + (C8_IN) ∗ dt
INIT C8 = 0
INFLOWS:
C8_IN = IF (B7+B8+B9+C7+C9+D7+D8+D9)=BIRTH_# AND C8=0 THEN 1 ELSE IF (B7+B8+B9+C7+C9+D7+D8+D9)<NEIGHBOR_MIN OR (B7+B8+B9+C7+C9+D7+D8+D9)>NEIGHBOR_MAX AND C8=1 THEN −1 ELSE 0
C9(t) = C9(t − dt) + (C9_IN) ∗ dt
INIT C9 = 0

INFLOWS:
C9_IN = IF (B10+B8+B9+C10+C8+D10+D8+D9)=BIRTH_# AND C9=0 THEN 1 ELSE IF (B10+B8+B9+C10+C8+D10+D8+D9)<NEIGHBOR_MIN OR (B10+B8+B9+C10+C8+D10+D8+D9)>NEIGHBOR_MAX AND C9=1 THEN −1 ELSE 0
D1(t) = D1(t − dt) + (D1_IN) ∗ dt
INIT D1 = 0
INFLOWS:
D1_IN = IF (C1+C2+D2+E1+E2)=BIRTH_# AND D1=0 THEN 1 ELSE IF (C1+C2+D2+E1+E2)<NEIGHBOR_MIN OR (C1+C2+D2+E1+E2)> NEIGHBOR_MAX AND D1=1 THEN −1 ELSE 0
D10(t) = D10(t − dt) + (D10_IN) ∗ dt
INIT D10 = 0
INFLOWS:
D10_IN = IF (C10+C9+D9+E10+E9)=BIRTH_# AND D10=0 THEN 1 ELSE IF (C10+C9+D9+E10+E9)<NEIGHBOR_MIN OR (C10+C9+D9+E10+E9)> NEIGHBOR_MAX AND D10=1 THEN −1 ELSE 0
D2(t) = D2(t − dt) + (D2_IN) ∗ dt
INIT D2 = 0
INFLOWS:
D2_IN = IF (C1+C2+C3+D1+D3+E1+E2+E3)=BIRTH_# AND D2=0 THEN 1 ELSE IF (C1+C2+C3+D1+D3+E1+E2+E3)<NEIGHBOR_MIN OR (C1+C2+C3+D1+D3+E1+E2+E3)>NEIGHBOR_MAX AND D2=1 THEN −1 ELSE 0
D3(t) = D3(t − dt) + (D3_IN) ∗ dt
INIT D3 = 0
INFLOWS:
D3_IN = IF (C2+C3+C4+D2+D4+E2+E3+E4)=BIRTH_# AND D3=0 THEN 1 ELSE IF (C2+C3+C4+D2+D4+E2+E3+E4)<NEIGHBOR_MIN OR (C2+C3+C4+D2+D4+E2+E3+E4)>NEIGHBOR_MAX AND D3=1 THEN −1 ELSE 0
D4(t) = D4(t − dt) + (D4_IN) ∗ dt
INIT D4 = 0
INFLOWS:
D4_IN = IF (C3+C4+C5+D3+D5+E3+E4+E5)=BIRTH_# AND D4=0 THEN 1 ELSE IF (C3+C4+C5+D3+D5+E3+E4+E5)<NEIGHBOR_MIN OR (C3+C4+C5+D3+D5+E3+E4+E5)>NEIGHBOR_MAX AND D4=1 THEN −1 ELSE 0
D5(t) = D5(t − dt) + (D5_IN) ∗ dt
INIT D5 = 0
INFLOWS:
D5_IN = IF (C4+C5+C6+D4+D6+E4+E5+E6)=BIRTH_# AND D5=0 THEN 1 ELSE IF (C4+C5+C6+D4+D6+E4+E5+E6)<NEIGHBOR_MIN OR (C4+C5+C6+D4+D6+E4+E5+E6)>NEIGHBOR_MAX AND D5=1 THEN −1 ELSE 0

D6(t) = D6(t − dt) + (D6_IN) ∗ dt
INIT D6 = 0
INFLOWS:
D6_IN = IF (C5+C6+C7+D5+D7+E5+E6+E7)=BIRTH_# AND D6=0 THEN 1 ELSE IF (C5+C6+C7+D5+D7+E5+E6+E7)<NEIGHBOR_MIN OR (C5+C6+C7+D5+D7+E5+E6+E7)>NEIGHBOR_MAX AND D6=1 THEN −1 ELSE 0
D7(t) = D7(t − dt) + (D7_IN) ∗ dt
INIT D7 = 0
INFLOWS:
D7_IN = IF (C6+C7+C8+D6+D8+E6+E7+E8)=BIRTH_# AND D7=0 THEN 1 ELSE IF (C6+C7+C8+D6+D8+E6+E7+E8)<NEIGHBOR_MIN OR (C6+C7+C8+D6+D8+E6+E7+E8)>NEIGHBOR_MAX AND D7=1 THEN −1 ELSE 0
D8(t) = D8(t − dt) + (D8_IN) ∗ dt
INIT D8 = 0
INFLOWS:
D8_IN = IF (C7+C8+C9+D7+D9+E7+E8+E9)=BIRTH_# AND D8=0 THEN 1 ELSE IF (C7+C8+C9+D7+D9+E7+E8+E9)<NEIGHBOR_MIN OR (C7+C8+C9+D7+D9+E7+E8+E9)>NEIGHBOR_MAX AND D8=1 THEN −1 ELSE 0
D9(t) = D9(t − dt) + (D9_IN) ∗ dt
INIT D9 = 0
INFLOWS:
D9_IN = IF (C10+C8+C9+D10+D8+E10+E8+E9)=BIRTH_# AND D9=0 THEN 1 ELSE IF (C10+C8+C9+D10+D8+E10+E8+E9)<NEIGHBOR_MIN OR (C10+C8+C9+D10+D8+E10+E8+E9)>NEIGHBOR_MAX AND D9=1 THEN −1 ELSE 0
E1(t) = E1(t − dt) + (E1_IN) ∗ dt
INIT E1 = 0
INFLOWS:
E1_IN = IF (D1+D2+E2+F1+F2)=BIRTH_# AND E1=0 THEN 1 ELSE IF (D1+D2+E2+F1+F2)<NEIGHBOR_MIN OR (D1+D2+E2+F1+F2)>NEIGHBOR_MAX AND E1=1 THEN −1 ELSE 0
E10(t) = E10(t − dt) + (E10_IN) ∗ dt
INIT E10 = 0
INFLOWS:
E10_IN = IF (D10+D9+E9+F10+F9)=BIRTH_# AND E10=0 THEN 1 ELSE IF (D10+D9+E9+F10+F9)<NEIGHBOR_MIN OR (D10+D9+E9+F10+F9)>NEIGHBOR_MAX AND E10=1 THEN −1 ELSE 0
E2(t) = E2(t − dt) + (E2_IN) ∗ dt
INIT E2 = 0

INFLOWS:
E2_IN = IF (D1+D2+D3+E1+E3+F1+F2+F3)=BIRTH_# AND E2=0 THEN 1 ELSE IF (D1+D2+D3+E1+E3+F1+F2+F3)<NEIGHBOR_MIN OR (D1+D2+D3+E1+E3+F1+F2+F3)>NEIGHBOR_MAX AND E2=1 THEN −1 ELSE 0
E3(t) = E3(t − dt) + (E3_IN) ∗ dt
INIT E3 = 0
INFLOWS:
E3_IN = IF (D2+D3+D4+E2+E4+F2+F3+F4)=BIRTH_# AND E3=0 THEN 1 ELSE IF (D2+D3+D4+E2+E4+F2+F3+F4)<NEIGHBOR_MIN OR (D2+D3+D4+E2+E4+F2+F3+F4)>NEIGHBOR_MAX AND E3=1 THEN −1 ELSE 0
E4(t) = E4(t − dt) + (E4_IN) ∗ dt
INIT E4 = 0
INFLOWS:
E4_IN = IF (D3+D4+D5+E3+E5+F3+F4+F5)=BIRTH_# AND E4=0 THEN 1 ELSE IF (D3+D4+D5+E3+E5+F3+F4+F5)<NEIGHBOR_MIN OR (D3+D4+D5+E3+E5+F3+F4+F5)>NEIGHBOR_MAX AND E4=1 THEN −1 ELSE 0
E5(t) = E5(t − dt) + (E5_IN) ∗ dt
INIT E5 = 0
INFLOWS:
E5_IN = IF (D4+D5+D6+E4+E6+F4+F5+F6)=BIRTH_# AND E5=0 THEN 1 ELSE IF (D4+D5+D6+E4+E6+F4+F5+F6)<NEIGHBOR_MIN OR (D4+D5+D6+E4+E6+F4+F5+F6)>NEIGHBOR_MAX AND E5=1 THEN −1 ELSE 0
E6(t) = E6(t − dt) + (E6_IN) ∗ dt
INIT E6 = 0
INFLOWS:
E6_IN = IF (D5+D6+D7+E5+E7+F5+F6+F7)=BIRTH_# AND E6=0 THEN 1 ELSE IF (D5+D6+D7+E5+E7+F5+F6+F7)<NEIGHBOR_MIN OR (D5+D6+D7+E5+E7+F5+F6+F7)>NEIGHBOR_MAX AND E6=1 THEN −1 ELSE 0
E7(t) = E7(t − dt) + (E7_IN) ∗ dt
INIT E7 = 0
INFLOWS:
E7_IN = IF (D6+D7+D8+E6+E8+F6+F7+F8)=BIRTH_# AND E7=0 THEN 1 ELSE IF (D6+D7+D8+E6+E8+F6+F7+F8)<NEIGHBOR_MIN OR (D6+D7+D8+E6+E8+F6+F7+F8)>NEIGHBOR_MAX AND E7=1 THEN −1 ELSE 0
E8(t) = E8(t − dt) + (E8_IN) ∗ dt
INIT E8 = 0
INFLOWS:
E8_IN = IF (D7+D8+D9+E7+E9+F7+F8+F9)=BIRTH_# AND E8=0 THEN 1 ELSE IF (D7+D8+D9+E7+E9+F7+F8+F9)<NEIGHBOR_MIN OR (D7+D8+D9+E7+E9+F7+F8+F9)>NEIGHBOR_MAX AND E8=1 THEN −1 ELSE 0
E9(t) = E9(t − dt) + (E9_IN) ∗ dt
INIT E9 = 0

