10 Nonlinear Dynamics in Simple Maps

Chaos is the score upon which reality is written.
— Henry Miller

... it may happen that small differences in the initial conditions produce very great ones in the final phenomena.
— Henri Poincaré

Prediction is difficult, especially of the future.
— Mark Twain (also attributed to Niels Bohr)

In the previous book part we saw how natural physical structures could be described in terms of fractal geometry. In this book part we will be concerned with the related topic of *chaos* in *nonlinear dynamical systems*. A dynamical system can be loosely defined as anything that has motion, such as swinging pendulums, bouncing balls, robot arms, reactions in a chemical process, water flowing in a stream, or an airplane in flight. For each of these examples there are two important aspects that must be considered. First, we need to determine what it is about a dynamical system that changes over time. In the case of the pendulum, both position and velocity vary over time, so we would be concerned with the "motion" of both of these states of the pendulum. A less obvious example is found in a chemical reaction, where the "motion" can be found in the ratio of reactants to reagents, or perhaps in some physical aspect of the chemicals, such as temperature or viscosity. The second aspect of a dynamical system that we must be concerned with is in the collection of rules that determine how a dynamical system changes over time. Usually, scientists have a mathematical model of how a real dynamical system works. The model will typically have equations that may be parameterized by time and the previous states of the system. Sometimes these equations can be used to get an estimate of what the future state of a dynamical system will be. In

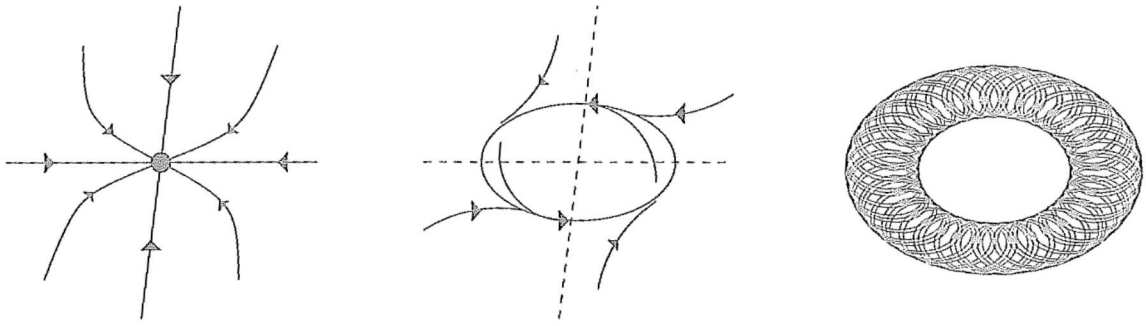


Figure 10.1 Different types of motion: (a) fixed point, (b) limit cycle (c) quasiperiodic

this spirit, let's agree that the "motion" of a dynamical system is dependent on how the state of the system changes over time. Moreover, there must exist a set of rules that governs how a dynamical system in some state evolves to another state. We may not know what the rules are, but if a dynamical system is deterministic, a set of rules for the time evolution of the system exists independently of our knowledge of it.

There are many different types of motion that can be exhibited by a dynamical system. The simplest is *fixed point* behavior, which can be seen in a pendulum when friction and gravity bring the system to a halt. Most fixed points can be likened to a ball placed on top of a hill, which rolls downward until at some point the ball sits on a flat spot and has no momentum to carry it further.

The next simplest type of motion is known as a *limit cycle* or *periodic* motion, which involves movement that repeats itself over and over. A lone planet orbiting a star in an elliptical orbit is an example of a limit cycle. Some limit cycles are more complicated than others. For example, a child on a swing drives the motion of the swing by periodically rocking in beat with the natural frequency of the swing, much like an idealized pendulum. Now, if the child rhythmically swings one leg at a higher frequency, the motion of that leg will cause the motion of his or her body to subtly wobble back and forth. If it is timed accurately, the child may be able to coerce the motion into behavior that is more complicated than that of the planet, with his or her body moving back and forth along the main axis of the swing, and perpendicularly left to right in step with the leg.

