10

Population Dynamics Modeling

Preview

Until now, we have dealt with *deductive modeling* exclusively, i.e., all our models were created on the basis of a physical understanding of the processes that we wished to capture. Therefore, this type of modeling is also frequently referred to as *physical modeling.* As we proceed to more and more complex systems, less and less meta–knowledge is available that would support physical modeling. Furthermore, the larger uncertainties inherent in most physical parameters of such systems make physical models less and less accurate. Consequently, researchers in fields such as biology or economy often prefer an entirely different approach to modeling. They make observations about the system under study, and then try to fit a model to the observed data. This modeling approach is called *inductive modeling.* The structural and parametric assumptions behind inductive models are not based on physical intuition, but on factual observation. This chapter illustrates some of the virtues and vices associated with inductive modeling, and it addresses the question under what conditions which of the two approaches to modeling is more adequate, and why this is so. This chapter documents how population dynamics models are created, it discusses the difference between structural and behavior complexity of models in general, it introduces the concept of chaotic motion, and it finally addresses a rather difficult issue, namely the question of *self–organization* within systems.

10.1 Growth and Decay

In any population dynamics study, the change of the population per unit time can be described through the difference between the birth rate and the death rate:

$$\dot{P} = BR - DR \tag{10.1}$$

where BR denotes the "birth rate" which includes both physical birth and migration into the system, and DR stands for the "death rate" which again includes both physical death and migration out of the system.

It is natural to assume that both the birth rate and the death rate are proportional to the current population:

$$BR = k_{BR} \cdot P \quad , \quad DR = k_{DR} \cdot P \tag{10.2}$$

and therefore:

$$\dot{P} = (k_{BR} - k_{DR}) \cdot P \tag{10.3}$$

If we assume that k_{BR} and k_{DR} are two constants, the model exhibits either an *exponential growth* of the population (if $k_{BR} > k_{DR}$), or it shows an *exponential decay* of the system (if $k_{BR} < k_{DR}$).

However, is the assumption of constant birth and death factors k_{BR} and k_{DR} indeed justified? In order for a population to grow, it must consume some form of energy. Since, in any closed system, the energy is limited, a population cannot grow exponentially forever. If we assume that the available energy is equally distributed among all members of the population, then we can conclude that the *per capita* energy $E_{p.c.}$ is inverse proportional to the population count:

$$E_{p.c.} = \frac{E_{total}}{P} \tag{10.4}$$

Both the birth factor and the death factor will somehow depend on the available energy. If the *per capita* energy becomes small, the birth factor declines (since malnutrition leads to impotence and increased infant mortality), and the death factor increases (due to famine and due to an increased vulnerability to epidemic diseases). Eventually, an equilibrium will be reached in which the death factor balances the birth factor out, thus the population reaches a steady–state, therefore, the *per capita* energy does not decrease any further, and therefore, the birth factor and the death factor remain constant as well. The population dependence of the birth and death factors is usually referred to as the *crowding effect.*

Unfortunately, a physical law that would allow us to specify an explicit and accurate relationship between the *per capita* energy and the birth and death factors is unknown, and we must therefore base our model on *inductive evidence* rather than on *deductive evidence.*

The most commonly made assumption is the following: A one species ecosystem can support a fixed number of animals (plants) over an extended period of time. Let us call this number P_{max}. We can model the smooth growth and saturation of the population in the following way:

$$\dot{P} = k \cdot (1.0 - \frac{P}{P_{max}}) \cdot P \tag{10.5}$$

where k denotes the difference between k_{BR} and k_{DR}. If the population P approaches its maximum allowed value P_{max}, the factor in the paranthesis becomes smaller and smaller, and it finally reaches zero for $P = P_{max}$. Thus, the population stops growing. Fig.10.1 shows the behavior of this model for the same initial population P_0, for the same maximum population P_{max}, but for three different values of k.

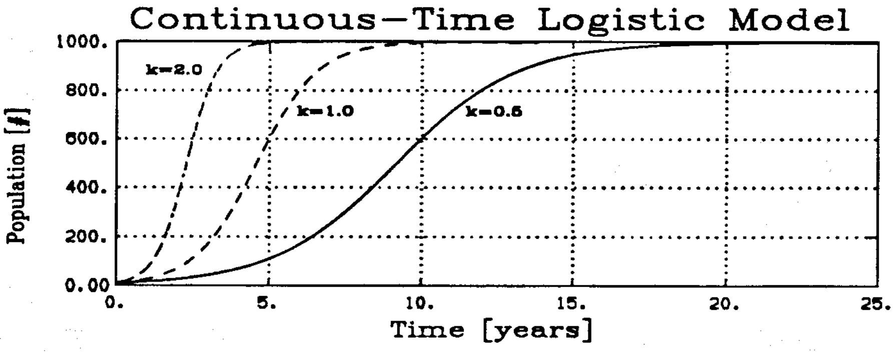


Figure 10.1. Trajectory behavior of the continuous–time logistic model

Eq(10.5) is called the *continuous–time logistic equation.* Notice that the crowding effect is modeled through a *quadratic term* in the population. For instance, the two–species model of hw(H10.1) uses the logistic approach to modeling the crowding effects within each of the two species.

Notice that absolutely no physical evidence is available which supports the correctness of the logistic model. However, it is true that many biological species exhibit a population dynamics behavior which can be fitted rather accurately with a logistic model.

The logistic model has several obvious disadvantages though.

(1) In order to identify the P_{max} value of a logistic model with reasonable accuracy, we must wait until a steady–state has actually occurred in the real system. However, this may be undesirable if the purpose of the model was to predict where the real system might settle to its steady–state value, and possibly, to find a means to influence the steady–state value of the real system before it ever occurs. For example, in world modeling, it is clearly undesirable if we have to wait until the human population has reached its steady–state value before we can construct a decent model for this fact.

(2) If an ecosystem contains several species, many different stable steady–state points may exist. It is the total energy of the system which is limited, but we have ample reason to assume that the ecosystem can support either more animals of one species and less of another or vice versa if both populations forage on the same food. Thus, it becomes difficult to come up with a meaningfully founded value for P_{max} under any circumstance, even if a steady–state has been observed in the real system.

Since inductive models are based on observation, such models are difficult to validate beyond the observed facts. Consequently, we may be forced to observe a disaster before we can devise a strategy that might have prevented it in the first place, had the strategy been known earlier. This is a clear disadvantage of inductive modeling.

Let me illustrate this problem a little further. Fig.10.2 shows the answer to hw(H10.4).

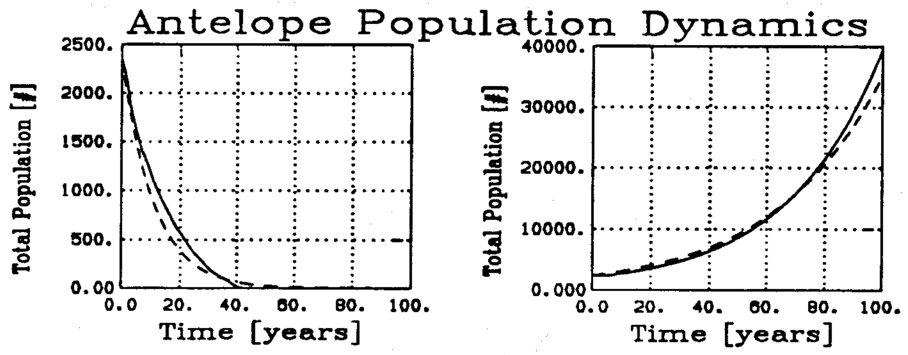


Figure 10.2. Antelope population dynamics with exponential fitting

Without giving the model away, we can analyze these results a little further. Hw(H10.3) presents a fairly accurate non–linear physical discrete–time model of an antelope population. Two simulations were performed which differ in the assumption made about the infant mortality rate. Other than that, the two models were identical. Hw(H10.4) then fitted exponential models to the results of hw(H10.3). Fig.10.2 shows the dramatic effect of the minor variation between the two models on the trajectory behavior. The continuous lines are the simulation results from the physical models, and the dashed lines represent the simulation results from the curve–fitted inductive meta–models. Fig.10.2 shows clearly that the exponential approximation is well justified. The available measurement data on which the models were based are certainly not accurate to such an extent that we could claim with confidence that the deductive physical model is more accurate than the inductive exponential model.

Yet, a very important difference exists between the two models. While the physical model requires more data items to be measured, we can obtain all the required data within a relatively short time span, say, two or three years. Once we measured the required data, we can predict the future behavior of the antelope population, and, if the model predicts the extinction of the entire population, we may be able to propose a countermeasure which could prevent the disaster from happening. As Fig.10.2 shows clearly, a minor modification in only a few physical parameters such as a minor change in rainfall statistics, an overgrazing by farm animals who compete for the same food, a new viral infection which increases the infant mortality by a few percent, or other seemingly insignificant changes in the environmental conditions can have drastic long term effects on the antelope population which may easily go undetected for quite some time. The fitting parameter of the exponential model is highly sensitive to changes in the physical parameters of the model. Without performing a simulation of the physical model over an extended time span, we have no decent way to estimate an accurate value of the fitting parameter. Consequently, if we rely on the inductive model alone, we may have to wait for a long time before we discover from measurements that a modification of the system parameters has occurred which endangers the population. By that time, it may be too late for any countermeasure to become effective.

We just learned that, while the simpler inductive model explains the data as well as does the more complex deductive model, and while it is normally advisable to operate on the simplest model that explains the available data adequately, other considerations may exist as well. In the above example, the fact that the deductive model can be identified accurately within a much shorter time span, is clearly a strong argument in favor of the deductive physical model.

Something else is interesting in the above example. While the physical system is clearly of high order, it behaves very similar to a low order system, i.e., inherent *structural complexity* of a system does not necessarily imply that the system must exhibit also *behavioral complexity*. This observation is much deeper than what could be expected on a first glance. We shall return to this discussion at a later time.

10.2 Predator–Prey Models

What happens if our ecosystem contains more than one species? It is clear that a model of a multi–species ecosystem must contain submodels that describe the behavior of the individual species, following the same arguments as before. However, we need additional terms that describe the interactions among the different species.

Obviously, the interaction terms must be such that no interaction takes place between two species if either of the two populations is extinct. The simplest expression that exhibits this property is the product of the two populations. If either of the two populations is zero, the product is zero as well.

