18
Unusual Dynamics of Nonlinear Systems

Nonlinear dynamics is one of the glorious successes of computational science. It has been explored by mathematicians, scientists and engineers, and usually with computers as an essential tool. (Even theologists have found motivation from the mathematical fact that simple systems can have very complicated behaviors.) The computed solutions have led to the discovery of new phenomena such as solitons, chaos, *and* fractals, *and even our brief treatment of the subject will permit you to uncover unusual properties on your own. In addition, because biological systems often have complex interactions, and may not be in thermodynamic equilibrium states, models of them are often nonlinear, with properties similar to other complex systems.*

In this chapter, we develop the logistics map as a model for how bug populations achieve dynamic equilibrium. It is an example of a very simple, but nonlinear, equation producing surprising complex behavior. In Chap. 19 we explore chaos for two continuous systems, the driven realistic pendulum [33, 34] and coupled predator–prey populations.

Problem: The population of insects and the patterns of weather do not appear to follow any simple laws.[1] At times they appear stable, at other times they vary periodically, and at other times they appear chaotic, only to settle down to something simple again. Your **problem** is to deduce if a simple and discrete law can produce such complicated behavior.

18.1 The Logistic Map (Model)

Imagine a bunch of insects reproducing, generation after generation. We start with N_0 bugs, then in the next generation we have to live with N_1 of them, and after i generations there are N_i bugs to bug us. We want to define a model for how N_n varies with the discrete generation number n. For guidance, we look to the radioactive decay simulation in Chap. 11 where the discrete decay law, $\Delta N/\Delta t = -\lambda N$, led to exponential-like decay. Likewise, if we reverse

[1] Excepting Oregon, where storms spend their weekends.

Computationyal Physics. Problem Solving with Computers (2nd edn).
Rubin H. Landau, Manuel José Páez, Cristian C. Bordeianu
Copyright © 2007 WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim
ISBN: 978-3-527-40626-5

of sign of λ we should get exponential-like growth, which is a good place to start our modeling. We assume that the bug breeding rate is proportional to the number of bugs:

$$\frac{\Delta N_i}{\Delta t} = \lambda\, N_i \tag{18.1}$$

We improve the model by incorporating the fact that bugs do live on love alone, they must also eat. But bugs, being not farmers, must compete for the available food supply, and this might limit their number to a maximum N_* (called the *carrying capacity*). Consequently, we modify the exponential growth model (18.1) by introducing a growth rate λ' that decreases as the population N_i approaches N_*:

$$\lambda' = \lambda(N_* - N_i) \qquad \Rightarrow \qquad \frac{\Delta N_i}{\Delta t} = \lambda'(N_* - N_i)N_i \tag{18.2}$$

We expect that when N_i is small compared to N_*, the population will grow exponentially, but as N_i approaches N_*, the growth rate will decrease, eventually becoming negative if N_i exceeds N_*.

Equation (18.2) is one form of the *logistic map*. It is usually written as a relation between the number of bugs in future and present generations:

$$N_{i+1} = N_i + \lambda'\,\Delta t(N_* - N_i)N_i \tag{18.3}$$

$$= N_i\left(1 + \lambda'\,\Delta t N_*\right)\left[1 - \frac{\lambda'\,\Delta t}{1 + \lambda'\,\Delta t N_*}N_i\right] \tag{18.4}$$

The map looks simple when expressed in terms of natural variables:

$$x_{i+1} = \mu x_i(1 - x_i) \tag{18.5}$$

$$\mu \overset{\text{def}}{=} 1 + \lambda'\,\Delta t N_* \qquad x_i \overset{\text{def}}{=} \frac{\lambda'\,\Delta t}{\mu}N_i \simeq \frac{N_i}{N_*} \tag{18.6}$$