A slightly more complicated form of motion is found in *quasiperiodic* systems, which are similar to periodic systems except that they never quite repeat themselves. For example, the moon orbits Earth, which orbits the sun, which, in turn, orbits the galactic center, and so on. In order for the combined motion of the moon and Earth to be truly periodic, they must at some future point return to some previously occupied state. But in order for that to happen, all of the individual motions must resonate, which means that there must exist a length of time that will evenly divide

all of the frequencies. Figure 10.1c shows quasiperiodic motion as consisting of two independent circular motions that fail to repeat due to a lack of resonance.

Up until the last thirty years or so, almost every scientist believed that everything in the universe fell into either fixed point, periodic, or quasiperiodic behavior. The belief in a clockwork universe, as exemplified by the mathematician Pierre-Simon de Laplace[1], held that, in principle, if one had an accurate measure of the state of the universe and knew all of the laws that govern the motion of everything, then one would be able to predict the future with near perfect accuracy. We now know that this is not true, since science was mistaken in its assumption that everything is either a fixed point or a limit cycle. Chaotic systems are not just exceptions to the norm but are, in fact, more prevalent than anyone could imagine. Chaos is everywhere: in the turbulence of water and air, in the wobble of planets as they follow complicated orbits, in global weather patterns, in the human brain's electrochemical activity, and even in the motion of a child on a swing. In all of these cases the complicated motion produced by chaos prohibits predicting the future in the long term. On the other hand, phenomena that were once thought to be purely random are now known to be chaotic. The good news in this case is that chaotic systems admit prediction in the short term.

Chaos is related to the other topics of this book in many ways. The pathological nature of the incomputable functions from Part I is very similar to the unpredictability of chaotic systems. The motion of chaotic systems can be described by fractal geometry. There is also a hypothesis known as "computation on the edge of chaos" that will be relevant in the next book part when we study complex systems.

In this chapter we will be primarily concerned with getting an intuitive feel for how chaos works in a simple, discrete time, iterative system. The nice thing about the examples that we will be looking at is that the richness, diversity, and beauty of chaos are exhibited by a deceptively simple dynamical system.

10.1 The Logistic Map

The logistic map (also known as the "quadratic map" or the "Feigenbaum map") is a simple population growth model that is defined by the iterative equation[2]

$$x_{t+1} = 4rx_t(1 - x_t),$$

where r is a parameter that can be set to reflect the reproduction rate of the population. The legal values of x_t (as well as x_{t+1}) range between 0 and 1 inclusively.

[1]Laplace claimed that "given for one instant an intelligence which could comprehend all the forces by which nature is animated and the respective positions of the beings which compose it ... nothing would be uncertain, and the future as the past would be present to its eyes."

[2]In many scientific writings the logistic map is given as $x_{t+1} = rx_t(1 - x_t)$, with r ranging from 0 to 4. I have deliberately added the 4 in my presentation so that r can range from 0 to 1.

If x_t equals 0, then we can interpret the population as being extinct. If x_t equals 1, then the system is overpopulated (with imminent extinction at the next time step). All other values between 0 and 1 represent intermediate population levels.

The value of r is also allowed to vary between 0 and 1. If r is 0, then nothing reproduces in the population. If r is equal to 1, then the members of the population are reproducing at the maximum rate. Over the course of this section, we will look at several different values of r. Since the logistic map is so simple, you may wish to use a calculator while reading this chapter, so as to simulate population growth yourself.

To use the logistic map, we must choose a constant value for r and an initial population, x_0. From x_0 we can compute x_1, then x_2, and so on. Intuitively, we can assign a useful interpretation to the individual terms of the dynamical system. We can think of $4rx_t$ as being positive feedback in the sense that as x_t grows in size, so does the value of $4rx_t$, which is akin to saying that a population size is partially determined by the product of the previous size and the rate at which members reproduce. The $(1 - x_t)$ portion of the equation can be thought of as negative feedback, since increasing x_t will decrease $(1 - x_t)$; thus, $(1 - x_t)$ can be thought of as population decline due to overpopulation and scarce resources.