The simplest model of this type is the *Lotka–Volterra model* [10.8]. It assumes that a population of predators x_{pred} forays on a population of preys x_{prey}. When a predator meets a prey, it gets fed, and a certain number of calories are exchanged. The predator population now has more calories, and the prey population has less. Usually, the Lotka–Volterra model introduces an efficiency factor, i.e., the prey population loses more calories than the predator population gains. It is also assumed that the predator population would die out when left without prey, while the prey population feeds on another species which is available in abundance and which is not itself contained in the model. Thus:

$$\dot{x}_{pred} = -a \cdot x_{pred} + k \cdot b \cdot x_{pred} \cdot x_{prey} \tag{10.6a}$$

$$\dot{x}_{prey} = c \cdot x_{prey} - b \cdot x_{pred} \cdot x_{prey} \tag{10.6b}$$

where $a > 0.0$ is the excess death rate of the predator population, $c > 0.0$ is the excess birth rate of the prey population, $b > 0.0$ is the foray (grazing) factor, and $0.0 < k \leq 1.0$ is the efficiency factor.

The Lotka–Volterra model exhibits a number of remarkable properties. In particular, this model does not approach a *continuous steady-state value.* Instead, it approaches a *periodic steady-state value*, i.e., the solution oscillates. The shape of the oscillation is very characteristic.

Fig.10.3 compares the measured data (dashed line) of a population of insects, the larch bud moth, *Zeiraphera diniana* (Guenée), which is endemic in the upper Engadine valley of south–eastern Switzerland, to simulation results (continuous line) stemming from an optimized inductive two species Lotka–Volterra model. The populations are expressed in larvae per kilogram of branches. Approximately once every nine years, we observe a large increase in the insect population. While the adult insects are quite harmless, their larvae feed on the needles of the larch trees, causing a lot of damage to the larch forest. Over several square kilometers, the green trees turn brown [10.6]. Therefore, it can be assumed that the insects are the predators in a predator–prey relationship with the larches being the prey. This assumption provided the rationale for the construction of the inductive model. Only the predator population is shown on the graph.

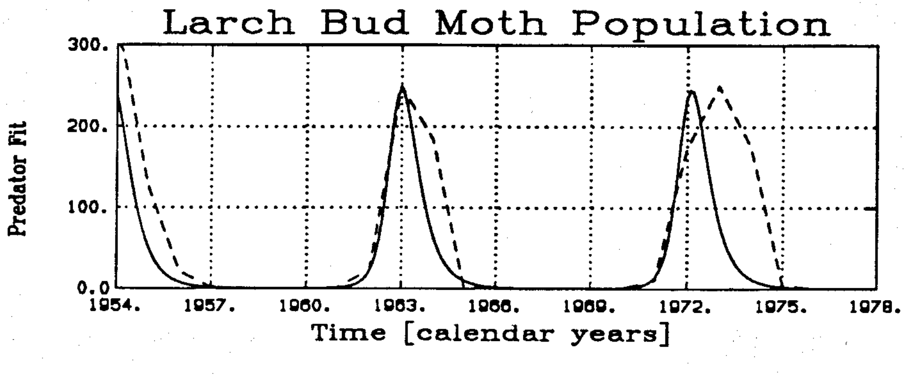


Figure 10.3. Lotka–Volterra model compared to measured insect data

The attractiveness of the Lotka–Volterra model becomes immediately evident from a comparison of the two curves. Clearly, our mathematical model reflects very well the data that have been observed. However, please remember that we have absolutely no physical evidence which would give credibility to the internal structure of our equations. All we can say is that this model fits well the measured data, and since we operate under periodic steady–state conditions, i.e., no trend exists in either the measured or the simulated data, this model can be used very well to predict the insect population over a long time span, given that no significant change occurs in climatic or other environmental conditions.

Notice that a good curve fit does not prove our equations to be correct. It does not even prove that our initial assumption of the larch bud moth being the predator in a two species predator–prey relationship is correct.

It was also observed that the adult insects suffer a lot from parasites. Obviously, the danger of epidemic deseases grows with the density of the insect population. Thus, it is equally reasonable to assume that the larch bud moth is in fact the prey in a two species predator–prey relationship with the parasites being the predator. Fig.10.4 shows another two species Lotka–Volterra model which was optimized under this new assumption. Again, the dashed line depicts the measured data, while the continuous line represents the simulated data. This time, only the prey population is shown on the graph.

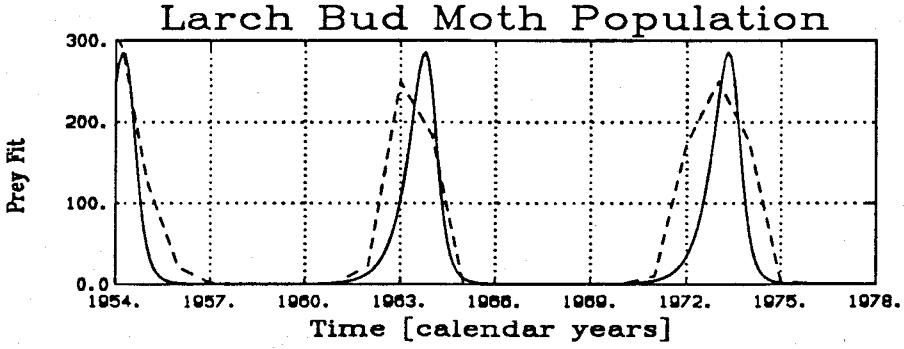


Figure 10.4. Lotka–Volterra model compared to measured insect data

Just from looking at these two simulations, we cannot say with certainty that either of the two assumptions leads to a significantly better curve fit, and is therefore more likely to be accurate. Consequently, we must be super cautious about concluding causal relationships among variables in a real system on the basis of a good curve fit by an inductively constructed model.

Fig.10.5 shows the simulation results (continuous lines) of the same two models once more, superposed with the measured data (dashed lines), but this time plotted on a semi–logarithmic scale.

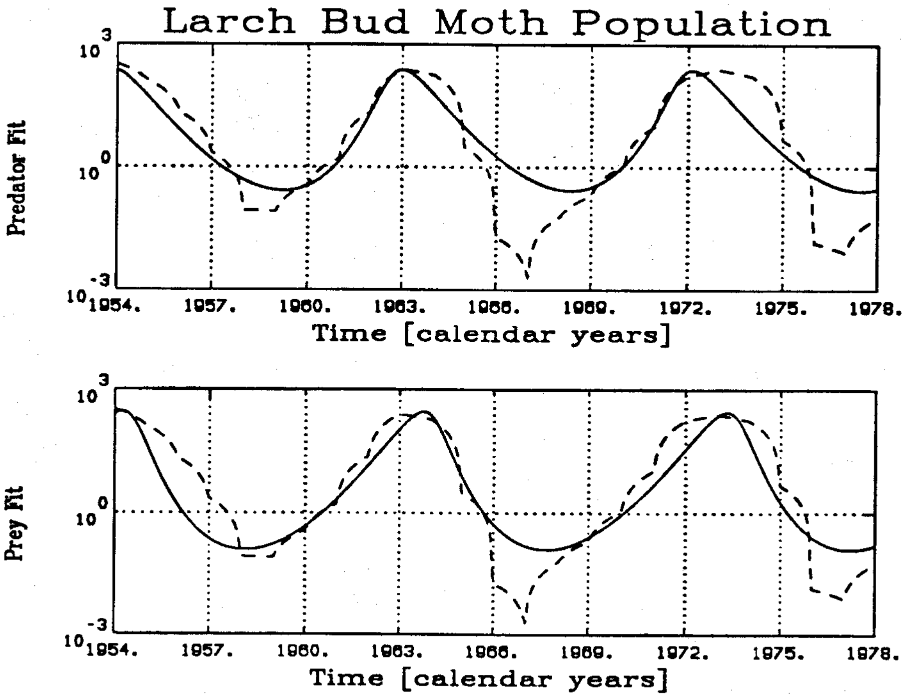


Figure 10.5. Lotka–Volterra models compared to measured insect data

This presentation shows that the predator fit exhibits a slow decay and a fast rise, whereas the prey fit exhibits a fast decay and a slow rise. The second model gives a slightly better fit, but the difference is certainly not significant.

Maybe, some truth lies in both assumptions, and we should use a three species Lotka–Volterra model with the insect being the "sandwich" population between the predator and the prey. Fig.10.6 compares a somewhat optimized three species Lotka–Volterra model to the measured data, but I didn't try real hard to get the best possible fit. The insect population is shown once using a linear scale, and once using a semi–logarithmic scale.

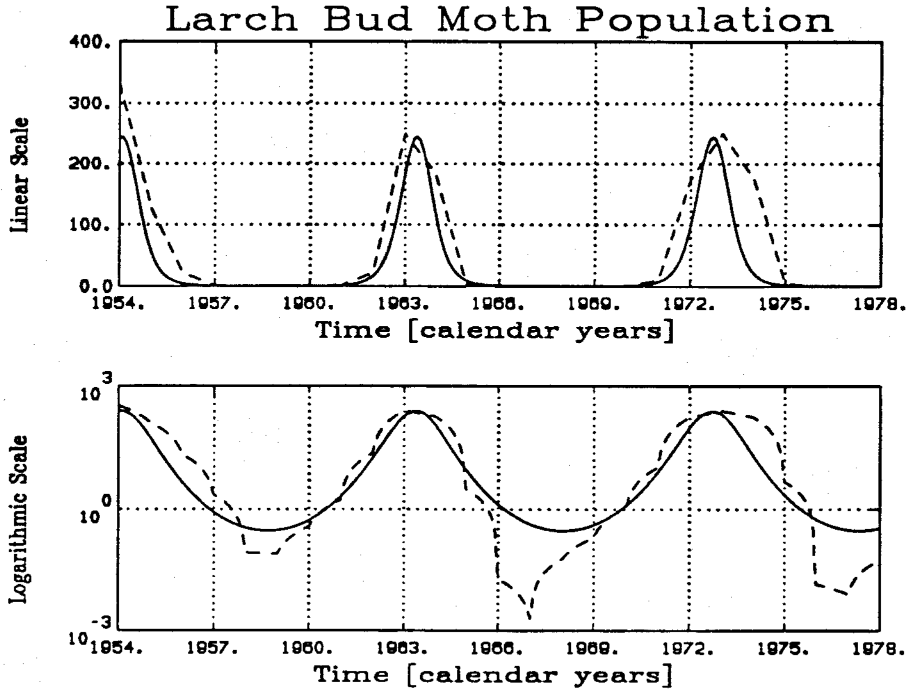


Figure 10.6. Three species model compared to measured insect data

This time, we observe a more symmetric rise/decay behavior. I am sure that, by trying a little harder, I could obtain an even better fit than shown on Fig.10.6. Yet, this is not the important message. Clearly, the measured data are insufficient to validate and therefore justify the more complex three species model.