where μ is a dimensionless growth parameter and x_i is a dimensionless population variable. Observe that the *growth rate* μ equals 1 when the breeding rate $\lambda' = 0$, and is otherwise expected to be larger than 1. If the number of bugs born per generation $\lambda'\,\Delta t$ is large, then $\mu \approx \lambda'\,\Delta t N_*$ and $x_i \approx N_i/N_*$. That is, x_i is essentially the fraction of the maximum population N_*. Consequently, we consider x values in the range $0 \leq x_i \leq 1$, where $x = 0$ corresponds to no bugs, and $x = 1$ to the maximum population. Note that there is clearly a linear and quadratic dependence of the RHS of (18.5) on x_i. In general, a map uses a function $f(x)$ to map one number in a sequence to another,

$$x_{i+1} = f(x_i) \tag{18.7}$$

For the logistic map, $f(x) = \mu x(1 - x)$, with the quadratic dependence of f on x makes this a nonlinear map, while the dependence on only the one variable x_i makes it a *one-dimensional* map.

18.2 Properties of Nonlinear Maps (Theory)

Rather than do some fancy mathematical analysis to determine properties of the logistic map [35], we prefer to have you study it directly on the computer by plotting x_i *versus* generation number i. Some typical behaviors are shown in Fig. 18.1. In A we see equilibration into a single population; in B we see oscillation between two population levels; in C we see oscillating among four levels, and in D we see a chaotic system. The initial population x_0 is known as the *seed*, and as long as it is not equal to zero, its exact value generally has little effect on the population dynamics (similar to what we found when generating pseudo-random numbers). In contrast, the dynamics are unusually sensitive to the value of the growth parameter μ. For those values of μ at which the dynamics are complex, there may be extreme sensitivity to the initial condition, x_0, as well as on the exact value of μ.

18.2.1 Fixed Points

An important property of the map (18.5) is the possibility of the sequence x_i reaching a *fixed* point at which x_i remains or fluctuates about. We denote such fixed points as x_*. At a *one-cycle* fixed point, there is no change in the population from generation i to generation $i + 1$, and so it must satisfy

$$x_{i+1} = x_i = x_* \tag{18.8}$$

Using the logistic map (18.5) to relate x_{i+1} to x_i, yields the algebraic equation

$$\mu x_*(1 - x_*) = x_* \qquad \Rightarrow \qquad x_* = 0 \text{ or } x_* = \frac{\mu - 1}{\mu} \tag{18.9}$$

The nonzero fixed point $x_* = (\mu - 1)/\mu$ corresponds to a stable population with a balance between birth and death that is reached regardless of the initial population (Fig. 18.1A). In contrast, the $x_* = 0$ point is unstable and the population remains static only as long as no bugs exist; if even a few bugs are introduced, exponential growth occurs.

Further analysis based on the rate of change of x tells us that the stability of a population is determined by the magnitude of the derivative of the mapping