What we would like to know is, for some value of r, what happens to the long-term behavior of the system as t goes to infinity. Let's consider some special cases. Suppose that r is less than or equal to $\frac{1}{4}$. In this case the time evolution is described by $\alpha x_t(1-x_t)$, with α, x_t, and $(1-x_t)$ all between 0 and 1. Since the population at the next time step is the product of three numbers between 0 and 1, the population at the next time step must always be smaller than what it is at the current time step. Thus, no matter what initial value we choose for x_t, if $r \leq \frac{1}{4}$, then the population is doomed to extinction.

Suppose that r is greater than $\frac{1}{4}$ but less than $\frac{3}{4}$. In this case the long-term behavior of the logistic map is to fall into a fixed point. Figure 10.2 illustrates this graphically. The graph on the left shows the values of x_t plotted over time for 100 steps. The graph on the right shows the *state space* of the dynamical system (also referred to as the *phase space*) that plots x_t versus x_{t+1} to illustrate how the next iterate depends on the current value. The parabola in the graph is a plot of $x_{t+1} = 4rx_t(1-x_t)$ with $r = \frac{7}{10}$. At any time step, we can geometrically determine the value of the next iterate by taking the current value of x_t, drawing a line at that value from the identity line to where it intersects the parabola, and then making a 90 degree turn toward the identity line again. In this way, the time evolution of the logistic map can be seen to be attracted to the fixed point. Moreover, it doesn't matter what initial value we pick for x_0, since any choice between 0 and 1 will cause the system to converge into the same fixed point.

If we set r greater than $\frac{7}{10}$ but less than $\frac{3}{4}$, then the value of the attracting fixed point will increase as we make r larger. Intuitively, this makes sense since one would expect a stable population to increase in size if the reproduction rate increases. All