INFLOWS:
E9_IN = IF (D10+D8+D9+E10+E8+F10+F8+F9)=BIRTH_# AND E9=0 THEN 1 ELSE IF (D10+D8+D9+E10+E8+F10+F8+F9)<NEIGHBOR_MIN OR (D10+D8+D9+E10+E8+F10+F8+F9)>NEIGHBOR_MAX AND E9=1 THEN −1 ELSE 0
F1(t) = F1(t − dt) + (F1_IN) ∗ dt
INIT F1 = 0
INFLOWS:
F1_IN = IF (E1+E2+F2+G1+G2) = BIRTH_# AND F1=0 THEN 1 ELSE IF (E1+E2+F2+G1+G2) <NEIGHBOR_MIN OR (E1+E2+F2+G1+G2)> NEIGHBOR_MAX AND F1=1 THEN −1 ELSE 0
F10(t) = F10(t − dt) + (F10_IN) ∗ dt
INIT F10 = 0
INFLOWS:
F10_IN = IF (E10+E9+F9+G10+G9)=BIRTH_# AND F10=0 THEN 1 ELSE IF (E10+E9+F9+G10+G9)<NEIGHBOR_MIN OR (E10+E9+F9+G10+G9)> NEIGHBOR_MAX AND F10=1 THEN −1 ELSE 0
F2(t) = F2(t − dt) + (F2_IN) ∗ dt
INIT F2 = 0
INFLOWS:
F2_IN = IF (E1+E2+E3+F1+F3+G1+G2+G3)=BIRTH_# AND F2=0 THEN 1 ELSE IF (E1+E2+E3+F1+F3+G1+G2+G3)<NEIGHBOR_MIN OR (E1+E2+E3+F1+F3+G1+G2+G3)>NEIGHBOR_MAX AND F2=1 THEN −1 ELSE 0
F3(t) = F3(t − dt) + (F3_IN) ∗ dt
INIT F3 = 0
INFLOWS:
F3_IN = IF (E2+E3+E4+F2+F4+G2+G3+G4)=BIRTH_# AND F3=0 THEN 1 ELSE IF (E2+E3+E4+F2+F4+G2+G3+G4)<NEIGHBOR_MIN OR (E2+E3+E4+F2+F4+G2+G3+G4)>NEIGHBOR_MAX AND F3=1 THEN −1 ELSE 0
F4(t) = F4(t − dt) + (F4_IN) ∗ dt
INIT F4 = 0
INFLOWS:
F4_IN = IF (E3+E4+E5+F3+F5+G3+G4+G5)=BIRTH_# AND F4=0 THEN 1 ELSE IF (E3+E4+E5+F3+F5+G3+G4+G5)<NEIGHBOR_MIN OR (E3+E4+E5+F3+F5+G3+G4+G5)>NEIGHBOR_MAX AND F4=1 THEN −1 ELSE 0
F5(t) = F5(t − dt) + (F5_IN) ∗ dt
INIT F5 = 0
INFLOWS:
F5_IN = IF (E4+E5+E6+F4+F6+G4+G5+G6)=BIRTH_# AND F5=0 THEN 1 ELSE IF (E4+E5+E6+F4+F6+G4+G5+G6)<NEIGHBOR_MIN OR (E4+E5+E6+F4+F6+G4+G5+G6)>NEIGHBOR_MAX AND F5=1 THEN −1 ELSE 0
F6(t) = F6(t − dt) + (F6_IN) ∗ dt
INIT F6 = 0

INFLOWS:
F6_IN = IF (E5+E6+E7+F5+F7+G5+G6+G7)=BIRTH_# AND F6=0 THEN 1 ELSE IF (E5+E6+E7+F5+F7+G5+G6+G7)<NEIGHBOR_MIN OR (E5+E6+E7+F5+F7+G5+G6+G7)>NEIGHBOR_MAX AND F6=1 THEN −1 ELSE 0
F7(t) = F7(t − dt) + (F7_IN) ∗ dt
INIT F7 = 0
INFLOWS:
F7_IN = IF (E6+E7+E8+F6+F8+G6+G7+G8)=BIRTH_# AND F7=0 THEN 1 ELSE IF (E6+E7+E8+F6+F8+G6+G7+G8)<NEIGHBOR_MIN OR (E6+E7+E8+F6+F8+G6+G7+G8)>NEIGHBOR_MAX AND F7=1 THEN −1 ELSE 0
F8(t) = F8(t − dt) + (F8_IN) ∗ dt
INIT F8 = 0
INFLOWS:
F8_IN = IF (E7+E8+E9+F7+F9+G7+G8+G9)=BIRTH_# AND F8=0 THEN 1 ELSE IF (E7+E8+E9+F7+F9+G7+G8+G9)<NEIGHBOR_MIN OR (E7+E8+E9+F7+F9+G7+G8+G9)>NEIGHBOR_MAX AND F8=1 THEN −1 ELSE 0
F9(t) = F9(t − dt) + (F9_IN) ∗ dt
INIT F9 = 0
INFLOWS:
F9_IN = IF (E10+E8+E9+F10+F8+G10+G8+G9)=BIRTH_# AND F9=0 THEN 1 ELSE IF (E10+E8+E9+F10+F8+G10+G8+G9)<NEIGHBOR_MIN OR (E10+E8+E9+F10+F8+G10+G8+G9)>NEIGHBOR_MAX AND F9=1 THEN −1 ELSE 0
G1(t) = G1(t − dt) + (G1_IN) ∗ dt
INIT G1 = 0
INFLOWS:
G1_IN = IF (F1+F2+G2+H1+H2)=BIRTH_# AND G1=0 THEN 1 ELSE IF (F1+F2+G2+H1+H2)<NEIGHBOR_MIN OR (F1+F2+G2+H1+H2)>NEIGHBOR_MAX AND G1=1 THEN −1 ELSE 0
G10(t) = G10(t − dt) + (G10_IN) ∗ dt
INIT G10 = 0
INFLOWS:
G10_IN = IF (F10+F9+G9+H10+H9)=BIRTH_# AND G10=0 THEN 1 ELSE IF (F10+F9+G9+H10+H9)<NEIGHBOR_MIN OR (F10+F9+G9+H10+H9)>NEIGHBOR_MAX AND G10=1 THEN −1 ELSE 0
G2(t) = G2(t − dt) + (G2_IN) ∗ dt
INIT G2 = 0
INFLOWS:
G2_IN = IF (F1+F2+F3+G1+G3+H1+H2+H3)=BIRTH_# AND G2=0 THEN 1 ELSE IF (F1+F2+F3+G1+G3+H1+H2+H3)<NEIGHBOR_MIN OR (F1+F2+F3+G1+G3+H1+H2+H3)>NEIGHBOR_MAX AND G2=1 THEN −1 ELSE 0
G3(t) = G3(t − dt) + (G3_IN) ∗ dt
INIT G3 = 0