Notice the difference in argument here. Previously, I argued in favor of a complex physical model over a simple inductive model. That is different. We need much less measurement data to justify a complex physical model than we need to justify a complex inductive model.

Notice that, just by adding sufficiently many parameters to our model, we can fit virtually *any* model structure to virtually *any* data. This is one of the most severe problems with many inductive modeling techniques. Inductive models lure us into love affairs with unhappy ending. In Chapter 13 of this text, I shall present another inductive modeling technique which does not suffer from this disease. It will simply refuse to let us identify models that exhibit a larger complexity than what is justifiable from the amount of available measured data.

Analyzing the above example a little further, we notice one of the major problems with modeling biological systems. Since we, the modelers, are a biological system ourselves, it is not surprising that the *time constants* of the biological systems to be modeled are generally of the same order of magnitude as our own time constants, i.e., in population dynamics, it usually takes *years* to obtain data that can support modeling. In the case of the larch bud moth ecosystem, the researchers had to spend three weeks every fall in the (undeniably very beautiful) upper Engadine valley during 30 *years*, climb trees, and count needles and larvae, to obtain the (still meager) data points used to validate their models. What a difference with electrical circuitry!

Fischlin and Baltensweiler [10.6] carried their model identification even further. They argued that the insects breed only during a relatively short time period every year. Therefore, a discrete–time model with a sampling period of one year may be a more reasonable assumption that the continuous–time Lotka–Volterra models shown in this chapter. Then, they argued that the Engadine valley is not a homogeneous ecosystem. Different parts of the valley exhibit different insect populations at different times. This claim was supported by data. Data had indeed been collected for different spots in the valley separately. Therefore, they compartmentalized the valley into 20 different segments. They used a *semi–physical* modeling technique, one that I shall introduce in Chapter 11, to model the dynamics within each segment, and they included the migration of insects between these segments in their model. Their modeling technique allowed them to create a model of this ecosystem which is strictly based on causal relationships among variables. The parameters of the model were identified individually on the basis of measurement statistics. Their model was able to predict the insect population fairly well without need to apply a global postoptimization to trim the model parameters. In Chapter 11, I shall present a simplified version of their model.

Postoptimization is, of course, a questionable enterprise. This model contains so many parameters that chances are that a good fit with the measured data could be found even if the model structure were incorrect. This problem can be overcome by assigning tolerance bands to all parameters of the model which describe the inaccuracy of the deductive model, i.e., parameters that have been well established are assigned narrow tolerance bands, while parameters for which few measurement data are available are assigned wide tolerance bands. We then could restrict the postoptimization in such a way that every single one of the parameters must remain within its assigned tolerance band.

The optimizations shown in this section are quite tricky. In the second volume of this text, I shall explain how they were accomplished. In fact, the optimization problems demonstrated in this section will turn into excellent homework problems then. But let's try to swallow one bite at a time.

10.3 Competition and Cooperation

In addition to the previously discussed predator–prey relationships among different species, two other relationships deserve to be mentioned: *competition* and *cooperation*.

A competitive situation occurs typically when several species compete for the same food source. In a way, this is similar to crowding, i.e., the more densely populated an area is, the more severe the competition will become. Similar to crowding, competition should thus be modeled as a quadratic effect. As in the predator–prey situation, if one of the populations is extinct, the competition stops. Consequently, we model competition again as a crossproduct of the competing populations, but this time, it appears with a negative sign in both equations, i.e.:

$$\dot{x}_1 = a \cdot x_1 - b \cdot x_1 \cdot x_2 \tag{10.7a}$$

$$\dot{x}_2 = c \cdot x_2 - d \cdot x_1 \cdot x_2 \tag{10.7b}$$

Here, both populations are expected to grow exponentially if they are not inhibited by competition, but due to competition, neither of the two populations will be able to grow forever.

The opposite situation is called *cooperation*. Cooperation occurs naturally in a variety of situations. A typical scenario is the *symbiosis* among two species. Neither of the two species can survive without the other.

Eq(10.8) describes the typical cooperation model:

$$\dot{x}_1 = -a \cdot x_1 + b \cdot x_1 \cdot x_2 \tag{10.8a}$$
$$\dot{x}_2 = -c \cdot x_2 + d \cdot x_1 \cdot x_2 \tag{10.8b}$$

Both populations have a tendency to decay, but in cooperation, a stable equilibrium can be reached which saves both populations from a destiny of extinction.

A similar effect is *grouping*. Grouping is the inverse mechanism of *crowding*. Many animals travel in herds, because in a herd, they are less vulnerable to being attacked by another species. A typical grouping model is shown below:

$$\dot{x} = -a \cdot x + b \cdot x^2 \tag{10.9}$$

Of course, all these effects can occur together, i.e., the typical population dynamics model involving n species can be written as:

$$\dot{x}_i = (a_i + \sum_{j=1}^{n} b_{ij} x_j) x_i \quad , \quad \forall i \in [1, n] \tag{10.10}$$

or using a matrix notation:

$$\dot{\mathbf{x}} = \big(diag(\mathbf{a}) + diag(\mathbf{x}) \cdot \mathbf{B}\big) \cdot \mathbf{x} \tag{10.11}$$

where the **a** vector captures the balance between birth rate factors and death rate factors, the diagonal elements of the **B** matrix describe the balance between grouping and crowding factors, and the off–diagonal elements of the **B** matrix cover the balance between cooperation and competition factors. They also include the predator–prey situation in which the predator considers the prey to be "cooperative", while the prey considers the predator to be "competitive". The *diag* function turns its argument vector into a diagonal matrix.

This model assumes that all relationships between species are binary, i.e., no crossproducts exist between more than two species. This is meaningful. If three species compete for the same food source, the competition terms should be modeled as:

$$-b_{12} \cdot x_1 \cdot x_2 - b_{13} \cdot x_1 \cdot x_3 - b_{23} \cdot x_2 \cdot x_3 \tag{10.12a}$$

and not as:

$$-b \cdot x_1 \cdot x_2 \cdot x_3 \tag{10.12b}$$

since, if one of the three populations is extinct, the other two still compete for the same food. Eq(10.12b) would suggest that competition stops as soon as one of the three populations dies out.

The situation would be quite different if we were to assume a symbiosis among three different species in which neither of the species can survive without the other two, but to my knowledge, such a phenomenon has never been observed on this planet.

What are the lessons learned? In short, we can postulate the following three rules:

(1) If we have a choice between a decent physical model and an inductive model, we always choose the physical model, even if it is more complex than the equivalent inductive model.

(2) If we have to content ourselves with an inductive model, we choose the simplest model that explains the data reasonably well. We start with a linear exponential growth/decay model, and add more terms until we obtain a reasonable fit. Among equally reasonable terms, we choose the one with the highest sensitivity, since it is probably the most important among the terms.

(3) We should avoid the love story cliff. We must be extremely self–critical when we try to deduct causal relationships among real system variables from an inductive model. Had this third rule been properly observed, Australia would not be plagued with rabbits and foxes today.

A more detailed discussion of the basic mechanisms of population dynamics can be found in a recent book by Edward Beltrami [10.1].

10.4 Chaos

Until now, every single autonomous system that we studied exhibited, in the steady–state, one of three types of behavior:

(1) *continuous steady–state*, i.e., after a long time, every variable in the system assumes a constant value,

(2) *periodic steady–state*, i.e., after all transients have died out, some variables oscillate with a fixed frequency, while others may assume constant values, or

(3) *no steady–state*, i.e., the transients never die out, on the contrary, they grow beyond all bounds, that is, the system is unstable.

An interesting question is whether these are indeed all possible types of behavior that autonomous systems can exhibit. Would it not be feasible that the transients in an autonomous system do not die out, and yet stay bounded? Let us discuss this question further by means of a three species population dynamics model.

The following set of equations describes one predator x_3 foraying on two different preys x_1 and x_2. The preys suffer furthermore from crowding and from competition:

$$\dot{x}_1 = x_1 - 0.001\, x_1^2 - 0.001\, x_1 x_2 - 0.01\, x_1 x_3 \tag{10.13a}$$

$$\dot{x}_2 = x_2 - 0.0015\, x_1 x_2 - 0.001\, x_2^2 - 0.001\, x_2 x_3 \tag{10.13b}$$

$$\dot{x}_3 = -x_3 + 0.005\, x_1 x_3 + 0.0005\, x_2 x_3 \tag{10.13c}$$

This set of equations has been analyzed by Michael Gilpin [10.7]. Fig.10.7 shows the trajectory behavior of this system simulated over 5000 time units. The initial populations of all three species were arbitrarily set to 100 each.

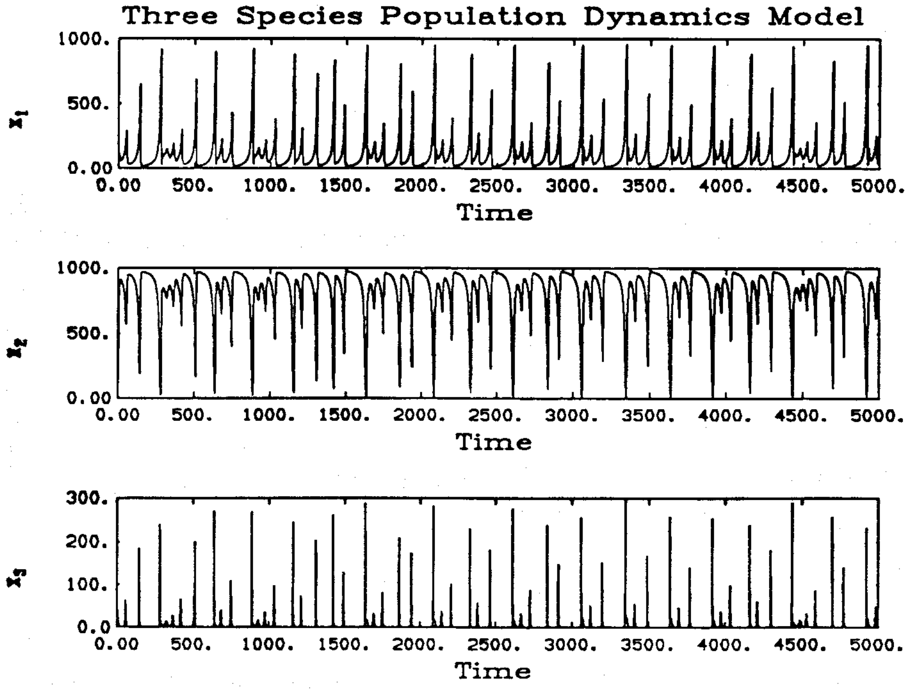


Figure 10.7. Chaotic three species Lotka–Volterra model

During most of the time, plenty of preys of the x_2 type are around, but from time to time, the predator population seems to explode. Then it reduces the x_2 population drastically. Since no more food is around, the predator population decreases again. Now, the x_1 population can grow which before was hampered by heavy competition from x_2. However, the x_2 population recovers quickly and resumes its dominant position in the ecosystem. And yet, each cycle seems to be a little different, and no periodic pattern seems to evolve. This type of behavior is called *chaotic motion.* Chaotic motion is characterized by a transient response which does not die out, yet which remains stable.