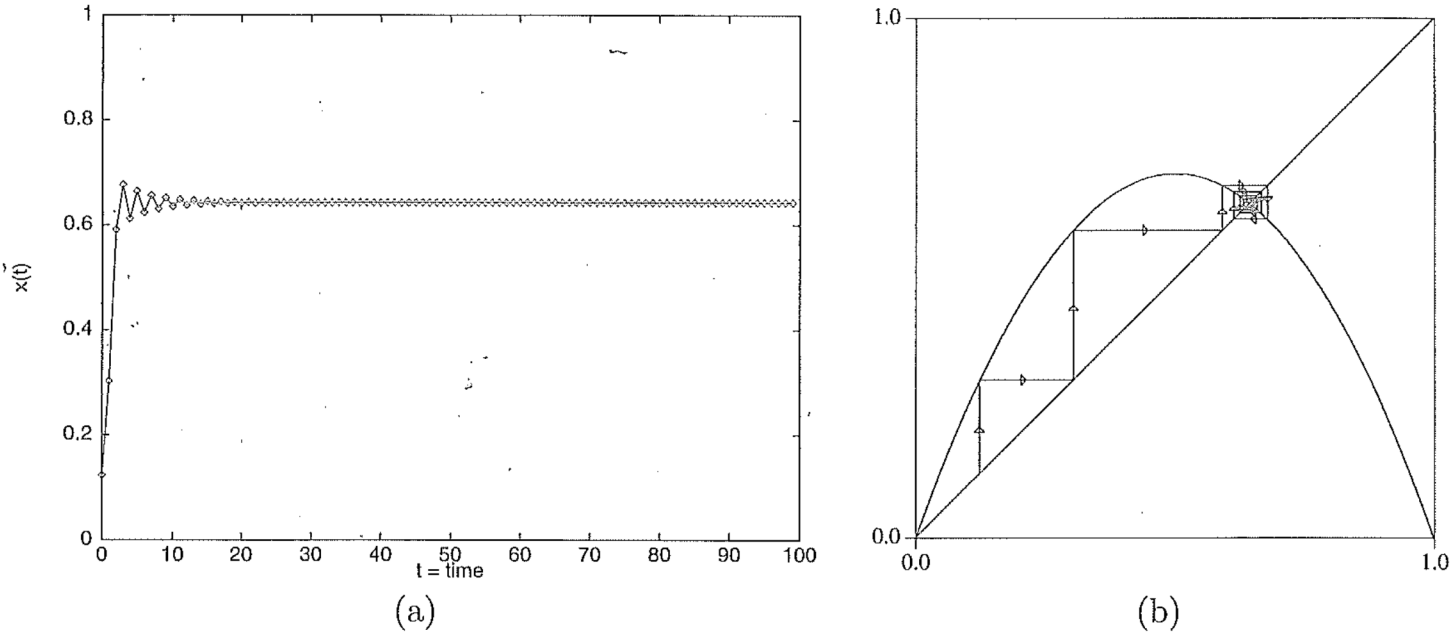


Figure 10.2 Logistic map with $r = \frac{7}{10}$: (a) The time series quickly stabilizes to a fixed point. (b) The state space of the same system shows how subsequent steps of the system get pulled into the fixed point.

fixed points, regardless of the value of r, can be found by seeing where the state space plot of x_t versus x_{t+1} intersects the identity line. With $\frac{1}{4} < r \leq \frac{3}{4}$ the parabola will always intersect the identity line at two points: the *stable fixed point*, which attracts all nonzero points, and 0 which is an *unstable fixed point*. The first fixed point is stable because if you randomly perturb the system away from the fixed point, the system will quickly converge back to the attracting fixed point again. Zero is an unstable fixed point because any perturbation will cause the system to leave that infinitesimal region forever. If you like, you can think of a stable fixed point as being similar to a ball at the bottom of a crater, valley, or depression. If you softly kick the ball, it will move just a bit, then stop. If your depression is shaped like a perfect bowl, then the ball will come to rest at the same location it started from. An unstable fixed point is like a ball perfectly balanced on the peak of a mountain. The slightest nudge will cause the ball to move away from the fixed point, never to return. We can squeeze one more useful notion out of this metaphor. Considering the stable fixed point once again, notice that there is an area defined by the perimeter of the depression. If you drop the ball anywhere within the perimeter, the ball will fall to the bottom of the depression. If the ball is dropped outside of the perimeter, the ball will go elsewhere. This area within the perimeter is sometimes referred to as the *basin of attraction*. An unstable fixed point has a basin of attraction that has 0 volume and area.

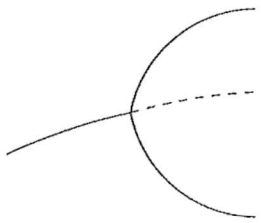


Figure 10.3 A single bifurcation

Something interesting happens when r is set just above $\frac{3}{4}$. For example, when r is equal to $\frac{8}{10}$, as long as the choice for x_0 is neither 0 nor $\frac{22}{32}$, the system will never converge to any fixed point. Instead, the population will settle into a period-2 limit cycle, which means that it will oscillate between two values. Zero will still be an unstable fixed point, but the fixed point near $\frac{22}{32}$ will now also be unstable. Figure 10.3 illustrates what's happening much better.

The leftmost portion of the drawing illustrates how the location of the fixed point gradually increases as we increase the value of r; however, at a critical value, the path will split in two. The two solid lines on the right illustrate how the location of the period-2 limit cycle changes as the value of r is increased. The dashed line shows how the once stable fixed point continues to exist, but in a form that is unstable. Thus, solid lines denote stable paths, and dashed lines denote unstable paths. Figure 10.4 illustrates the period-2 limit cycle in a form that is similar to Figure 10.2. Notice that the state space diagram now clearly shows how the state of the system oscillates between two values.