INFLOWS:
G3_IN = IF (F2+F3+F4+G2+G4+H2+H3+H4)=BIRTH_# AND G3=0 THEN 1 ELSE IF (F2+F3+F4+G2+G4+H2+H3+H4)<NEIGHBOR_MIN OR (F2+F3+F4+G2+G4+H2+H3+H4)>NEIGHBOR_MAX AND G3=1 THEN −1 ELSE 0
G4(t) = G4(t − dt) + (G4_IN) ∗ dt
INIT G4 = 0
INFLOWS:
G4_IN = IF (F3+F4+F5+G3+G5+H3+H4+H5)=BIRTH_# AND G4=0 THEN 1 ELSE IF (F3+F4+F5+G3+G5+H3+H4+H5)<NEIGHBOR_MIN OR (F3+F4+F5+G3+G5+H3+H4+H5)>NEIGHBOR_MAX AND G4=1 THEN −1 ELSE 0
G5(t) = G5(t − dt) + (G5_IN) ∗ dt
INIT G5 = 0
INFLOWS:
G5_IN = IF (F4+F5+F6+G4+G6+H4+H5+H6)=BIRTH_# AND G5=0 THEN 1 ELSE IF (F4+F5+F6+G4+G6+H4+H5+H6)<NEIGHBOR_MIN OR (F4+F5+F6+G4+G6+H4+H5+H6)>NEIGHBOR_MAX AND G5=1 THEN −1 ELSE 0
G6(t) = G6(t − dt) + (G6_IN) ∗ dt
INIT G6 = 0
INFLOWS:
G6_IN = IF (F5+F6+F7+G5+G7+H5+H6+H7)=BIRTH_# AND G6=0 THEN 1 ELSE IF (F5+F6+F7+G5+G7+H5+H6+H7)<NEIGHBOR_MIN OR (F5+F6+F7+G5+G7+H5+H6+H7)>NEIGHBOR_MAX AND G6=1 THEN −1 ELSE 0
G7(t) = G7(t − dt) + (G7_IN) ∗ dt
INIT G7 = 0
INFLOWS:
G7_IN = IF (F6+F7+F8+G6+G8+H6+H7+H8)=BIRTH_# AND G7=0 THEN 1 ELSE IF (F6+F7+F8+G6+G8+H6+H7+H8)<NEIGHBOR_MIN OR (F6+F7+F8+G6+G8+H6+H7+H8)>NEIGHBOR_MAX AND G7=1 THEN −1 ELSE 0
G8(t) = G8(t − dt) + (G8_IN) ∗ dt
INIT G8 = 0
INFLOWS:
G8_IN = IF (F7+F8+F9+G7+G9+H7+H8+H9)=BIRTH_# AND G8=0 THEN 1 ELSE IF (F7+F8+F9+G7+G9+H7+H8+H9)<NEIGHBOR_MIN OR (F7+F8+F9+G7+G9+H7+H8+H9)>NEIGHBOR_MAX AND G8=1 THEN −1 ELSE 0
G9(t) = G9(t − dt) + (G9_IN) ∗ dt
INIT G9 = 0
INFLOWS:
G9_IN = IF (F10+F8+F9+G10+G8+H10+H8+H9)=BIRTH_# AND G9=0 THEN 1 ELSE IF (F10+F8+F9+G10+G8+H10+H8+H9)<NEIGHBOR_MIN OR (F10+F8+F9+G10+G8+H10+H8+H9)>NEIGHBOR_MAX AND G9=1 THEN −1 ELSE 0
H1(t) = H1(t − dt) + (H1_IN) ∗ dt
INIT H1 = 0

INFLOWS:
H1_IN = IF (G1+G2+H2+I1+I2)=BIRTH_# AND H1=0 THEN 1 ELSE IF (G1+G2+H2+I1+I2)<NEIGHBOR_MIN OR (G1+G2+H2+I1+I2)>NEIGHBOR_MAX AND H1=1 THEN −1 ELSE 0
H10(t) = H10(t − dt) + (H10_IN) ∗ dt
INIT H10 = 0
INFLOWS:
H10_IN = IF (G10+G9+H9+I10+I9)=BIRTH_# AND H10=0 THEN 1 ELSE IF (G10+G9+H9+I10+I9)<NEIGHBOR_MIN OR (G10+G9+H9+I10+I9)>NEIGHBOR_MAX OR H10=1 THEN −1 ELSE 0
H2(t) = H2(t − dt) + (H2_IN) ∗ dt
INIT H2 = 0
INFLOWS:
H2_IN = IF (G1+G2+G3+H1+H3+I1+I2+I3)=BIRTH_# AND H2=0 THEN 1 ELSE IF (G1+G2+G3+H1+H3+I1+I2+I3)<NEIGHBOR_MIN OR (G1+G2+G3+H1+H3+I1+I2+I3)>NEIGHBOR_MAX AND H2=1 THEN −1 ELSE 0
H3(t) = H3(t − dt) + (H3_IN) ∗ dt
INIT H3 = 0
INFLOWS:
H3_IN = IF (G2+G3+G4+H2+H4+I2+I3+I4)=BIRTH_# AND H3=0 THEN 1 ELSE IF (G2+G3+G4+H2+H4+I2+I3+I4)<NEIGHBOR_MIN OR (G2+G3+G4+H2+H4+I2+I3+I4)>NEIGHBOR_MAX AND H3=1 THEN −1 ELSE 0
H4(t) = H4(t − dt) + (H4_IN) ∗ dt
INIT H4 = 0
INFLOWS:
H4_IN = IF (G3+G4+G5+H3+H5+I3+I4+I5)=BIRTH_# AND H4=0 THEN 1 ELSE IF (G3+G4+G5+H3+H5+I3+I4+I5)<NEIGHBOR_MIN OR (G3+G4+G5+H3+H5+I3+I4+I5)>NEIGHBOR_MAX AND H4=1 THEN −1 ELSE 0
H5(t) = H5(t − dt) + (H5_IN) ∗ dt
INIT H5 = 0
INFLOWS:
H5_IN = IF (G4+G5+G6+H4+H6+I4+I5+I6)=BIRTH_# AND H5=0 THEN 1 ELSE IF (G4+G5+G6+H4+H6+I4+I5+I6)<NEIGHBOR_MIN OR (G4+G5+G6+H4+H6+I4+I5+I6)>NEIGHBOR_MAX AND H5=1 THEN −1 ELSE 0
H6(t) = H6(t − dt) + (H6_IN) ∗ dt
INIT H6 = 0
INFLOWS:
H6_IN = IF (G5+G6+G7+H5+H7+I5+I6+I7)=BIRTH_# AND H6=0 THEN 1 ELSE IF (G5+G6+G7+H5+H7+I5+I6+I7)<NEIGHBOR_MIN OR (G5+G6+G7+H5+H7+I5+I6+I7)>NEIGHBOR_MAX AND H6=1 THEN −1 ELSE 0
H7(t) = H7(t − dt) + (H7_IN) ∗ dt
INIT H7 = 0

INFLOWS:
H7_IN = IF (G6+G7+G8+H6+H8+I6+I7+I8)=BIRTH_# AND H7=0 THEN 1 ELSE IF (G6+G7+G8+H6+H8+I6+I7+I8)<NEIGHBOR_MIN OR (G6+G7+G8+H6+H8+I6+I7+I8)>NEIGHBOR_MAX AND H7=1 THEN −1 ELSE 0
H8(t) = H8(t − dt) + (H8_IN) ∗ dt
INIT H8 = 0
INFLOWS:
H8_IN = IF (G7+G8+G9+H7+H9+I7+I8+I9)=BIRTH_# AND H8=0 THEN 1 ELSE IF (G7+G8+G9+H7+H9+I7+I8+I9)<NEIGHBOR_MIN OR (G7+G8+G9+H7+H9+I7+I8+I9)>NEIGHBOR_MAX AND H8=1 THEN −1 ELSE 0
H9(t) = H9(t − dt) + (H9_IN) ∗ dt
INIT H9 = 0
INFLOWS:
H9_IN = IF (G10+G8+G9+H10+H8+I10+I8+I9)=BIRTH_# AND H9=0 THEN 1 ELSE IF (G10+G8+G9+H10+H8+I10+I8+I9)<NEIGHBOR_MIN OR (G10+G8+G9+H10+H8+I10+I8+I9)>NEIGHBOR_MAX AND H9=1 THEN −1 ELSE 0
I1(t) = I1(t − dt) + (I1_IN) ∗ dt
INIT I1 = 0
INFLOWS:
I1_IN = IF (H1+H2+I2+J1+J2)=BIRTH_# AND I1=0 THEN 1 ELSE IF (H1+H2+I2+J1+J2)<NEIGHBOR_MIN OR (H1+H2+I2+J1+J2)>NEIGHBOR_MAX AND I1=1 THEN −1 ELSE 0
I10(t) = I10(t − dt) + (I10_IN) ∗ dt
INIT I10 = 0
INFLOWS:
I10_IN = IF (H10+H9+I9+J10+J9)=BIRTH_# AND I10=0 THEN 1 ELSE IF (H10+H9+I9+J10+J9)<NEIGHBOR_MIN OR (H10+H9+I9+J10+J9)>NEIGHBOR_MAX AND I10=1 THEN −1 ELSE 0
I2(t) = I2(t − dt) + (I2_IN) ∗ dt
INIT I2 = 0
INFLOWS:
I2_IN = IF (H1+H2+H3+I1+I3+J1+J2+J3)=BIRTH_# AND I2=0 THEN 1 ELSE IF (H1+H2+H3+I1+I3+J1+J2+J3)<NEIGHBOR_MIN OR (H1+H2+H3+I1+I3+J1+J2+J3)>NEIGHBOR_MAX AND I2=1 THEN −1 ELSE 0
I3(t) = I3(t − dt) + (I3_IN) ∗ dt
INIT I3 = 0
INFLOWS:
I3_IN = IF (H2+H3+H4+I2+I4+J2+J3+J4)=BIRTH_# AND I3=0 THEN 1 ELSE IF (H2+H3+H4+I2+I4+J2+J3+J4)<NEIGHBOR_MIN OR (H2+H3+H4+I2+I4+J2+J3+J4)>NEIGHBOR_MAX AND I3=1 THEN −1 ELSE 0
I4(t) = I4(t − dt) + (I4_IN) ∗ dt
INIT I4 = 0