Let us look at the same problem in the phase plane. Fig.10.8 shows two of the three phase portrays:

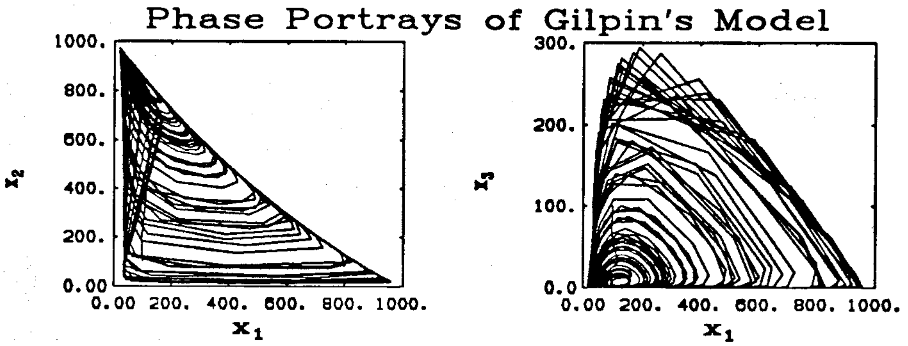


Figure 10.8. Chaotic Lotka–Volterra model in the phase plane

Most of the time, the trajectory lingers around the left upper corner of the $< x_1, x_2 >$ plane, and around the left lower corner of the $< x_1, x_3 >$ plane. The transitions away from these corners take place very rapidly, and they last only a very short time. The non–smooth characteristic of the $< x_1, x_3 >$ graph shows the influence of the numerical integration. In reality, this curve should be smooth. We must therefore be suspicious. Is this behavior for real, or is it simply an artifact of the numerical integration, i.e., if we were to integrate with higher precision on a machine with a large mantissa, would the behavior then turn out to be periodic, or is what we observe a real property of a real system?

In a rigorous sense, this question is still unanswered. Let me try to provide a better answer than has been given previously, but before

I do so, we want to analyze a much simpler example, an example which is so simple that we can tackle it analytically.

The discrete–time version of the logistic equation can be written as:

$$x_{k+1} = a \cdot x_k(1.0 - x_k) \tag{10.14}$$

Let us analyze the steady–state behavior of this equation as a function of the single parameter a. Fig.10.9 shows the trajectory behavior of this system for a random initial condition between 0.25 and 0.75, and for four different values of a chosen between 0.0 and 1.0. I plotted the left hand side of eq(10.14) as one curve, and the right hand side as another. In this way, we can solve the recursion equation graphically, which will be useful for a full understanding of what is going on.

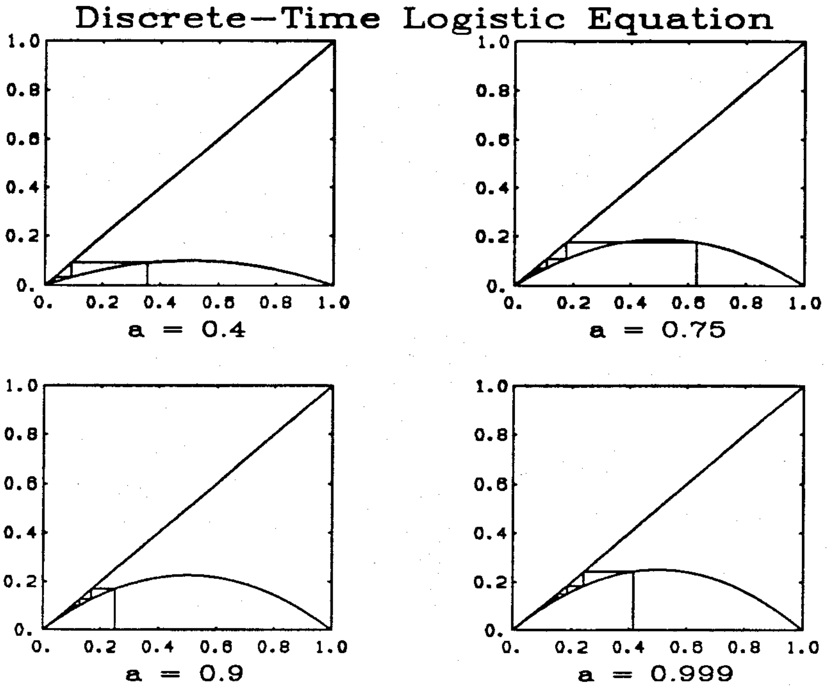


Figure 10.9. Discrete–time logistic equation

In the given range of the parameter a, the two curves intersect only in a single point, namely for $x = 0.0$. This turns out to be a stable continuous steady–state point of the discrete–time logistic equation.

As we approach $a = 1.0$, the area between the two curves becomes more and more narrow, and consequently, it will take more and more iterations to reach the steady–state point. At $a = 1.0$, this steady–state point becomes unstable. If you are curious about how Fig.10.9 was produced, solve hw(H10.6).

Fig.10.10 shows what happens in the range $a \in [1.0, 3.0]$.

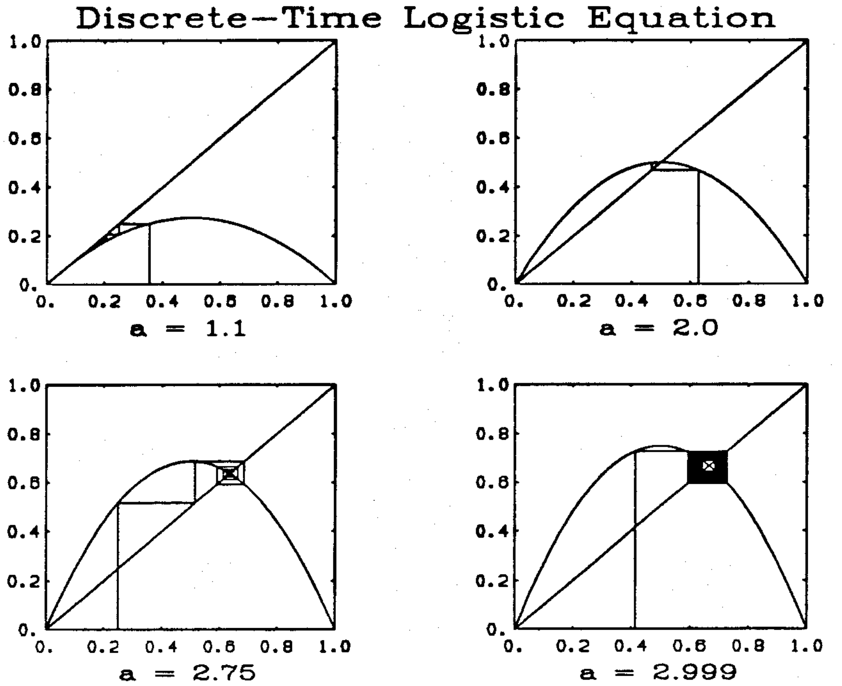


Figure 10.10. Discrete–time logistic equation

Now, the two curves intersect in two points. The second intersection is now a stable continuous steady–state solution of the discrete–time logistic equation. At $a = 1.0$, this solution is identical with the previous one, and is thus marginally stable. As we leave the vicinity of $a = 1.0$, the steady–state point becomes more and more stable. For example, at $a = 2.0$, it takes only very few iterations to reach the steady–state point. As we approach $a = 3.0$, this steady–state point becomes again less and less stable. At $a = 3.0$, we lose stability again. At $a = 3.0 - \epsilon$, it takes infinitely many iterations to reach the steady–state point.

Fig.10.11 shows what happens in the range $a \in [3.0, 3.5]$.

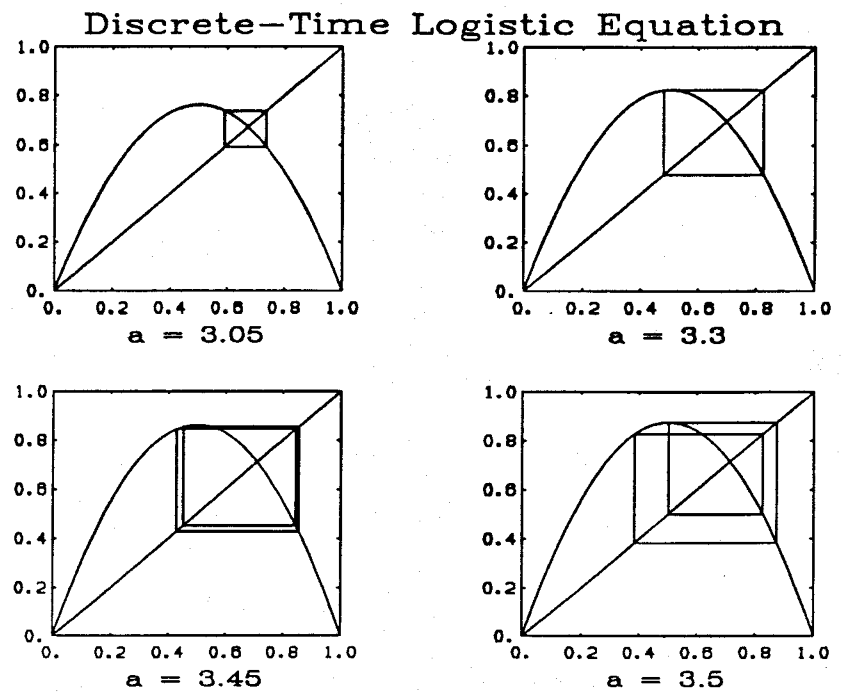


Figure 10.11. Discrete–time logistic equation

This time, the first 100 iterations were skipped in order to show the steady–state solution directly. At $a = 3.1$, a periodic steady–state exists with a period of 2 around the intersection of the two curves. As we approach $a = 3.0$ from the upper side, the steady-state rectangle shrinks more and more. At $a = 3.0 + \epsilon$, the rectangle is infinitely small, and it takes infinitely many iterations to reach it. As we leave the area of $a = 3.0$ in the positive direction, the periodic steady–state becomes more and more stable, i.e., it takes fewer and fewer iterations to reach the limit cycle. In the vicinity of $a = 3.45$, the next accident happens. Again, the solution has become marginally stable. Now, the limit cycle splits, and we obtain a new limit cycle with a period of 4.