10.2 Stability and Instability

For one-dimensional maps like the logistic map, there is a very simple method to determine if a fixed point or limit cycle is stable or unstable. The idea behind the technique is to examine the local behavior of the map in the vicinity of a fixed point or limit cycle. For notational purposes, let's refer to the mapping function as $f(x)$, that is, $f(x) = 4rx(1-x)$ for the logistic map. The first derivative of $f(x)$, which is equal to $f'(x) = 4r(1-2x)$, tells us how steeply sloped the function is in the vicinity of x. Now, suppose there exists a fixed point, which we will call x_F, and that we would like to know whether the fixed point is stable or unstable. If we look at the derivative of $f(x)$ evaluated at x_F, then the following possibilities may characterize the behavior of $f(x)$ at x_F:

$$\begin{array}{rcll} |f'(x_F)| & < & 1 & \text{attracting and stable} \\ f'(x_F) & = & 0 & \text{super-stable} \\ |f'(x_F)| & > & 1 & \text{repelling and unstable} \\ |f'(x_F)| & = & 1 & \text{neutral} \end{array}$$

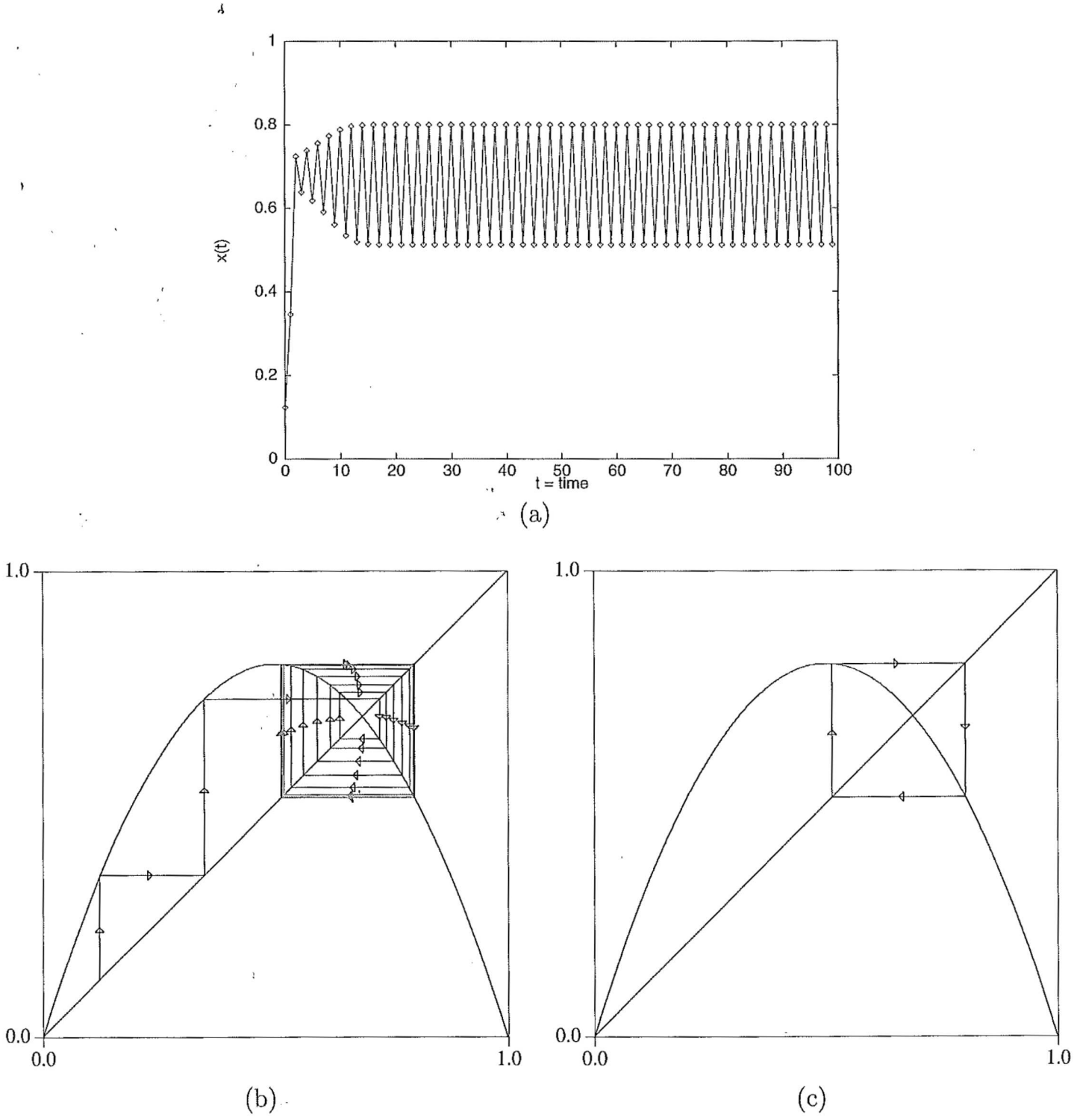


Figure 10.4 Logistic map with $r = \frac{8}{10}$: (a) The time series quickly stabilizes to a period-2 limit cycle. (b) The state space of the same system shows how subsequent steps of the system get pulled into the limit cycle. (c) The state space of the same system but with only the converged values for x_t plotted, so as to clearly show the limit cycle's location.