INFLOWS:
I4_IN = IF (H3+H4+H5+I3+I5+J3+J4+J5)=BIRTH_# AND I4=0 THEN 1 ELSE IF (H3+H4+H5+I3+I5+J3+J4+J5)<NEIGHBOR_MIN OR (H3+H4+H5+I3+I5+J3+J4+J5)>NEIGHBOR_MAX AND I4=1 THEN −1 ELSE 0
I5(t) = I5(t − dt) + (I5_IN) ∗ dt
INIT I5 = 0
INFLOWS:
I5_IN = IF (H4+H5+H6+I4+I6+J4+J5+J6)=BIRTH_# AND I5=0 THEN 1 ELSE IF (H4+H5+H6+I4+I6+J4+J5+J6)<NEIGHBOR_MIN OR (H4+H5+H6+I4+I6+J4+J5+J6)>NEIGHBOR_MAX AND I5=1 THEN −1 ELSE 0
I6(t) = I6(t − dt) + (I6_IN) ∗ dt
INIT I6 = 0
INFLOWS:
I6_IN = IF (H5+H6+H7+I5+I7+J5+J6+J7)=BIRTH_# AND I6=0 THEN 1 ELSE IF (H5+H6+H7+I5+I7+J5+J6+J7)<NEIGHBOR_MIN OR (H5+H6+H7+I5+I7+J5+J6+J7)>NEIGHBOR_MAX AND I6=1 THEN −1 ELSE 0
I7(t) = I7(t − dt) + (I7_IN) ∗ dt
INIT I7 = 0
INFLOWS:
I7_IN = IF (H6+H7+H8+I6+I8+J6+J7+J8)=BIRTH_# AND I7=0 THEN 1 ELSE IF (H6+H7+H8+I6+I8+J6+J7+J8)<NEIGHBOR_MIN OR (H6+H7+H8+I6+I8+J6+J7+J8)>NEIGHBOR_MAX AND I7=1 THEN −1 ELSE 0
I8(t) = I8(t − dt) + (I8_IN) ∗ dt
INIT I8 = 0
INFLOWS:
I8_IN = IF (H7+H8+H9+I7+I9+J7+J8+J9)=BIRTH_# AND I8=0 THEN 1 ELSE IF (H7+H8+H9+I7+I9+J7+J8+J9)<NEIGHBOR_MIN OR (H7+H8+H9+I7+I9+J7+J8+J9)>NEIGHBOR_MAX AND I8=1 THEN −1 ELSE 0
I9(t) = I9(t − dt) + (I9_IN) ∗ dt
INIT I9 = 0
INFLOWS:
I9_IN = IF (H10+H8+I10+I8+H9+J10+J8+J9)=BIRTH_# AND I9=0 THEN 1 ELSE IF (H10+H8+I10+I8+H9+J10+J8+J9)<NEIGHBOR_MIN OR (H10+H8+I10+I8+H9+J10+J8+J9)>NEIGHBOR_MAX AND I9=1 THEN −1 ELSE 0
J1(t) = J1(t − dt) + (J1_IN) ∗ dt
INIT J1 = 0
INFLOWS:
J1_IN = IF (I1+I2+J2)=BIRTH_# AND J1=0 THEN 1 ELSE IF (I1+I2+J2)<NEIGHBOR_MIN OR (I1+I2+J2)>NEIGHBOR_MAX AND J1=1 THEN −1 ELSE 0
J10(t) = J10(t − dt) + (J10_IN) ∗ dt
INIT J10 = 0

INFLOWS:
J10_IN = IF (I10+I9+J9)=BIRTH_# AND J10=0 THEN 1 ELSE IF (I10+I9+J9)< NEIGHBOR_MIN OR (I10+I9+J9)>NEIGHBOR_MAX AND J10=1 THEN −1 ELSE 0
J2(t) = J2(t − dt) + (J2_IN) ∗ dt
INIT J2 = 0
INFLOWS:
J2_IN = IF (I1+I2+I3+J1+J3)=BIRTH_# AND J2=0 THEN 1 ELSE IF (I1+I2+I3 +J1+J3)<NEIGHBOR_MIN OR (I1+I2+I3+J1+J3)>NEIGHBOR_MAX AND J2=1 THEN −1 ELSE 0
J3(t) = J3(t − dt) + (J3_IN) ∗ dt
INIT J3 = 0
INFLOWS:
J3_IN = IF (I2+I3+I4+J2+J4)=BIRTH_# AND J3=0 THEN 1 ELSE IF (I2+I3+I4 +J2+J4)<NEIGHBOR_MIN OR (I2+I3+I4+J2+J4)>NEIGHBOR_MAX AND J3=1 THEN −1 ELSE 0
J4(t) = J4(t − dt) + (J4_IN) ∗ dt
INIT J4 = 0
INFLOWS:
J4_IN = IF (I3+I4+I5+J3+J5)=BIRTH_# AND J4=0 THEN 1 ELSE IF (I3+I4+I5 +J3+J5)<NEIGHBOR_MIN OR (I3+I4+I5+J3+J5)>NEIGHBOR_MAX AND J4=1 THEN −1 ELSE 0
J5(t) = J5(t − dt) + (J5_IN) ∗ dt
INIT J5 = 0
INFLOWS:
J5_IN = IF (I4+I5+I6+J4+J6)=BIRTH_# AND J5=0 THEN 1 ELSE IF (I4+I5+I6 +J4+J6)<NEIGHBOR_MIN OR (I4+I5+I6+J4+J6)>NEIGHBOR_MAX AND J5=1 THEN −1 ELSE 0
J6(t) = J6(t − dt) + (J6_IN) ∗ dt
INIT J6 = 0
INFLOWS:
J6_IN = IF (I5+I6+I7+J5+J7)=BIRTH_# AND J6=0 THEN 1 ELSE IF (I5+I6+I7 +J5+J7)<NEIGHBOR_MIN OR (I5+I6+I7+J5+J7)>NEIGHBOR_MAX AND J6=1 THEN −1 ELSE 0
J7(t) = J7(t − dt) + (J7_IN) ∗ dt
INIT J7 = 0
INFLOWS:
J7_IN = IF (I6+I7+I8+J6+J8)=BIRTH_# AND J7=0 THEN 1 ELSE IF (I6+I7+I8 +J6+J8)<NEIGHBOR_MIN OR (I6+I7+I8+J6+J8)>NEIGHBOR_MAX AND J7=1 THEN −1 ELSE 0
J8(t) = J8(t − dt) + (J8_IN) ∗ dt
INIT J8 = 0

INFLOWS:
J8_IN = IF (I7+I8+I9+J7+J9)=BIRTH_# AND J8=0 THEN 1 ELSE IF (I7+I8+I9 +J7+J9)<NEIGHBOR_MIN OR (I7+I8+I9+J7+J9)>NEIGHBOR_MAX AND J8=1 THEN −1 ELSE 0
J9(t) = J9(t − dt) + (J9_IN) ∗ dt
INIT J9 = 0
INFLOWS:
J9_IN = IF (I10+I8+I9+J10+J8)=BIRTH_# AND J9=0 THEN 1 ELSE IF (I10+I8 +I9+J10+J8) <NEIGHBOR_MIN OR (I10+I8+I9+J10+J8)> NEIGHBOR_MAX AND J9=1 THEN −1 ELSE 0
BIRTH_# = 3 {NUMBER OF NEIGHBOR CELLS REQUIRED TO GIVE BIRTH TO THE CELL}
EAT_LAG = 3
NEIGHBOR_MAX = 3 {Number of neighbor cells above which the cell dies from crowding}
NEIGHBOR_MIN = 2 {Minimum number of neighboring cells below which the cell is no longer viable}
REPROD_PROB = 1

References

1. Orgel LE (1994) The origin of life on the earth. Sci Am 271:77–83
2. Schrödinger E (1944) What is life. Cambridge University Press, Cambridge
3. Von Neumann J (1966) Theory of self-reproducing automata. University of Illinois Press, Chicago
4. Poundstone W (1985) The recursive universe. Contemporary Books, Chicago
5. Bak P, Chen K, Creutz M (1989) Self-organized criticality in the "game of life". Nature 342:780–782
6. Ruth M, Pieper F (1994) Modeling spatial dynamics of sea level rise in a coastal area. Syst Dyn Rev 10(4):375–389
7. Bak P, Chen K, Tang C (1990) A forest fire model and some thoughts on turbulence. Phys Lett 147(5–6):297–300

Chapter 44
Daisyworld

One ring to rule them all, one ring to bind them, One ring to bring them all, and in the darkness bind them.

(J.R.R. Tolkien, The Lord of the Rings, Part 1, Ballantine Books, NY 1965)

44.1 Daisyworld Model

During the late 1960s and early 1970s, James Lovelock, an independent inventor and scientist, and Lynn Margulis, a professor at Boston University, worked with the NASA Jet Propulsion Laboratory to develop a means to detect life on Mars. It was noted in the progress of this work that one striking property of the Earth is that its atmosphere is far from chemical equilibrium since the biota use it as a reservoir for nutrients and waste products. In other words, the atmosphere is, in the steady state, not derived of ordinary chemistry and physics. In fact, it is derived of life.