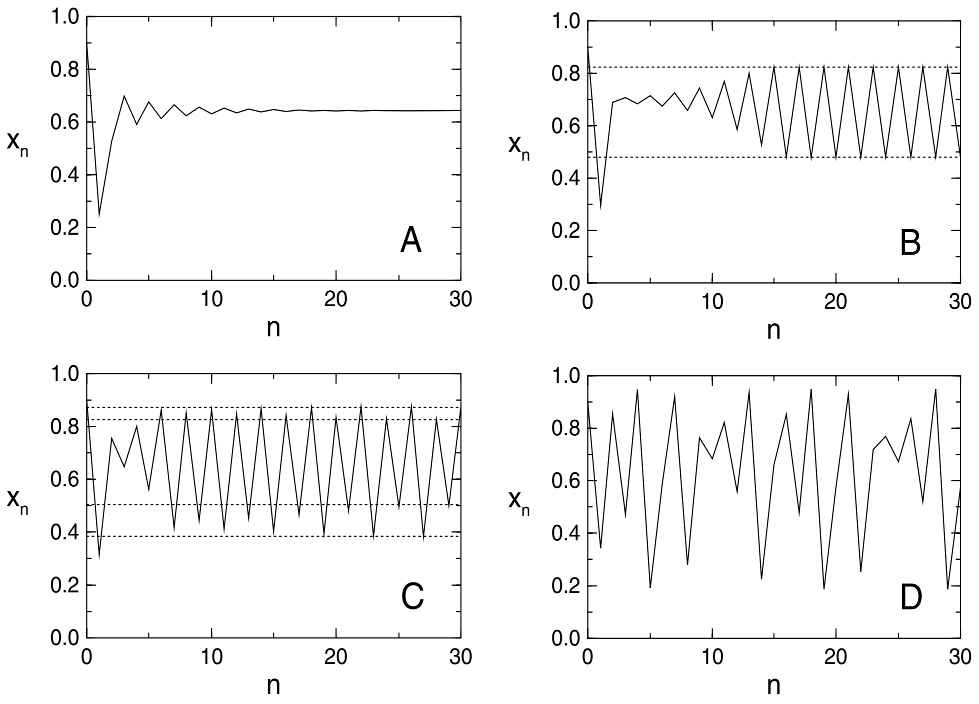

Fig. 18.1 The insect population x_n versus generation number n for various growth rates. (A) $\mu = 2.8$, a period-one cycle. If the fixed point is $x_n = 0$, the system becomes extinct. (B) $\mu = 3.3$, a period-two cycle. (C) $\mu = 3.5$, a period-four cycle. (D) $\mu = 3.8$, a chaotic regime. (If $\mu < 1$, the population goes extinct.

function $f(x_i)$ at the fixed point [35]:

$$\left|\frac{df}{dx}\right|_{x_*} < 1 \quad \text{(stable)} \tag{18.10}$$

For the one-cycle of the logistic map (18.5), we have

$$\left.\frac{df}{dx}\right|_{x_*} = \mu - 2\mu x_* = \begin{cases} \mu & \text{stable at } x_* = 0 \ \text{ if } \mu < 1 \\ 2 - \mu & \text{stable at } x_* = \frac{\mu - 1}{\mu} \ \text{ if } \mu < 3 \end{cases} \tag{18.11}$$

18.2.2 Period Doubling, Attractors

Equation (18.11) tells us that while the equation for fixed points (18.9) may be satisfied for all values of μ, the populations will not be stable if $\mu > 3$. For $\mu > 3$, the system's long-term population *bifurcates* into two populations (a *two-cycle*), an effect known as *period doubling* (Fig. 18.1B). Because the system now acts as if it were attracted to two populations, these populations are called *attractors* or *cycle points*. We can easily predict the x values for these two-cycle attractors by requiring that generation $i+2$ have the same population as generation i:

$$x_i = x_{i+2} = \mu x_{i+1}(1 - x_{i+1}) \tag{18.12}$$

$$\Rightarrow \quad x_* = \frac{1 + \mu \pm \sqrt{\mu^2 - 2\mu - 3}}{2\mu} \tag{18.13}$$

We see that as long as $\mu > 3$, the square root produces a real number and thus physical solutions exist (complex or negative x_* values are unphysical).

We leave it for your computer explorations to discover how the system continues to double periods as for μ continues to increase. In all cases the pattern is the same: one of the populations bifurcates into two.

18.3 Explicit Mapping Implementation

Program up the logistic map to produce a sequence of population values x_i as a function of the generation number i. These are called *map orbits*. The assessment consists of confirmation of Feigenbaum's observations [36] of the different behavior patterns shown in Fig. 18.1. These occur for growth parameter $\mu = (0.4, 2.4, 3.2, 3.6, 3.8304)$ and seed population $x_0 = 0.75$. Identify the following on your graphs of x_i versus i:

1. **Transients:** Irregular behaviors before reaching a steady state. These transients differ for different seeds.
2. **Asymptotes:** In some cases the steady state is reached after only 20 generations, while for larger μ values, hundreds of generations may be needed. These steady-state populations are independent of the seed.
3. **Extinction:** If the growth rate is too low, $\mu \leq 1$, the population dies off.
4. **Stable states:** The stable, single-population states attained for $\mu < 3$ should agree with the prediction (18.9).
5. **Multiple cycles:** Examine the map orbits (x_i *versus* i) for a growth parameter μ increasing continuously through 3. Observe how the system continues to double periods as μ increases. To illustrate, in Fig. 18.1C with $\mu = 3.5$, we notice a steady state in which the population alternates among four attractors (a *four-cycle*).
6. **Intermittency:** Observe simulations for $3.8264 < \mu < 3.8304$. Here the system should appear stable for a finite number of generations, then jump all around, only to become stable again.