Fig.10.12 shows what happens in the range $a \in [3.5, 4.0]$.

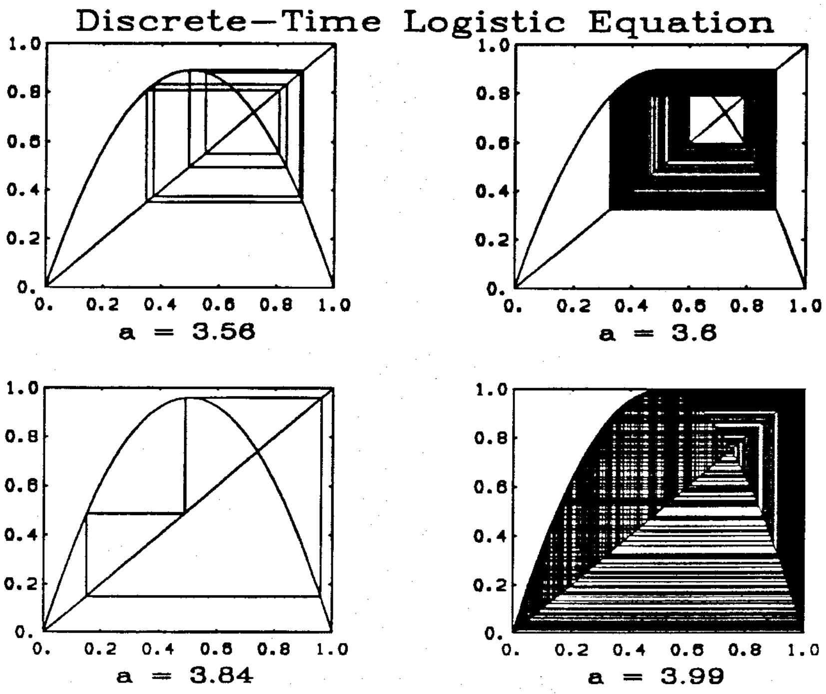


Figure 10.12. Discrete–time logistic equation

The stable intervals between two new bifurcations become smaller and smaller. Each time, the period of the periodic steady–state doubles. At $a = 3.56$, we observe a period of 8. Then, the period becomes *infinite*. At $a = 3.6$, the signal has become non–periodic, yet stable. We call this a *chaotic steady–state*.

However, even this is not the full truth. At $a = 3.84$, we observe another stable limit cycle, this time with a period of 3. At $a = 3.99$, we obtain a totally aperiodic behavior with "random" values of x anywhere between 0.0 and 1.0. Finally, at $a = 4.0$, the equation becomes unstable. In the second volume of this text, we shall return to this discussion, and analyze the properties of the logistic equation for $a = 3.99$ as a random number generator.

Fig.10.13 shows the behavior of the discrete–time logistic equation in a so–called *bifurcation map* for $a \in [2.9, 4.0]$.

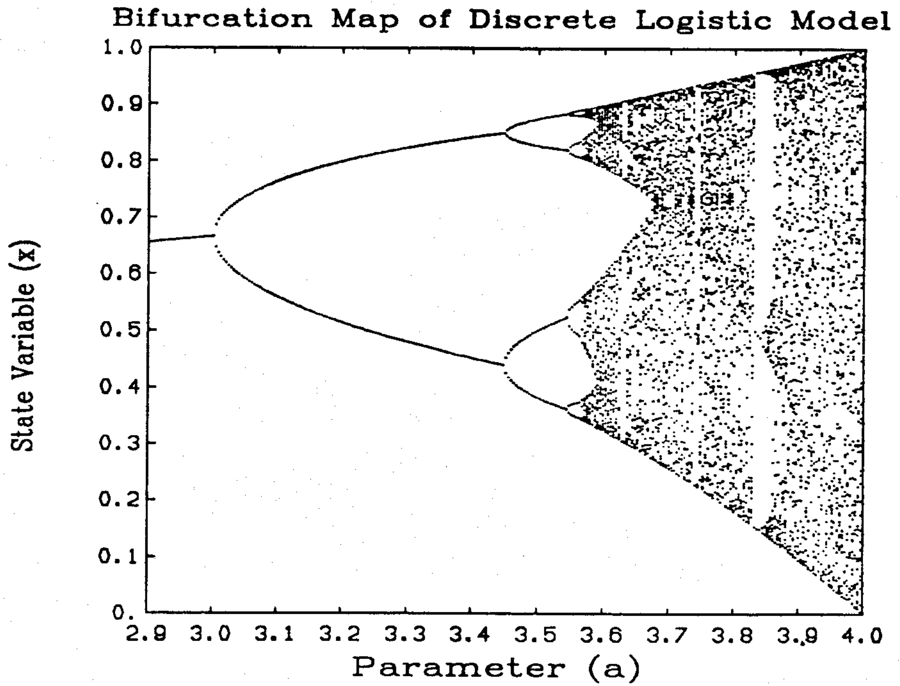


Figure 10.13. Bifurcation map of the discrete–time logistic equation

On the independent axis, the parameter a is varied. On the dependent axis, possible steady–state values of x are shown. For $a < 3.0$, only one such value exists, but for $a > 3.0$, several such values exist. The two branches of the map show the upper and the lower value of x in the periodic steady–state. The dark areas are chaotic. As can be seen, even for larger values of a, chaotic areas are interspersed with non–chaotic areas. If you are curious about how Fig.10.13 was created, solve hw(H10.7).

This time, we know for a fact that we have not observed merely an artifact. It is a straightforward (though tedious) exercise to compute the *bifurcation points*, i.e., the points on the bifurcation map where the number of branches doubles. Let me sketch an algorithm how this can be achieved. We start with the assumption of a continuous steady–state, i.e.:

$$x_{k+1} = a \cdot x_k(1.0 - x_k) \equiv x_k \quad , \quad k \rightarrow \infty \tag{10.15}$$

which has the two solutions $x_k = 0.0$, and $x_k = \frac{a-1.0}{a}$. We move the second solution into the origin with the transformation: $\xi_k = x_k - \frac{a-1.0}{a}$. This generates the difference equation:

$$\xi_{k+1} = -a \cdot \xi_k^2 + (2.0 - a)\xi_k \tag{10.16}$$

We linearize this difference equation around the origin, and find:

$$\xi_{k+1} = (2.0 - a)\xi_k \tag{10.17}$$

which is marginally stable for $a = 1.0$ and $a = 3.0$. Now, we repeat this analysis with the assumption of a periodic steady–state with period 2, thus:

$$x_{k+2} = a \cdot x_{k+1}(1.0 - x_{k+1}) \equiv x_k \quad , \quad k \rightarrow \infty \tag{10.18}$$

We evaluate eq(10.18) recursively, until x_{k+2} has become a function of x_k. This leaves us with a fourth order polynomial in x_k. Obviously, the previously found two solutions must also be solutions of this new equation. Thus, we can divide through, and obtain again a second order polynomial with two additional solutions. We move those into the origin, linearize, and obtain two new values for a, one of which will again be $a = 3.0$, the other will be the next bifurcation point. We continue in the same manner. Each time, the previous solutions will be solutions of the new equation as well. Of course, it is more meaningful, to perform this analysis on a computer rather than manually, and software systems exist meanwhile that allow us to solve such problems elegantly.

Notice that simulation is a very poor tool for determining the logistic bifurcation points accurately, since unfortunately, the logistic equation is marginally stable at every single one of the bifurcation points. A better numerical technique might be to exploit the marginal stability explicitly, i.e., compute root loci of the difference equations as a function of a, and solve for the points where the root loci intersect with the unit circle. However, this is of little concern to our cause. All we wanted to achieve was to convince ourselves that the bifurcations are for real, and are not merely artifacts. A more detailed discussion of analytical and semi–analytical techniques for the explicit evaluation of bifurcation points can be found in a paper by Mitchell Feigenbaum [10.5].

Let us now return to the Gilpin equations. These equations are far too complicated to perform a closed analysis as in the case of the logistic equation. The problem is the following. Even if we assume that we operate on a computer with a mantissa of infinite length, the numerical integration algorithm still converts our differential equations into difference equations, and we don't know for sure whether the observed chaotic behavior is due to the differential equations themselves, or whether it was introduced in the process of discretizing the differential equations into difference equations. Consequently, the observed chaotic behavior could, in fact, be a numerical artifact.

Several researchers have argued that an easy way to decide this question is to simply switch the integration algorithm. If the chaotic behavior occurs for the same parameter values, then it can be concluded that the numerical integration is not the culprit. Unfortunately, this argument does not hold. As we see clearly from Fig.10.7, the trajectory behavior of the Gilpin model is highly irregular, almost discontinuous at times. Consequently, a fixed step integration algorithm will compute garbage, while a variable step algorithm will reduce the step size to very small values in the vicinity of the spikes. As we shall see in the second volume, all currently advocated numerical integration algorithms are based on polynomial extrapolation. When the step size is reduced, the higher order terms in the approximation become less and less important. Ultimately, for a sufficiently small step size, every integration algorithm will behave numerically exactly like a forward Euler algorithm. Therefore, by switching the integration algorithm, we haven't really achieved anything. We have just replaced one Euler algorithm by another.