Conventional thought is that the Earth and the life upon it evolved separately by different mechanisms. The evolution of the planet's surface features and the atmosphere is a result of chemical and physical processes to which the biota must adapt to prevent extinction. In some instances, it is very clear that life has had a significant effect on the environment, yet these feedbacks are regarded as accidental. Lovelock has criticized this view for not adequately describing the sophisticated interaction between life and its environment and has developed his own theory known as the Gaia Hypothesis.

The Gaia Hypothesis states quite simply that "the Earth is homeostatic, with the biota actively seeking to keep the environment optimal for life" [1]. Lovelock suggests that life has the capacity to regulate the environment of the Earth in

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_44,

order to maintain its fitness for that life. Thus, according to the Gaia hypothesis, life and the environment evolve together as a single system. The species that leaves the most progeny tend to inherit a particular environment, and in turn, the environment that favors the most progeny is itself sustained.

There has been little success in finding the feedback mechanisms that regulate Gaia, but these mechanisms are likely to be subtle and complex. Scientists have questioned the validity of the Gaia hypothesis for this reason and others. One of the most important criticisms has been that the active regulation of the environment that is the central property of Gaia supposes natural selection for traits that will not be beneficial for thousands of years. But the ecosystem has no capacity for conscious forethought and planning, and it is clear that the biota would not evolve altruistic planetary regulation systems in the pursuit of local self-interest. In other words, if bacteria appear on the Earth's surface, how can these bacteria regulate the atmosphere to suit their own needs when they cannot possibly have a significant effect on the atmosphere's chemistry for thousands of years? Lovelock chose to challenge this criticism with a paradigm and a numerical model: Daisyworld. Since he first proposed the model, he also asserted that Daisyworld fully exhibits all of the characteristics of a Gaian world.

In our simplified view, Daisyworld is a right cylindrical planet in Earth's orbit around a sun of Sol's luminescence. This planet has a clear atmosphere and is populated by two life forms: light and dark daisies. Here, albedo is specified as a unitless measure of the reflectivity of a surface where 1 means perfectly reflective and 0 means perfectly absorptive. Light daisies have an albedo greater than the planet surface albedo and dark daisies have an albedo less than that of the planet's surface. If there is bare planet surface available, the daisies will grow and cover it, and change its albedo.

The model of Daisyworld is based on Lovelock's equations. The original equations and parameters are listed in *Tellus* [2]. We briefly present them here. There are only two stocks in this model, LIGHT DAISY AREA and DARK DAISY AREA. These stocks represent the fraction of planet surface covered by each type of daisy (Fig. 44.1).

Growth of the daisy populations is temperature dependent. For each daisy species, there is a temperature "window" where growth can occur. That window extends from 5 to 45 °C. Within this window, growth is described by the parabola defined with Eqs. (44.1) and (44.2).

$$\text{LIGHT DAISY GROWTH} = 1 - 0.003265(25 - \text{LIGHT DAISY TEMP})\text{^}2. \tag{44.1}$$

$$\text{DARK DAISY GROWTH} = 1 - 0.003265 * (25 - \text{DARK DAISY TEMP})\text{^}2 \tag{44.2}$$

These functions are maximal at a temperature of 25 °C. The actual daisy growth is then given by the differential equations

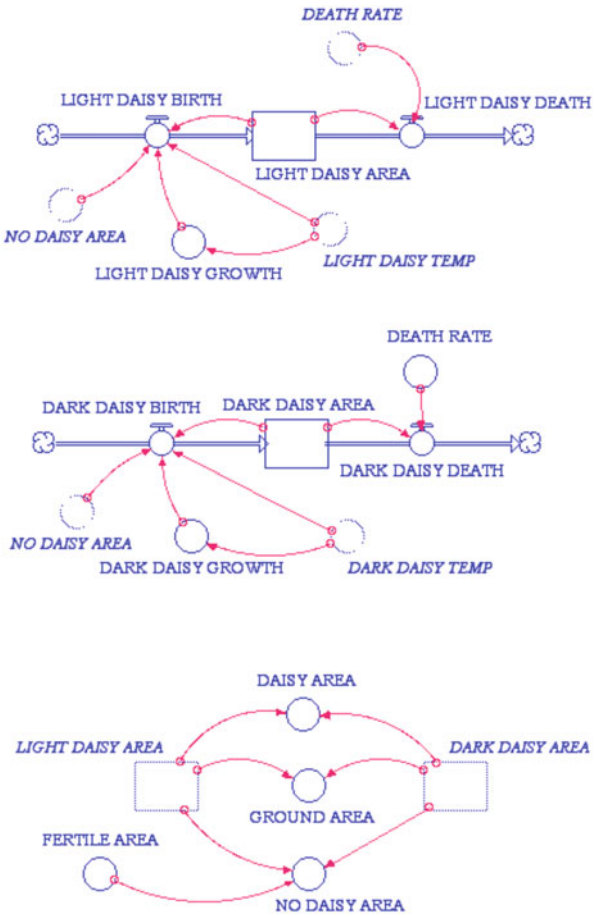

Fig. 44.1

$$\begin{aligned} \mathrm{d}(\text{LIGHT DAISY AREA})/\mathrm{dt} = {} & \text{LIGHT DAISY AREA} * \text{NO DAISY AREA} \\ & * \text{LIGHT DAISY GROWTH} \end{aligned} \tag{44.3}$$

$$\begin{aligned} \mathrm{d}(\text{DARK DAISY AREA})/\mathrm{dt} = {} & \text{DARK DAISY AREA} * \text{NO DAISY AREA} \\ & * \text{DARK DAISY GROWTH} \end{aligned} \tag{44.4}$$

where NO DAISY AREA is the fraction of fertile planet surface area not populated by daisies:

$$\begin{aligned} \text{NO DAISY AREA} = {} & \text{FERTILE AREA} - \text{LIGHT DAISY AREA} \\ & - \text{DARK DAISY AREA} \end{aligned} \tag{44.5}$$

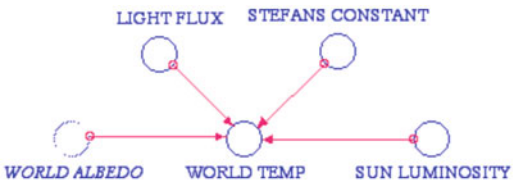

Fig. 44.2

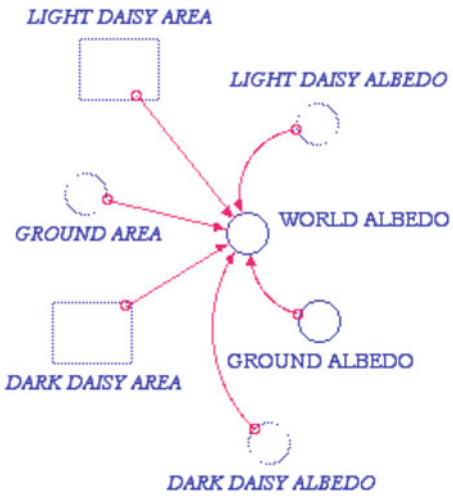

Fig. 44.3

Fertile Area is the exogenously given fraction of the planet surface that is habitable by daisies. Similarly, the area occupied by daisies is

$$\text{DAISY AREA} = \text{LIGHT DAISY AREA} + \text{DARK DAISY AREA} \tag{44.6}$$

and the area not occupied by daisies is

$$\text{GROUND AREA} = 1 - \text{DARK DAISY AREA} - \text{LIGHT DAISY AREA} \tag{44.7}$$

The daisies are also subject to death at the rate of 30 % of the population per unit time.

The most crucial aspect of the Daisyworld model lies in the calculation of the temperature of the world. The modules of Figs. 44.2 and 44.3 are devoted to this calculation. The first of these modules (Fig. 44.2) calculates world temperature. Physically, the planet surface temperature is determined by the amount of solar radiation incident upon the planet and the amount reflected by the planet. Using the Stefan-Boltzmann rule, WORLD TEMP is given by

$$\begin{aligned}\text{WORLD TEMP} = &\big((\text{SUN LUMINOSITY} * \text{LIGHT FLUX} \\ &* (1 - \text{WORLD ALBEDO})\big)/\text{STEFANS CONSTANT})\text{\^{}}0.25 - 273\end{aligned} \tag{44.8}$$

where LIGHT FLUX is equal to the solar flux of our sun, SUN LUMINOSITY is a dimensionless measure of the sun's luminosity, and STEFANS CONSTANT is the proportionality between solar radiation flux and temperature.