7. **Chaos:** We define *chaos* as the deterministic behavior of a system displaying no discernible regularity. This may seem contradictory; if a system is deterministic, it must have step-to-step correlations (which, when added up, means long-range correlations); but if it is chaotic, the complexity of the behavior may hide the simplicity within. In an operational sense, a chaotic system is one with an extremely high sensitivity to parameters or initial conditions. This sensitivity to even minuscule changes is so high that it is impossible to predict the long-range behavior unless the parameters are known to infinite precision (a physical impossibility).

   The system's behavior in the chaotic region is critically dependent on the exact value of μ and x_0. Systems may start out with nearly identical values for μ and x_0, but end up quite different. In some cases the complicated behaviors of nonlinear systems will be *chaotic*, but unless you have a bug in your program, they will not be random.[2]

   (a) Compare the long-term behaviors of starting with the two essentially identical seeds, $x_0 = 0.75$, and $x_0' = 0.75(1+\epsilon)$, where $\epsilon \simeq 2 \times 10^{-14}$.

   (b) Repeat the simulation with $x_0 = 0.75$, and two essentially identical survival parameters, $\mu = 4.0$, and $\mu' = 4.0(1-\epsilon)$. Both simulations should start off the same, but eventually diverge.

18.4 Bifurcation Diagram (Assessment)

Computing and watching the population change with generation number gives a good idea of the basic dynamics, at least until they get too complicated to discern patterns. In particular, as the number of bifurcations keeps increasing and the system becomes chaotic, it is hard for our minds to see a simple underlying structure within the complicated behavior. One way to visualize what is going on is to concentrate on the attractors, that is, those populations that appear to attract the solutions and to which the solutions continuously return. A plot of these attractors (long-term iterates) of the logistic map as a function of the growth parameter μ is an elegant way to summarize the results of extensive computer simulations.

A *bifurcation diagram* for the logistics map is given in Fig. 18.2, while one for a different map is given in Fig. 18.3. For each value of μ, hundreds of iterations are made to make sure that all transients essentially die out, and

[2] You may recall from Chap. 11 that a random sequence of events does not even have step-by-step correlations.

then the values (μ, x_*) are written to a file for hundreds of iterations after that. If the system falls into an n cycle for this μ value, then there should predominantly be n different values written to the file. Next, the value of the initial populations x_0 is changed slightly, and the entire procedure repeated to ensure that no fixed points are missed. When done, your program will have stepped through all values of growth parameter μ, and for each value of μ will have stepped through all values of initial population x_0.

18.4.1 Bifurcation Diagram Implementation

The last part of this problem is to reproduce Fig. 18.2 at various levels of detail.[3] While the best way to make a visualization of this sort would be with visualization software that permits you to vary the intensity of each individual point on the screen, we simply plot individual points and have the density in each region determined by the number of points plotted there. When thinking about plotting many individual points to draw a figure, it is important to keep in mind that your screen resolution is ~100 dots per inch and your laser printer resolution may be 300 dots per inch. This means that if you plotted

[3] You can listen to a sonification of this diagram on the CD.