Let me propose another approach. We apply a logarithmic transformation to our populations:

$$y_i = \log(x_i) \tag{10.19}$$

Thereby, the Gilpin equations are transformed into:

$$\begin{aligned}
\dot{y}_1 &= 1.0 - 0.001\ \exp(y_1) - 0.001\ \exp(y_2) - 0.01\ \exp(y_3) && (10.20a)\\
\dot{y}_2 &= 1.0 - 0.0015\ \exp(y_1) - 0.001\ \exp(y_2) - 0.001\ \exp(y_3) && (10.20b)\\
\dot{y}_3 &= -1.0 + 0.005\ \exp(y_1) + 0.0005\ \exp(y_2) && (10.20c)
\end{aligned}$$

If we simulate eq(10.20) instead of eq(10.13), we are confronted with a numerically different problem. Yet, the analytical solution must be exactly the same. Thus, if eq(10.13) and eq(10.20) give rise to the same bifurcation map, we can indeed believe that the map was caused by the differential equations themselves, and had not been introduced in the process of discretization.

Let us apply the following experiment. The parameter that we are going to vary is the competition factor. For this purpose, we rewrite the Gilpin equations in the following way:

$$\dot{x}_1 = x_1 - 0.001\, x_1^2 - k \cdot 0.001\, x_1 x_2 - 0.01\, x_1 x_3 \tag{10.21a}$$
$$\dot{x}_2 = x_2 - k \cdot 0.0015\, x_1 x_2 - 0.001\, x_2^2 - 0.001\, x_2 x_3 \tag{10.21b}$$
$$\dot{x}_3 = -x_3 + 0.005\, x_1 x_3 + 0.0005\, x_2 x_3 \tag{10.21c}$$

For $k = 1.0$, we obtain the same solution as before. $k < 1.0$ reduces competition, while $k > 1.0$ increases competition. Fig.10.14 shows the behavior of the Gilpin model for a slightly reduced competition. Only the first prey x_1 is shown.

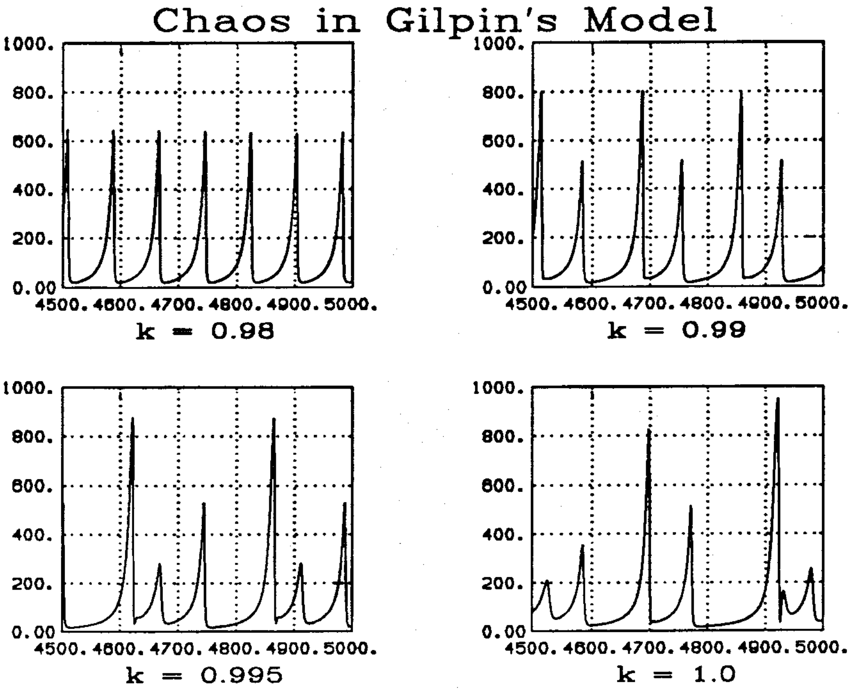
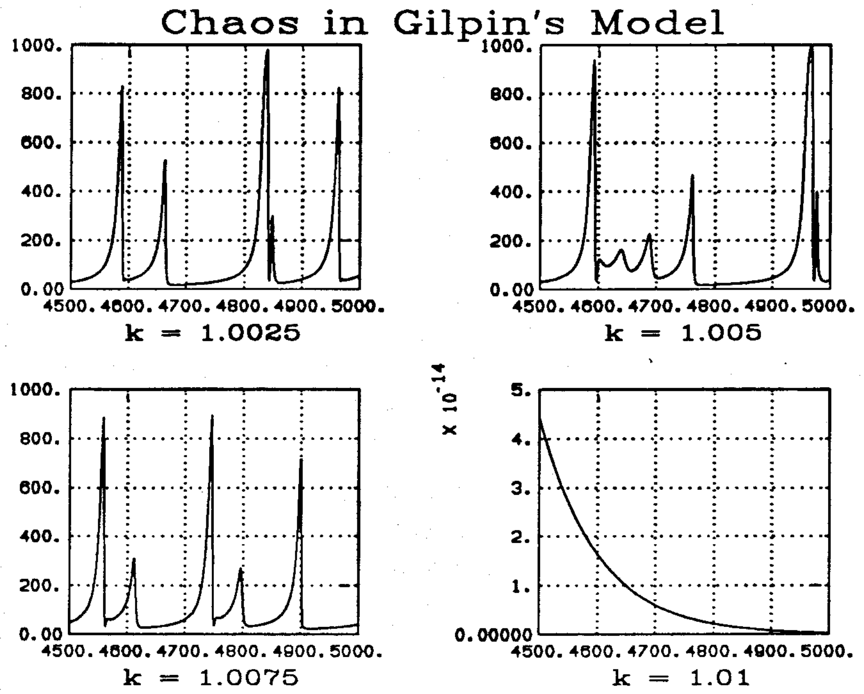


Figure 10.14. The influence of competition in Gilpin's model

With 98% competition, we observe a stable periodic steady–state with a "discrete period" of 1, meaning that each peak reaches the same height. For 99% competition, the discrete period has doubled.

For 99.5% competition, the period has tripled. Somewhere just below k = 1.0, the model turns chaotic. For lower values of competition, the Gilpin model exhibits a continuous steady–state, i.e., no oscillation occurs at all. Let us see what happens if we increase the competition.



k = 1.0025

k = 1.005

k = 1.0075

k = 1.01

Figure 10.15. The influence of competition in Gilpin's model

The behavior of the Gilpin model stays chaotic only up to $k = 1.0089$. For higher competition values, the x_1 population dies out altogether. We notice an incredible manifold of different possible behavioral patterns of this structurally simple model.

I repeated the same experiment for the modified Gilpin model, i.e., the model after application of the logarithmic transformation. The results were the same, except that in the chaotic cases, the curves looked different which is easily explained by the high sensitivity of the model to roundoff errors. We would need a computer with a mantissa of infinite length in order to obtain the same trajectories.

Fig.10.16 shows the two "bifurcation maps" of Gilpin's model with and without the logarithmic transformation. For this purpose, I

recorded all extrema of the x_1 population, i.e., the values for which $\dot{x}_1 = 0.0$.

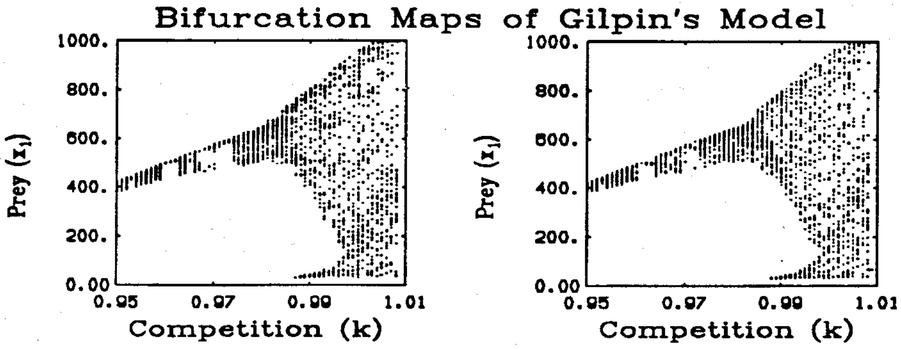


Figure 10.16. Bifurcation map of Gilpin's model

The bifurcation maps are somewhat inaccurate due to the large gradients in the immediate vicinity of the peaks. Yet, Fig.10.16 shows clearly that the two bifurcation maps are qualitatively the same. Thus, we conclude that the observed chaotic behavior is indeed a property of the physical system, and not merely an artifact of the numerical solution technique.

The Gilpin model is of course rather artificial. I don't think that a real system with such a high vulnerability to competition would be able to survive, and to thrive in a chaotic mode. Yet, this is a particularity of the Gilpin model, and not a property of chaos *per se*. Chaotic motion has been observed on numerous occasions. A good account of chaotic observations in biological systems can be found in Degn *et al* [10.3]. Chaotic motion is by no means limited to biological systems alone. I simply found this to be a good place to introduce the concept. A well written survey of the mechanisms of chaotic motion can be looked up in a recent book by Robert Devaney [10.4].

In the first section of this chapter, we analyzed the antelope population dynamics problem, which exhibited a fairly large structural complexity, and yet, the behavioral complexity of that model was very low. In the last section, we saw several examples of systems with very limited structural complexity which exhibited a stunning wealth of behavioral complexity. Obviously, structural and behavioral complexity do not have to go hand in hand. This comes as a surprise.

In earlier days, observed erratic behavior of physical processes had always been attributed to their structural complexity. Chaos theory has taught us that even structurally simple systems can exhibit an astounding variation of behavioral patterns. However, the reverse is also true. Structural complexity does not necessarily lead to behavioral complexity, and in fact, it commonly won't. This statement deserves additional consideration. The next section will provide just that.

10.5 The Forces of Creation

The behavioral complexity of a system is usually larger than its structural complexity. We see this already by means of the simplest differential equation system:

$$\dot{x} = -a \cdot x \quad , \quad x(t = 0.0) = x_0 \tag{10.22}$$

has the solution:

$$x(t) = \exp(-at) \cdot x_0 \tag{10.23}$$

While the differential equation itself is linear, its solution is already exponential. The differential equation describes the *structure* of the system, while the solution describes its *behavior.*

What chaos theory taught us is the fact that the behavioral complexity of a system can even be *much* greater than its structural complexity, more so than we had thought possible before chaos had been discovered.

This is the answer to the question which I posed at the end of Chapter 9. In the past, researchers looked at thermodynamics only in the steady–state. To do so, they analyzed a limited subset of the *behavioral* patterns of thermic systems, namely those observed under steady–state conditions. Yet, even those patterns were already quite complex, and it was necessary to distinguish between different types of behavioral patterns, such as those observed under reversible conditions *vs* those observed under irreversible conditions. By working with the differential equations directly, we concentrate on the *structural* patterns of thermic systems, and this turns out to be simpler.