WORLD ALBEDO, in turn, is determined by

$$\begin{aligned}\text{WORLD ALBEDO} = &\text{LIGHT DAISY AREA} * \text{LIGHT DAISY ALBEDO} \\ &+ \text{DARK DAISY ALBEDO} * \text{DARK DAISY AREA} \\ &+ \text{GROUND AREA} * \text{GROUND ALBEDO}.\end{aligned} \tag{44.9}$$

WORLD ALBEDO is calculated in the module of Fig. 44.3. In our first scenario, the sun's luminosity is assumed to change over time:

$$\begin{aligned}\text{SUN LUMINOSITY} = &\text{IF TIME} < 150 \text{ THEN } 0.01 * \text{TIME} \\ &\text{ELSE } 1.5 - 0.01 * (\text{TIME} - 150)\end{aligned} \tag{44.10}$$

The key to Daisyworld's ability to regulate temperature is the different albedos of the light and dark daisies. Because dark daisies absorb more solar radiation, they are warmer at a given solar flux than light daisies. Thus, they are at a different point on their growth curve. Equations (44.8) and (44.9) show how Daisyworld's temperature is regulated by the daisies. But the WORLD TEMP is not what determines the growth of the daisy species. Rather, it is the local temperature of each daisy that regulates growth. The local temperature for the two daisy species is given by

$$\begin{aligned}\text{LIGHT DAISY TEMP} = &\text{LOCAL TEMP FUNC} * \big(\text{WORLD ALBEDO} \\ &- \text{LIGHT DAISY ALBEDO}\big) + \text{WORLD TEMP}\end{aligned} \tag{44.11}$$

$$\begin{aligned}\text{DARK DAISY TEMP} = &\text{LOCAL TEMP FUNC} * \big(\text{WORLD ALBEDO} \\ &- \text{DARK DAISY ALBEDO}\big) + \text{WORLD TEMP}\end{aligned} \tag{44.12}$$

where LOCAL TEMP FUNC is a constant that measures the degree of insulation between daisies (Fig. 44.4). A low LOCAL TEMP FUNC means that there is a great deal of heat conduction between daisy species and all local temperatures equal the mean temperature while a high LOCAL TEMP FUNC (on the order of 10^2) indicates insulation between high and low temperature regions on the surface. For this simulation, LOCAL TEMP FUNC has been chosen to be 20, which is between the two extremes, but closer to a "conductive" planet surface.

We assume that the sun's luminosity changes over time according to Eq. (44.10) in order to investigate how a Gaian system reacts to perturbations. Since the

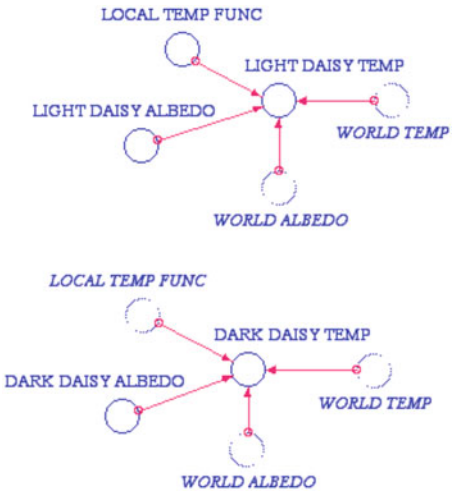

Fig. 44.4

environment of Daisyworld is solely specified by temperature, the best way to test the Gaia Hypothesis is to affect Daisyworld in such a way that temperature is the independent variable. Note, however, that most of the graphs presented here have time as the independent variable. This is somewhat misleading since time is not significant to Daisyworld; instead the luminosity of the sun is the independent variable used in this investigation. But SUN LUMINOSITY is varied linearly with respect to time, so the time axes in the graphs can easily be translated to a SUN LUMINOSITY axis.

First, let us consider what is expected of a Daisyworld unaffected by biology. In this case, the albedo of the planet surface is constant and equal to 0.5. As the solar luminosity increases, the temperature of the planet surface increases (almost) linearly by the Stefan-Boltzmann rule. There is only a small range of luminosities where the planet surface temperature is suitable for daisy growth, so we expect few daisies to appear. This is clearly not a situation where life that requires a certain temperature to grow would thrive. It is this type of environment that life without Gaia faces. Whatever biota are present must be able to survive in the given environmental conditions. As we see in the graphs of Figs. 44.5 and 44.6, daisies are only present during the short interval in which the planet's surface temperature is appropriate for their growth. Daisies are entirely subject to their environment.

How should a Gaian Daisyworld respond to changing solar flux? Lovelock asserts that the daisies should regulate the temperature of Daisyworld at the optimum growth temperature, 25 °C, over a range of solar luminosities. Furthermore, perturbations that are not too large should be accommodated by Daisyworld. The most important characteristic of Gaia is that Gaian feedbacks are homeostatic and should increase stability of the system. Indeed, Lovelock claims the model always shows greater stability with daisies than it does without them. Thus, if we find that Daisyworld is not more stable with daisies than without, then we can safely

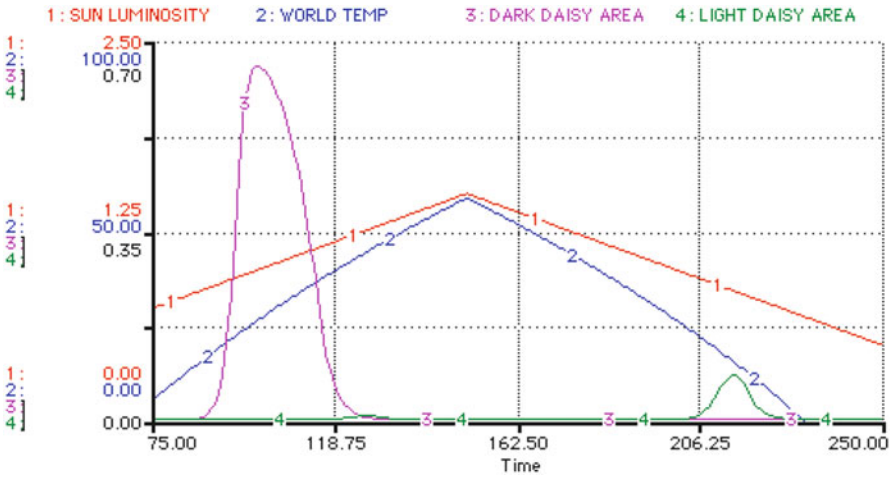

Fig. 44.5

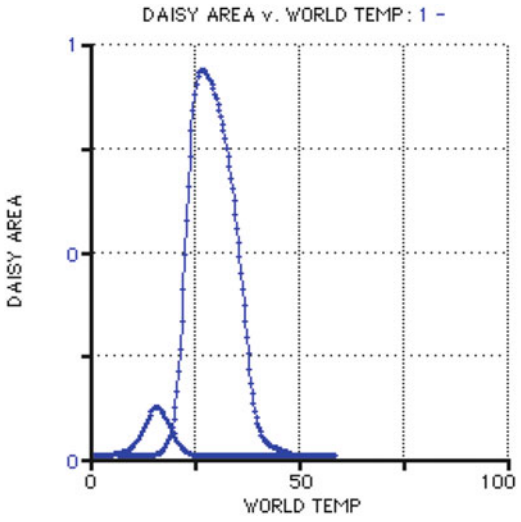

Fig. 44.6

assert that Daisyworld is not regulated by Gaia. The results in Figs. 44.5 and 44.6 provide a baseline for our assessment.

What happens when there is feedback between the biota and the environment? Figures 44.7, 44.8, and 44.9 show the "classic" Daisyworld which Lovelock has published numerous times. This is a fascinating graph and it demonstrates quite clearly that the daisies are capable without foresight, or planning, of regulating the temperature of Daisyworld to the optimum 25 °C. The graph shows the mean temperature, sun luminosity, and planet surface area covered by light and dark daisies.

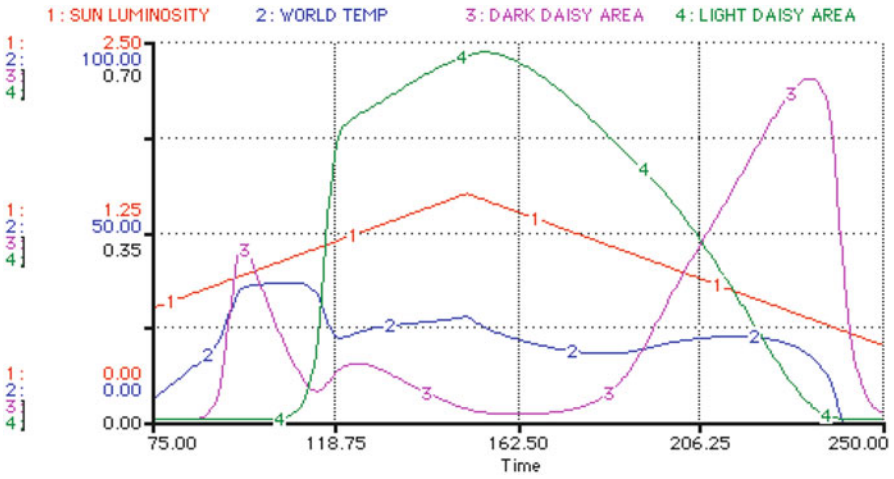

Fig. 44.7

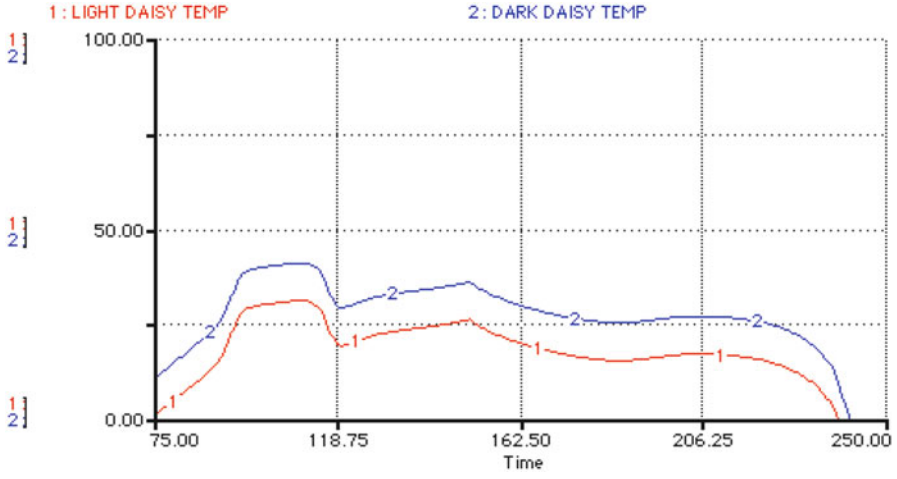

Fig. 44.8

When the solar flux on Daisyworld is so small that the world temperature is less than 25 °C, dark daisies flourish. Only when solar luminosity declines significantly can the daisies not survive.