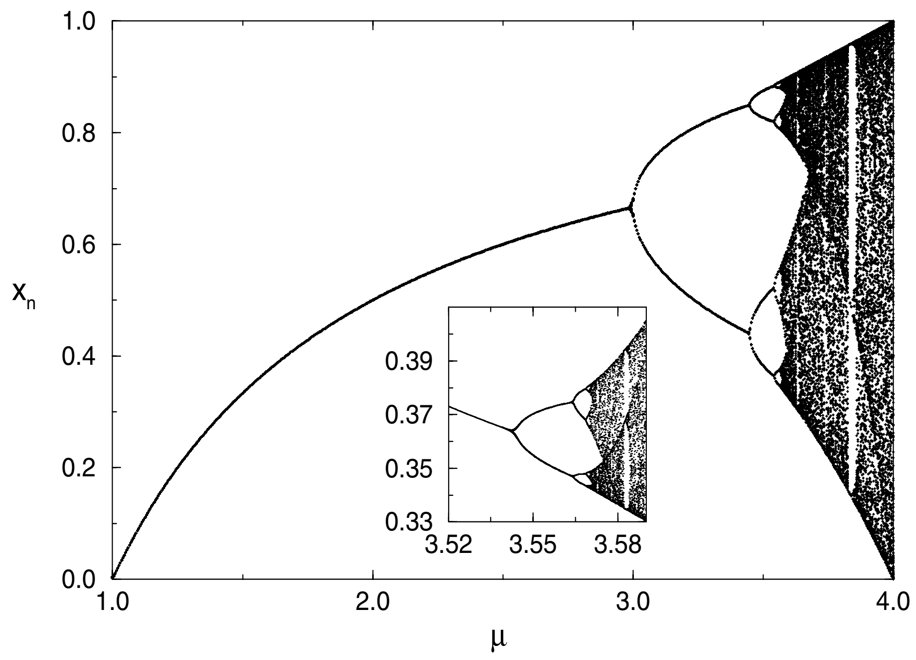

Fig. 18.2 The bifurcation plot, attractor populations versus growth rate, for the logistic map.

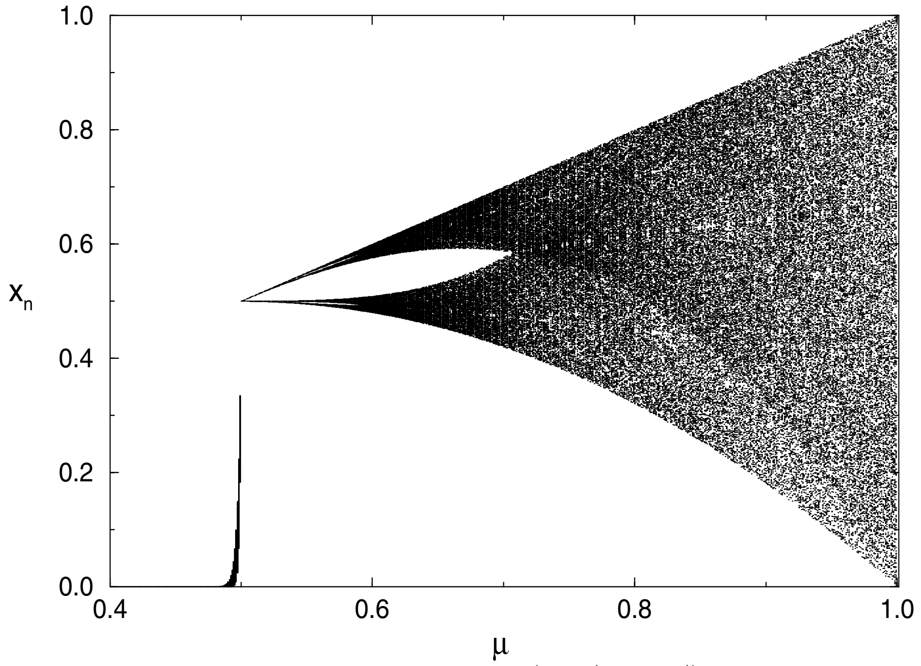

Fig. 18.3 A bifurcation plot for the tent map, $x_{i+1} = \mu(1 - 2\,|x_i - 1/2|)$.

a point at each pixel, you would be plotting $\sim$3000 $\times$ 3000 $\simeq$ 10 million elements. *Beware,* this can require some time and may choke a printer. In any case, printing at finer resolution is a waste of time.

18.4.2 Visualization Algorithm: Binning

1. Break up the range $1 \le \mu \le 4$ into 1,000 steps and loop through them. These are the "bins" into which we will place the x_* values.

2. In order not to miss any structures in your bifurcation diagram, loop through a range of initial x_0 values as well.

3. Wait at least 200 generations for the transients to die out, and then print out the next several hundred values of (μ, x_*) to a file.