The next question is: Which mechanisms exist that limit the behavioral complexity of a system? If we add more and more structural components to a system, why is it that its behavioral complexity does not grow beyond all bounds? I would like to identify three different mechanisms which limit system complexity.

(1) *Physical constraints:* In connecting two subsystems, their combined degrees of freedom are usually lower than the sum of the degrees of freedom of the subsystems.

(2) *Control mechanisms:* Controllers in a system tend to restrict the possible modes of behavior of a system.

(3) *Energy:* The laws of thermodynamics state that each system sheds as much energy as it can, i.e., it maintains the lowest amount of energy feasible. This mechanism also limits the complexity.

Let us discuss these three mechanisms in more detail. If we take a lever, and describe all possible motions of that lever, we need six differential equations to do so, i.e., we formulate Newton's law three times for the three translational motions, and three times for the three rotational motions. When we now take two levers, and we connect them in one point, we notice that the total number of differential equations needed to describe all possible motions of the combined system is *nine* rather than *twelve.* The connection between the two levers has introduced three *structural singularities.* We noticed this fact before when we analyzed the behavior of a DC–motor with a mechanical gear. The gear was responsible for a structural singularity. Consequently, physical constraints restrict the structural complexity of a system.

If we introduce a controller to a system, we reduce the sensitivity of that system to parameter variations. We could also say: a controller reduces the sensitivity of the behavior of a system to structural changes, i.e., the controller restricts the behavioral complexity of the system. A system with a controller will exhibit less modes of operation than the same system would if the controller were removed. This is why the device is called a "controller". It controls the behavior of a system. This is equally true for human made controllers as for controllers in nature. Why is it that all trees grow new leaves in spring, and shed their leaves in the fall? The control mechanisms built into the system regulate this behavior. Due to these controls, the trees react *uniformly*, and the overall behavioral complexity is much reduced. Without such control mechanisms, trees could grow leaves arbitrarily, and they would have more "freedom" in determining their behavior.

The third mechanism is energy. To return to our previous example, why is it that the motion of a lever can be described by six differential equations, although this metal bar contains billions of atoms? The true structural complexity of this system is phenomenal, and yet, its apparent structural complexity is very limited. The reason is simple: it is cheaper that way, i.e., the total energy content of the system can be reduced by forcing the atoms into a crystalline structure.

Obviously, these three mechanisms are not independent of each other. In our population dynamics models, *crowding* certainly exerts control over a population. Yet, the crowding effect is caused by energy considerations. The limited energy content of a closed system is a great regulator in population dynamics. Returning once more to our metal bar: The laws of thermodynamics manifest themselves in an apparent control mechanism which restricts the motion of the individual atoms. From a more macroscopic point of view, if we ignore the dynamics of this control mechanism, we can view the global effects of this control as a constraint. All atoms seem to move in unison, and therefore, we experience our metal bar as a rigid body.

So, why then is there any complexity at all in this universe? Why does our planet exhibit such a wealth of different systems and different behaviors? We don't know the answer to this question. However, we can observe that a second force is at work besides *energy*, which is the *entropy*. Every system tries to maximize its own entropy, i.e., it strives to reach a state of highest disorder. We don't know where this force comes from. We don't know whether the laws of thermodynamics are globally true, or whether they are local dynamic aberrations. Is entropy a global force, or is it simply a reminiscent of the dynamics of energy transfer in the big bang? We are currently unable to answer these questions, and my guess is, we never will. The laws of thermodynamics are *empirical laws*, i.e., they are based on observation and not on deduction. I believe that these laws are so fundamental to the functioning of this universe of ours, that we shall never be able to unravel their origin. Yet, within the framework of our understanding of physics, these laws have certainly been confirmed over and over again.

We notice that two separate forces are at work. *Entropy* (or rather the law of entropy maximization) is the great innovator. Entropy tries hard to "create a mess". *Energy* (or rather the law of energy minimization) is the great organizer. Energy tries hard to "clean up" behind the mess that entropy created. Together they manage the *evolution*.

I know that this a very simplistic view of an impossibly complex problem. I know all too well the dangers behind oversimplification. Didn't I warn constantly about the love story cliff? Yet, this view raises some interesting questions in the context of modeling. Is it possible to create a computer algorithm which somehow exhibits elements of evolution? First attempts in this direction have been reported. Brown and Vincent [10.2] obtained interesting results in the study of *evolutionary games.* This topic is also closely related to the area of *machine learning.* Several recent advances in machine learning can be viewed in the light of modeling evolution, in particular, some of the reported research efforts in *neural network learning*, and in *genetic algorithms.* We shall return to this discussion in Chapter 14 of this text. Yet, this research area is still in its infancy, and more research is highly encouraged.

From the above discussion, I conclude that any algorithm which attempts to replicate elements of evolution needs two separate mechanisms:

(1) *An Innovator,* i.e., a mechanism to generate new behavioral elements. In a computer model, this will probably have to be triggered by some sort of random number generator, and

(2) *An Organizer,*, i.e. a mechanism to restrict behavioral complexity and to guarantee overall system stability. In a computer model, this will probably have to be some sort of optimization algorithm.

Don't get me wrong. I do not suggest that *energy* and *entropy* should be made responsible for every single move in our daily lives. I am aware of the fact that modeling (by definition) must be reductionistic. The more highly organized a system is, the more will appear other influencing factors which will eventually dominate the behavior of the system. The shortcomings of "social Darwinism" have long been discovered, and the methodology has been discredited as a mechanism to describe e.g. the social behavior of human organizations. Yet, at the bottom of every system, underneath all other competing factors, the two primeval forces are always at work, the tidal forces of our universe, the laws of thermodynamics.

10.6 Summary

In this chapter, we analyzed basic models that describe the behavioral patterns of population dynamics. For this purpose, it was necessary to leave the road of physical modeling, and introduce a new concept, the technique of inductive modeling. We then proceeded to more advanced topics, we introduced the concept of chaotic motion, and we dealt with the relationship between structural complexity and behavioral complexity in a model.

Notice that the inductive modeling methodology proposed in this chapter is clearly limited to the discussion of population dynamics. In the next chapter, we shall generalize this idea, and introduce a methodology which will allow us to model a much wider variety of systems in a semi–physical and semi–inductive manner.

References

[10.1] Edward Beltrami (1987), *Mathematics for Dynamic Modeling*, Academic Press, Boston, MA.

[10.2] Joel S. Brown, and Thomas L. Vincent (1987), "A Theory for the Evolutionary Game", *Theoretical Population Biology*, **31**(1), pp. 140–166.

[10.3] Hans Degn, Arun V. Holden, and Lars F. Olsen, eds. (1987), *Chaos in Biological Systems*, Plenum Press, New York, NATO ASI Series, Series A: Life Sciences, **138**.

[10.4] Robert L. Devaney (1986), *An Introduction to Chaotic Dynamical Systems*, Benjamin/Cummings, Menlo Park, CA.

[10.5] Mitchell J. Feigenbaum (1978), "Quantitative Universality for a Chaos of Nonlinear Transformations", *J. Statistical Physics*, **19**(1), pp. 25–52.

[10.6] Andreas Fischlin, and Werner Baltensweiler (1979), "Systems Analysis of the Larch Bud Moth System. Part 1: The Larch — Larch Bud Moth Relationship", Mitteilungen der Schweizerischen Entomologischen Gesellschaft, **52**, pp. 273–289.

[10.7] Michael E. Gilpin (1979), "Spiral Chaos in a Predator–Prey Model", *The American Naturalist*, **113**, pp. 306–308.

[10.8] Alfred J. Lotka (1956), *Elements of Mathematical Biology*, Dover Publications, New York.

[10.9] R. B. Williams (1971), "Computer Simulation of Energy Flow in Cedar Bog Lake, Minnesota Based on the Classical Studies of Lindeman", in: *Systems Analysis and Simulation in Ecology, Vol.1*, (B.C. Patten, ed.), Academic Press, New York.

Bibliography

[B10.1] Alan Berryman, ed. (1988), *Dynamics of Forest Insect Populations: Patterns, Causes, Implications*, Plenum Press, New York.

[B10.2] Mitchell J. Feigenbaum (1980), "The Transition to Aperiodic Behavior in Turbulent Systems", *Commun. Mathematical Physics*, **77**, pp. 65–86.

[B10.3] Tom Fenchel (1987), *Ecology, Potentials and Limitations*, Ecology Institute, D–2124 Oldendorf/Luhe, FRG.

[B10.4] Michael Mesterton–Gibbons (1989), *A Concrete Approach to Mathematical Modelling*, Addison–Wesley, Redwood City, CA.

Homework Problems

[H10.1] Predator–Prey Model

A simple ecosystem consists of a population of seasnails p_s that forages on a population of algae p_a. The populations are measured in number of species per unit surface. The dynamics of this ecosystem can be described by two differential equations:

$$\dot{p}_s = b \cdot c_1 \cdot p_s \cdot p_a - c_2 \cdot p_s - c_3 \cdot p_s^2 \qquad (H10.1a)$$

$$\dot{p}_a = c_4 \cdot p_a - c_5 \cdot p_a^2 - c_1 \cdot p_s \cdot p_a \qquad (H10.1b)$$

where $c_1 = 10^{-3}$ is the grazing factor, $c_2 = 0.9$ is the excess mortality rate factor of the snails, $c_3 = 10^{-4}$ is the crowding ratio of the snails, $c_4 = 1.1$ is the excess reproduction rate of the plants, $c_5 = 10^{-5}$ is the crowding ratio of the plants, and $b = 0.02$ is the grazing efficiency factor. All constants have been converted to a per day basis.

An accident in a nearby chemical plant diminishes the two populations to values of $p_s = 10.0$ and $p_a = 100.0$. Simulate the ecosystem for a period of 30 *days* to check whether the two populations can recover.

Use simulation to determine the range of initial conditions from which the two populations can recover.

[H10.2] Linear Regression Model

Williams [10.9] developed a linear regression model for the ecosystem of Cedar Bog Lake, MN. The model includes three biological species: a population of carnivores x_c that feeds on a population of herbivores x_h which, in turn, feeds on a population of seaweed x_p. All populations are expressed in terms of their energy content measured in *cal* cm^{-2}. In addition, the model considers the biological sediment that forms on the bottom of the lake, and the loss of biomass to the environment (carried out of the lake with the water). The model is driven by solar energy x_s which enables the growth of the plants.