Dark daisies, because they are dark, absorb more light than either the bare ground or light daisies, so they are warmer under the same solar flux conditions and can grow when it is too cool for light daisies. As the dark daisy population increases, the world temperature increases, and soon it is warm enough for light daisies. But the solar luminosity is increasing at the same time, so now Daisyworld

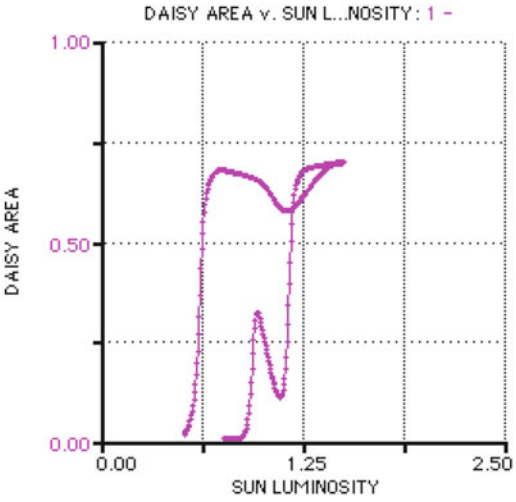

Fig. 44.9

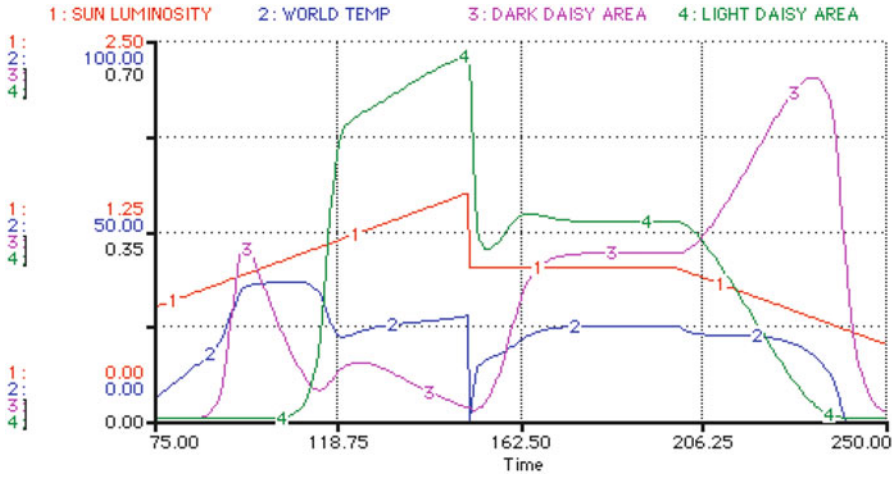

Fig. 44.10

is getting too hot for the dark daisies and they start to die off. The light daisies cool off the planet surface, so mean temperature drops. As one can see, the light and dark daisies adjust their respective populations to maintain the temperature of Daisyworld near the optimal 25 °C! The temperature regulation is not perfect, of course, in the case of only dark and light daisies. Increase the number of shades to improve Daisyworld.

Daisyworld must also be able to withstand perturbations more dramatic than a simple steady change of solar flux. The following shows a situation where there is a discontinuity in solar flux (Figs. 44.10 and 44.11).

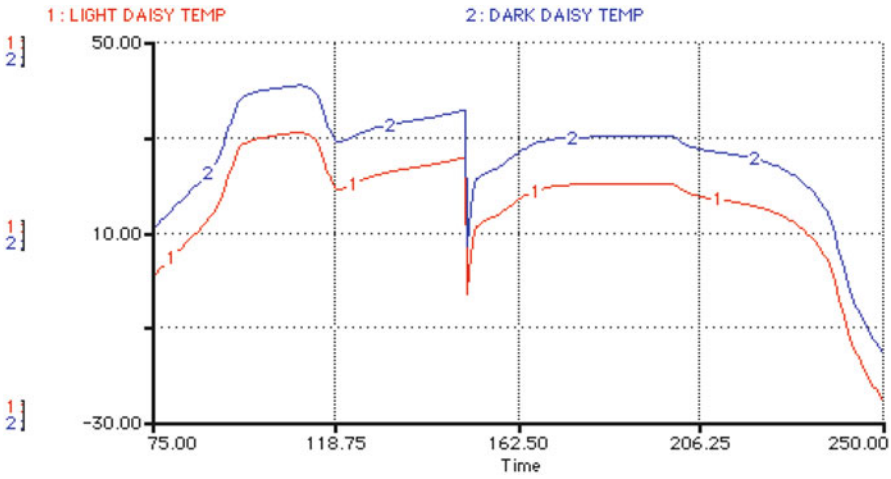

Fig. 44.11

$$\begin{aligned}\text{SUN LUMINOSITY} = {} & \text{IF TIME} < 150 \text{ THEN } 0.01 * \text{TIME} \\ & \text{ELSE IF TIME} \geq 150 \text{ AND TIME} \leq 200 \text{ THEN } 1 \\ & \text{ELSE } 1.5 - 0.01 * (\text{TIME} - 150)\end{aligned} \tag{44.13}$$

The effect of the jump in solar flux is to eradicate approximately half of the light daisies. But notice that Daisyworld recovers almost immediately! The dark daisies begin to grow in response to the lower temperature of Daisyworld and the newly available surface area no longer occupied by light daisies. Consequently, the temperature of Daisyworld is quickly returned to the optimum 25 °C.

This is a dramatic demonstration of the ability of Daisyworld to regulate itself to an optimum point and is certainly evidence in favor of Gaia. This kind of behavior is also realistic in that life on Earth has withstood such catastrophes before and survived, witness the extinction of the dinosaurs. Granted, dinosaurs were replaced by another genus altogether rather than a new bunch of the same dinosaur species, but the point is that Gaia must be robust enough to sustain life under drastically changing conditions.

Thus far we have demonstrated two characteristics of Gaia: regulation of the environment to a point optimal for the growth of the regulating life forms, and maintenance of that optimum even under severe shock. But what about the last, and probably most important point of the Gaia Hypothesis that there will be increased stability on the planet? In order to assess the implications of Gaia for the stability of the system you must first define what is meant by stability. You also must specify the range of perturbations that you allow for the system. For example, find a solar luminosity that yields a steady state for the LIGHT DAISY AREA and DARK

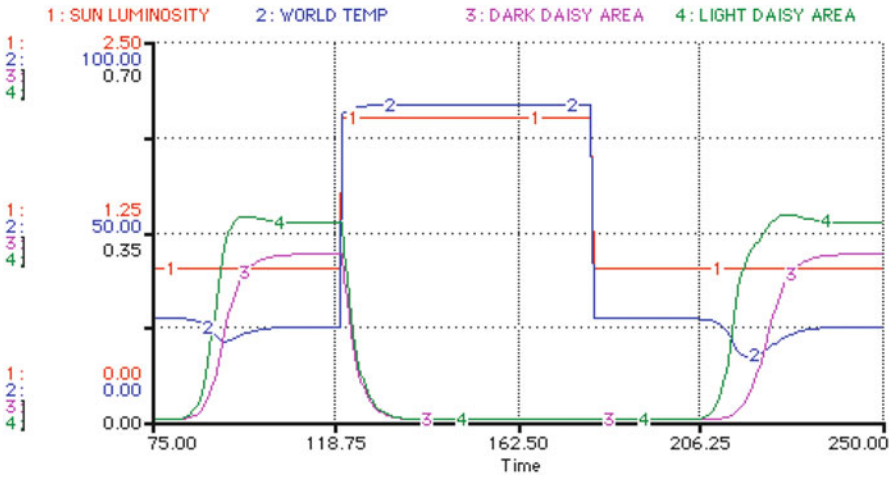

Fig. 44.12

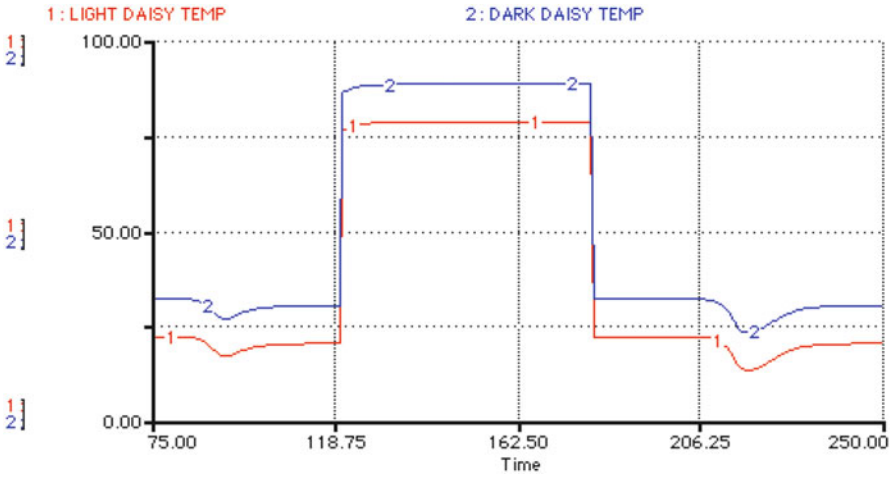

Fig. 44.13

DAISY AREA. Then perturb the system and note how close the system gets to the original steady state after disturbance occurred. We have done this with