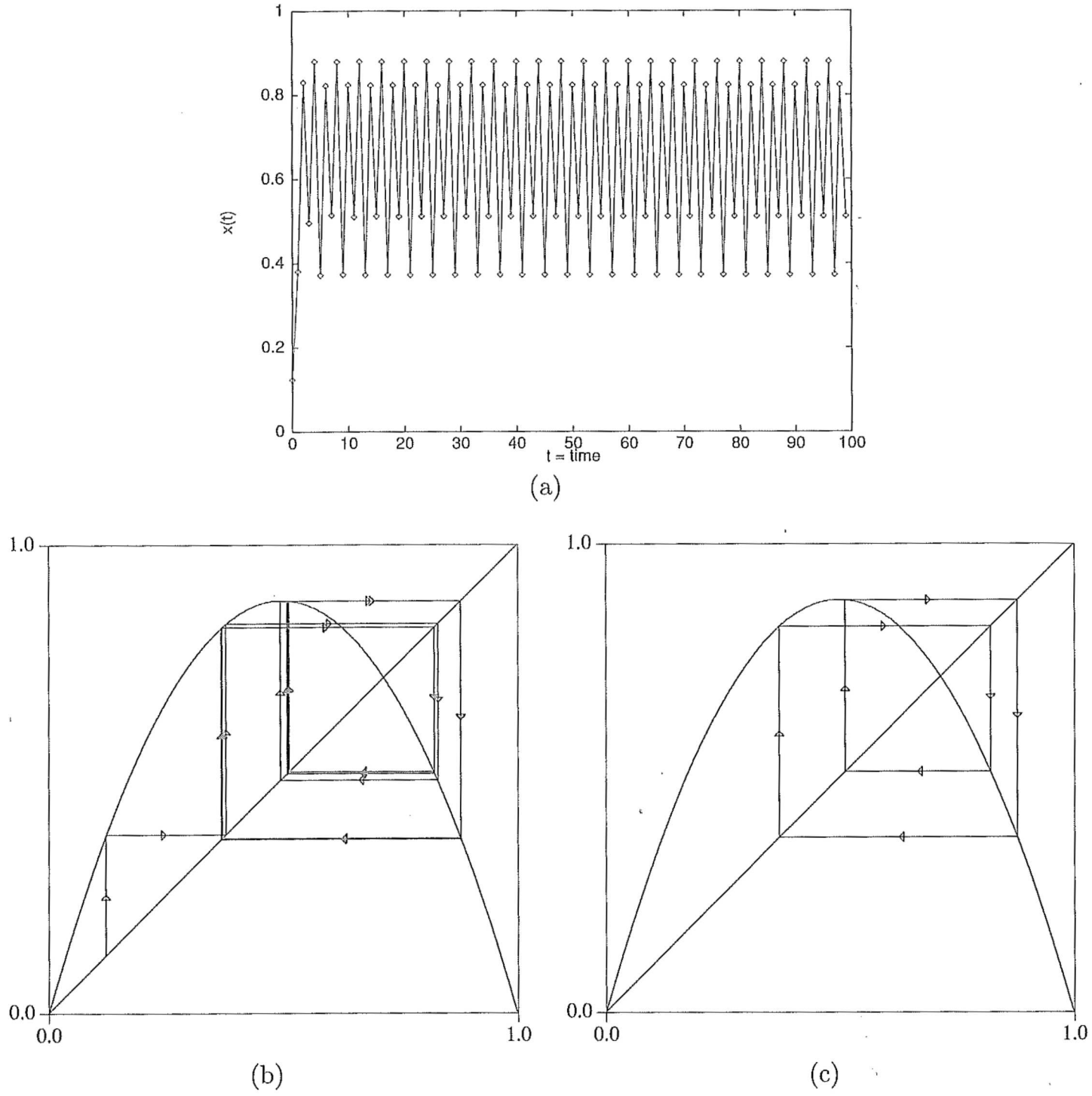
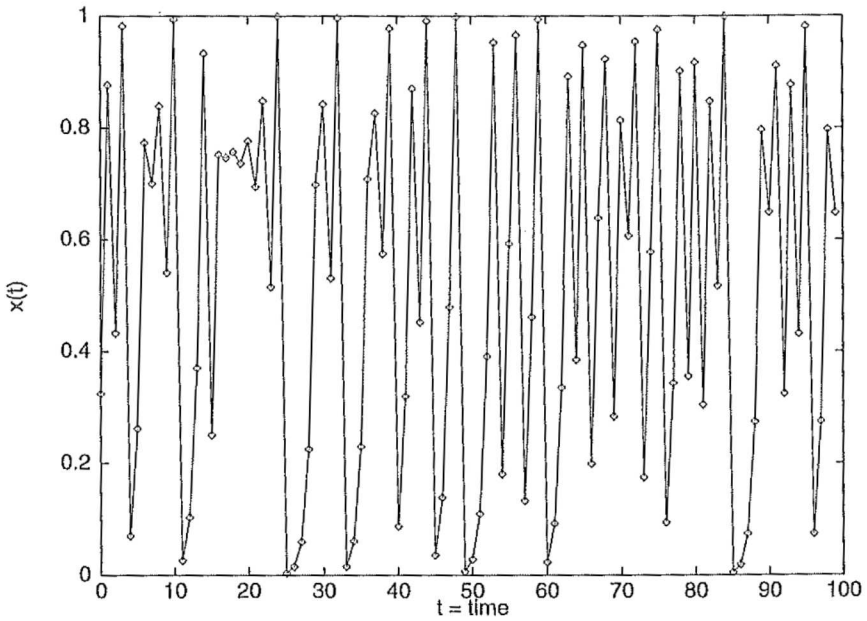