The model can be described by the following set of differential equations:

$$x_s = 95.9 \cdot (1.0 + 0.635 \cdot \sin(2\pi t)) \tag{H10.2a}$$

$$\dot{x}_p = x_s - 4.03 \cdot x_p \tag{H10.2b}$$

$$\dot{x}_h = 0.48 \cdot x_p - 17.87 \cdot x_h \tag{H10.2c}$$

$$\dot{x}_c = 4.85 \cdot x_h - 4.65 \cdot x_c \tag{H10.2d}$$

$$\dot{x}_0 = 2.55 \cdot x_p + 6.12 \cdot x_h + 1.95 \cdot x_c \tag{H10.2e}$$

$$\dot{x}_e = 1.0 \cdot x_p + 6.9 \cdot x_h + 2.7 \cdot x_c \tag{H10.2f}$$

All constants have been converted to a per year basis. The equation for x_s models the solar radiation as it changes over the year due to the varying position of the sun in the sky.

Assume the following initial conditions: $x_p = 0.83$, $x_h = 0.003$, $x_c = 0.0001$, $x_0 = x_e = 0.0$, and simulate this ecosystem over a period of 2 *years*.

Except for the variables x_0 and x_e, a *periodic steady-state* will occur, i.e., the three variables x_p, x_h, and x_c will become periodic with a period of 1 *year*. Think of a way how the periodic steady-state could be computed faster than by simply simulating the system over a long time.

[H10.3] Antelope Population Model

We wish to create a detailed model of the population dynamics of a breed of antelopes in Serengeti National Park at the border between Kenya and Tanzania (Eastern Africa). The antelopes are monogamous. They choose their partner at age three, and they stay together for the rest of their lives. When one partner dies, the other stays single except at age three before they had offspring when s/he would look for a new partner. All antelopes select partners of their own age. If a three year old cannot find a partner, s/he migrates out of the park.

All available adults mate and produce offspring every year between ages four and eight. Each year, every couple produces one calf. The probability of male/female offspring is 55% for male and 45% for female.

The mortality rates are given in Table H10.3a.

Table H10.3a Mortality rates

age [*years*]	male [%]	female [%]
1	60	40
2	10	10
3	5	5
4	5	8
5	5	8
6	5	8
7	6	9
8	7	11
9	10	10
10	25	25
11	70	70
12	100	100

Young females are a little stronger than young males. However, during the reproduction period, the females have a slightly higher mortality rate than their male partners. The numbers are given as percentages of the population of the same sex one year younger. The chances of dying are assumed to be statistically independent of the animal's societal status, i.e., single animals die equally often as mated animals, and the death of a mated animal does not influence the life expectancy of its partner.

We want to model this system through a set of difference equations (i.e., as a discrete–time system). $M_i(k)$ denotes the number of single male animals of age i in year k, $F_i(k)$ denotes the number of single females, and $C_i(k)$ denotes the number of couples.

Simulate this discrete–time system over 100 years. Assume the initial conditions as given in Table H10.3b:

Table H10.3b Initial conditions

age [*years*]	male [#]	female [#]	couples [#]
1	100	100	0
2	100	100	0
3	100	100	0
4	0	0	100
5	3	3	95
6	7	7	90
7	12	12	85
8	15	15	80
9	30	30	70
10	30	30	50
11	20	20	25
12	5	5	0

Use ACSL to model this system. This application won't require a DERIVATIVE block. The model is coded in a single DISCRETE block which is executed once every year (use the INTERVAL statement). Declare all population variables as INTEGER except for the total animal population P_{tot} which is to be exported to CTRL–C or MATLAB. (CTRL–C does not import INTEGER variables properly from ACSL. I haven't checked whether MATLAB handles this problem any better).

Repeat the above simulation with the modified assumption that the first year mortality rates are half as large as assumed above. Plot the total animal population over time on separate plots for the two cases.

Save the results from this simulation on a data file for later reuse (use the SAVE statement of CTRL–C or MATLAB).

[H10.4] Meta–Modeling

We wish to model the behavior of the above antelope population dynamics system through a meta–model. The assumed meta–model is of the type:

$$\dot{P} = a \cdot P \tag{H10.4a}$$

with the solution:

$$P(t) = \exp(at) \cdot P_0 \tag{H10.4b}$$

where P_0 is known while a is unknown. We want to find the best possible values of a for the two cases of hw(H10.3).

For this purpose, we want to apply linear regression analysis. We compute the logarithm of eq(H10.4b):

$$\log(P) - \log(P_0) = t \cdot a \tag{H10.4c}$$

We can read in the results from the hw(H10.3) using CTRL–C's (or MATLAB's) LOAD command. For each of the two cases, we have a vector of P values and a vector of t values. We can thus interpret eq(H10.4c) as an overdetermined set of linear equations with one unknown parameter, a, that can be solved in a least square's sense, i.e.

$$\log(P) - \log(P_0) - t \cdot a = residue \tag{H10.4d}$$

where a should be chosen such that the L_2 norm of the ***residue*** vector is minimized. This can be easily achieved in CTRL–C (MATLAB) using the notation:

```
[>  y = log(P) - log(P0) * ONES(P)
[>  a = t\y
```

Find the best a values for the two cases, compute the meta–model P trajectories by plugging these a values into eq(H10.4b), and plot together the "true" populations and the meta–model populations on separate plots for the two cases.

In one of the two cases, the population decreases to zero. Make sure to cut the trajectory for the regression analysis before this happens since your computer won't like the request to compute the logarithm of zero.

[H10.5] Lotka–Volterra Models

Whenever you can fit the predator of a two species Lotka–Volterra model to measurement data exhibiting a periodic steady–state, you can invariably also fit the prey of such a model. Prove this statement by applying the time reversal algorithm of Chapter 8 to the Lotka–Volterra model.

[H10.6] Logistic Equation

Write a program either in CTRL–C or in MATLAB which will reproduce Fig.10.9 to Fig.10.12. The concept is actually quite simple. First, you need to create your own scaling. Both languages provide you with such a feature. Next, you plot the two curves onto the same plot. Thereafter, you plot the zigzag path onto the plot. All you need to do is to store the x and y values of the path corners into two arrays in the correct sequence. Thereafter, PLOT will produce the path for you.

[H10.7] Logistic Bifurcation Map

Reproduce Fig.10.13. To create Fig.10.13, I wrote an ACSL program. In the initial segment, I set $a = 2.9$, I chose an initial value for x ($x = 0.5$), and I iterated the logistic equation 1000 times in a DO loop. I used an integer counter which I set to zero.

The DYNAMIC segment consisted of a single DISCRETE block. In this block, I iterated the logistic equation once, and I incremented the counter. I then tested whether the counter had reached a value of 50. If this was the case, I reset the counter to zero, and incremented a by 0.004. Thereafter, I iterated the logistic equation again 1000 times in a DO loop to obtain the new steady–state. The simulation terminated on $a \geq 4.0$. I avoided iterating the logistic equation when $a = 4.0$, since at this point, the logistic equation becomes unstable.

Finally, I exported the resulting trajectories of a and x into CTRL–C, and I plotted $x(a)$ using a point type plot.

[H10.8]* Gilpin's Bifurcation Map

Reproduce Fig.10.16. To create Fig.10.16, I wrote an ACSL program. In the initial segment, I set $k = 0.95$. I used an integer counter which I initialized to zero. I limited the step size to 1.0 (using MAXTERVAL), and I disabled the communication by setting arbitrarily $cint = 10^{12}$.

The DYNAMIC segment consisted of a DERIVATIVE block and a DISCRETE block. In the DERIVATIVE block, I solved the differential equations, and I scheduled a state–event to occur whenever the derivative of prey x_1 crosses through zero. The state–event triggers execution of the DISCRETE block. The DISCRETE block is similar to the one of hw(H10.7), except that I incremented the competition factor k by 0.001, and I did not iterate to determine the new steady–state. Also, I used the DISCRETE block to manually record the values of all variables using the CALL LOGD statement. It was necessary to place the TERMT statement inside the discrete block. Since we disabled communication, it won't be tested at the usual place.

Finally, I exported the resulting trajectories of k and x_1 into CTRL–C, and I plotted $x_1(k)$ using a point type plot.

Projects

[P10.1] Chaos in Gilpin's Model

We have seen that the chaotic range as a function of competition is very limited in Gilpin's model. Analyze the chaotic range as a function of other parameters. Design a method which will allow you to determine the bifurcation points in Gilpin's model accurately, and see whether you can find a relation between the bifurcation patterns of Gilpin's model as a function of the various model parameters.

Research

[R10.1] Evolutionary Games

Study the literature on evolutionary games, and try to identify common patterns among the strategies. What are the *innovators* in these models, and how do the *organizers* work? Try to design a differential equation model which periodically generates new differential equations of a modified type, and which periodically eliminates older differential equations that are not "fit" according to some chosen criterion. Analyze what are the conditions for self–organization to occur.

[R10.2] Chaotic Domains

Chaos is not *per se* a bad phenomenon. Without chaos, none of us would be alive. Our global ecology operates under chaotic steady–state conditions. Chaos ensures that all necessary minerals are constantly being recycled. A "continuous steady–state" of a variable (such as a mineral) would indicate that this variable has come to rest somewhere, and no longer participates in the recycling process.

In world ecology, these chaotic conditions are seemingly quite robust, i.e., chaos occurs for a wide range of parameter values. Contrary to this, we found that in the Gilpin equations, chaos existed only for a small set of parameter values. It is of considerable interest to understand how chaotic regions in the parameter space are related to the system structure. We know that, in a continuous–time model, no chaotic behavior can result for systems of orders one or two. The smallest system order that can lead to chaos is three. However, at order three, the chaotic regions are small and vulnerable. I suspect that with increasing system order, the chaotic regions in the parameter space will become more and more dominant, i.e., more and more robust.

Analyze arbitrary Lotka–Volterra type models, i.e., models of the type:

$$\dot{\mathbf{x}} = \left(diag(\mathbf{a}) + diag(\mathbf{x}) \cdot \mathbf{B}\right) \cdot \mathbf{x} \qquad (R10.2a)$$

Try to determine a general expression for the size of the chaotic regions in the parameter space as a function of the system order.