$$\begin{aligned}\text{SUN LUMINOSITY} = \text{IF TIME} \geq 120 \text{ AND TIME} \\ \leq 180 \text{ THEN 2 ELSE 1}\end{aligned} \tag{44.14}$$

and the results are shown in Figs. 44.12, 44.13, and 44.14.

How are the results changed if the optimum temperature range for the two daisies differs? You will find for some ranges that there is hysteresis in Daisyworld,

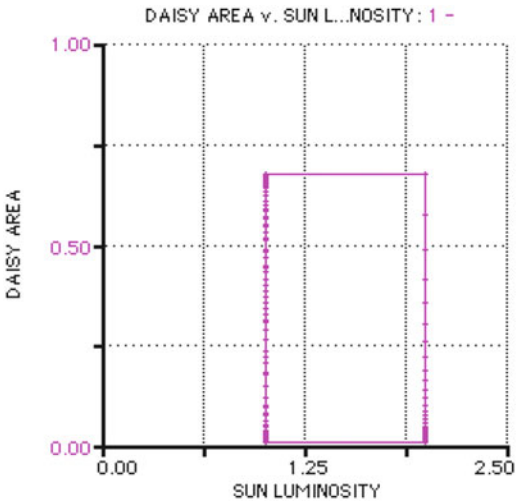

Fig. 44.14

implying that for a given amount of solar radiation on Daisyworld there are two stable temperatures and two stable daisy population sizes. How can this finding be reconciled with the notion of Gaia's stability?

44.2 Daisyworld Model Equations

DARK_DAISY_AREA(t) = DARK_DAISY_AREA(t − dt) + (DARK_DAISY_BIRTH − DARK_DAISY_DEATH) ∗ dt
INIT DARK_DAISY_AREA = 0.0001
INFLOWS:
DARK_DAISY_BIRTH = if ((DARK_DAISY_TEMP>5) and (DARK_DAISY_TEMP<45)) then DARK_DAISY_AREA∗NO_DAISY_AREA∗DARK_DAISY_GROWTH else 0
OUTFLOWS:
DARK_DAISY_DEATH = DARK_DAISY_AREA∗DEATH_RATE
LIGHT_DAISY_AREA(t) = LIGHT_DAISY_AREA(t − dt) + (LIGHT_DAISY_BIRTH − LIGHT_DAISY_DEATH) ∗ dt
INIT LIGHT_DAISY_AREA = 0.0001
INFLOWS:
LIGHT_DAISY_BIRTH = if ((LIGHT_DAISY_TEMP>5) and (LIGHT_DAISY_TEMP<45)) then LIGHT_DAISY_AREA∗NO_DAISY_AREA ∗LIGHT_DAISY_GROWTH else 0

OUTFLOWS:
LIGHT_DAISY_DEATH = LIGHT_DAISY_AREA*DEATH_RATE
DAISY_AREA = LIGHT_DAISY_AREA+DARK_DAISY_AREA {Fraction of area covered by daisies}
DARK_DAISY_ALBEDO = 0.25
DARK_DAISY_GROWTH = 1 − 0.003265*(25 − DARK_DAISY_TEMP)^2
DARK_DAISY_TEMP = LOCAL_TEMP_FUNC*(WORLD_ALBEDO − DARK_ DAISY_ALBEDO) + WORLD_TEMP
DEATH_RATE = 0.3
FERTILE_AREA = 1
GROUND_ALBEDO = 0.5
GROUND_AREA = 1 − DARK_DAISY_AREA − LIGHT_DAISY_AREA {Fraction of total area not covered by daisies}
LIGHT_DAISY_ALBEDO = 0.75
LIGHT_DAISY_GROWTH = 1 − 0.003265*(25 − LIGHT_DAISY_TEMP)^2
LIGHT_DAISY_TEMP = LOCAL_TEMP_FUNC*(WORLD_ALBEDO − LIGHT_DAISY_ALBEDO) + WORLD_TEMP
LIGHT_FLUX = 9.17e2 {W/(sec m^2)}
LOCAL_TEMP_FUNC = 20
NO_DAISY_AREA = FERTILE_AREA − LIGHT_DAISY_AREA − DARK_DAISY_AREA {Fraction of fertile area not covered by daisies}
STEFANS_CONSTANT = 5.6703e−8 {W/(m^2 K^4)}
SUN_LUMINOSITY = IF TIME < 150 THEN 0.01*TIME ELSE 1.5−0.01*(TIME−150)
WORLD_ALBEDO = (LIGHT_DAISY_AREA*LIGHT_DAISY_ALBEDO + DARK_DAISY_ALBEDO*DARK_DAISY_AREA + GROUND_AREA *GROUND_ALBEDO)
WORLD_TEMP = ((SUN_LUMINOSITY*LIGHT_FLUX*(1 − WORLD_ALBEDO))/STEFANS_CONSTANT)^0.25 − 273

References

1. Lovelock JE (1988) The ages of Gaia. Norton, New York
2. Watson AJ, Lovelock JE (1983) Biological homeostasis of the global environment. Tellus 35B(4):284

Part VIII
Conclusion

Chapter 45
Building a Modeling Community

I alone cannot change the world, but I can cast a stone across the waters to create many ripples.

(Mother Teresa)

The models and concepts that we developed in this book are powerful means to investigate the behavior of biological and ecological systems. The modeling approach that we chose is dynamic with regard to four issues. First, the systems that we modeled are dynamic ones, and we portray their dynamics, rather than use a static or comparative-static approach or a statistical model. Second, our model development process itself is dynamic. We encouraged you to start with simple models of complex systems. You will soon find that your models become increasingly complicated but more representative of the system about which you are asking important questions. STELLA, through its use of graphics, is an excellent tool to organize and assess the various aspects of the system that you wish to capture. Once the model system is sufficiently understood, you can easily move on to expand on the model and capture additional features of the real system.

Third, the learning process that accompanies model development and model runs is a dynamic one. By carefully phrasing the questions that the model should answer, and by stating the assumptions that underlie the model that we develop, we structure our knowledge about a system. By running the model and observing the results we learn about some aspect of a system. Subsequent model runs and model refinements provide more of this insight and should sharpen our focus for future model development and improve our intuition about the behavior of dynamic systems.

Fourth, through building and running computer models we provide a basis for communicating data and model assumptions. Frequently, model efforts become

A save-disabled version of STELLA and the computer models of this book are available at www.iseesystems.com/modelingdynamicbiologicalsystems.

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*,
Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9_45,

large-scale multidisciplinary endeavors. STELLA is sufficiently versatile to enable development of large-scale dynamic models. Such models can include a variety of features that are typically not dealt with by an individual modeler. Through easy incorporation of new modules into existing dynamic models and flexibility in adjusting models to specific real-world problems, STELLA fosters dialog and collaboration among modelers. It is a superb organizing and knowledge-capturing device for model building in an interdisciplinary arena. Individuals can easily integrate their knowledge into a STELLA model without "losing sight" of, or influence on, their particular part of the model. We anticipate that the modeling approach presented in this book will increase interaction among modelers and will be generating new momentum for interdisciplinary and cross-cultural exchange of ideas.

With the various books in the Modeling Dynamic Systems book series, of which this volume is a part, we wish to initiate a dialogue with and among you and other modelers. We invite you to share with us your ideas, suggestions and criticisms of the book, its models, and its presentation format. We also encourage you to send us your best STELLA models. We intend to make the best models available to a larger audience, possibly in the form of books, acknowledging you or your group as one of the selected contributors. The models will be chosen based on their simplicity, their application to an interesting phenomenon, or real-world problem.

We believe that the dynamic modeling enthusiasm, the ecolate skill, spreads by word of mouth, by people in groups of two or three sitting around a computer doing this modeling together, building a new model, or reviewing one by another such group. Share your thoughts and insights with us, and through us, with other modelers.

Bruce Hannon
University of Illinois
Urbana, IL, USA

Matthias Ruth
Northeastern University
Boston, MA, USA

Index

B. Hannon and M. Ruth, *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, DOI 10.1007/978-3-319-05615-9,

D

E