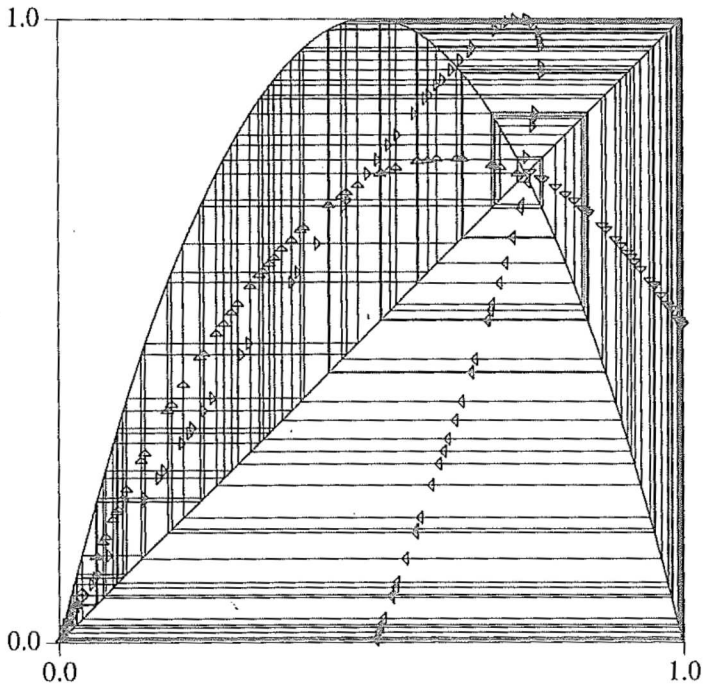