4. Print out your x_* values to no more than three or four decimal places. You will not be able to resolve more places than this on your plot, and this restriction will keep your output files smaller by permitting you to remove duplicates. It is hard to control the number of decimal places in output with Java's standard print commands (although `printf` does permit control). A simple approach is to multiply the x_i values by 1000

and then throw away the part to the right of the decimal point. Because $0 \leq x_n \leq 1$, this means that $0 \leq 100 * x_n \leq 1000$, and you can throw away the decimal part by casting the resulting numbers as integers:

```
Ix[i]= (int)(1000*x[i])                    // Convert to 0< ints < 1000
```

You may then divide by 1000 if you want floating point numbers.

5. You also need to remove duplicate values of (x, μ) from your file (they just take up space and plot on top of each other). You can do that in Unix/Linus with the *sort -u* command.
6. Plot up your file of x_* versus μ. Use small symbols for the points and do not connect them.
7. Enlarge sections of your plot and notice that a similar bifurcation diagram tends to be contained within each magnified portion (this is called *self-similarity*).
8. Look over the series of bifurcations occurring at

   $$\mu_k \simeq 3,\ 3.449,\ 3.544,\ 3.5644,\ 3.5688,\ 3.569692,\ 3.56989, \ldots. \qquad (18.14)$$

   The end of this series is a region of chaotic behavior.
9. Inspect the way this and other sequences begin and then end in chaos. The changes sometimes occur quickly and so you may have to make plots over a very small range of μ values to see the structures.
10. Close examination of Fig. 18.2 shows regions where, with a slight increase in μ, a very large number of populations suddenly change to very few populations. Whereas these may appear to be artifacts of the video display, this is a real effect and these regions are called *windows*. Check that at $\mu = 3.828427$, chaos turns into a three-cycle population.

18.5 Random Numbers via Logistic Map (Exploration)

There are claims that the logistic map in the chaotic region ($\mu \geq 4$)

$$x_{i+1} \simeq 4x_i(1 - x_i) \qquad (18.15)$$

can be used to generate random numbers [37]. Although successive x_i's are correlated, if the population for every $\sim$ 6th generation is examined, the correlations die out and effectively random numbers result. To make the sequence more uniform, a trigonometric transformation is used:

$$y_i = \frac{1}{\pi} \cos^{-1}(1 - 2x_i) \qquad (18.16)$$

Use the random-number tests discussed in Chap. 11 to test this claim.

18.6 Feigenbaum Constants (Exploration)

Feigenbaum discovered that the sequence of μ_k values (18.14) at which bifurcations occur follow a regular pattern [36]. Specifically, it converges geometrically when expressed in terms of the distance between bifurcations δ:

$$\mu_k \quad \to \quad \mu_\infty - \frac{c}{\delta^k} \qquad\qquad \delta = \lim_{k\to\infty} \frac{\mu_k - \mu_{k-1}}{\mu_{k+1} - \mu_k} \tag{18.17}$$

Use your sequence of μ_k values to determine the constants in (18.17) and compare to those found by Feigenbaum:

$$\mu_\infty \simeq 3.56995 \qquad c \simeq 2.637 \qquad \delta \simeq 4.6692 \tag{18.18}$$

Amazingly, the value of the *Feigenbaum constant* δ is universal for all second-order maps.

18.7 Other Maps (Exploration)

Bifurcations and chaos are characteristic properties of nonlinear systems. Yet systems can be nonlinear in a number of ways. Table 18.1 lists four maps that generate x_i sequences containing bifurcations. The tent map (Fig. 18.3) derives its nonlinear dependence from the absolute value operator, while the logistics map is a subclass of the ecology map. Explore the properties of these other maps, and note the similarities and differences.

Tab. 18.1 Several nonlinear maps to explore.

Name	$f(x)$	Name	$f(x)$
Logistic	$\mu x(1-x)$	Tent	$\mu(1-2\,\lvert x-1/2\rvert)$
Ecology	$xe^{\mu(1-x)}$	Quartic	$\mu[1-(2x-1)^4]$