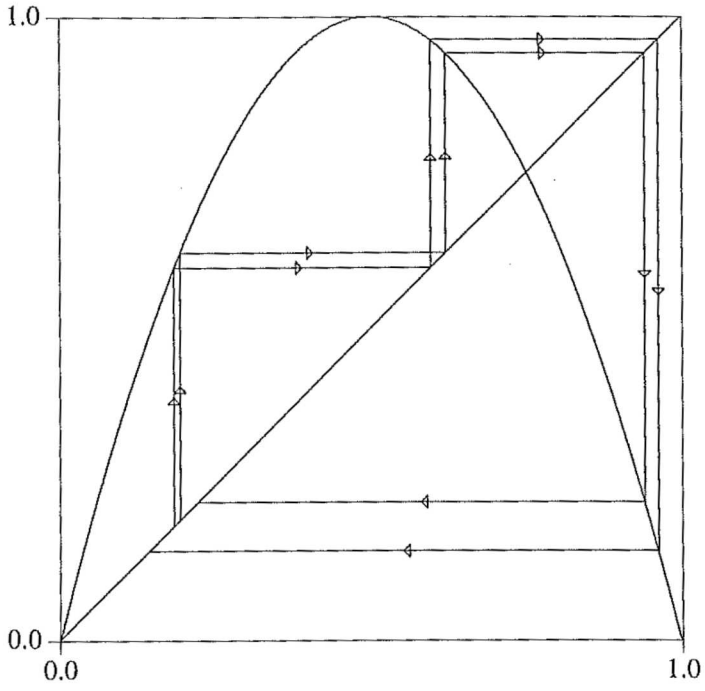
Figure 10.5 Logistic map with $r = \frac{88}{100}$: (a) The time series quickly stabilizes to a period-4 limit cycle. (b) The state space of the same system. (c) The state space of the same system but with only the converged values for x_t plotted.



(a)



(b)



(c)

Figure 10.6 Logistic map with $r = 1$: (a) The time series is chaotic and has the appearance of noise. (b) The state space of the same system, which illustrates how the system's trajectory visits every local region. (c) The state space of the same system with only four steps plotted, so as to show how small differences turn into larger differences.

As a specific case, consider $r = \frac{3}{4}$ and $x_F = \frac{2}{3}$, which would give us $|f'(x_F)| = |3(1 - \frac{4}{3})| = 1$, which further means that the system is neutral at that x_F for the given value of r. Let's see what happens when we change r by just a very small amount. First, let's make r a little bit smaller: $r = \frac{3}{4} - \epsilon$, where ϵ represents a very small positive number. The new location of the fixed point can be found by solving for x_F in the equation $x_F = 4rx_F(1 - x_F)$. If we take this new value and plug it into $|f'(x_F)|$, we find that the result will be just less than 1, which means that the fixed point is now stable. Similarly, if we set r to $\frac{3}{4} + \epsilon$, the new fixed point can be computed and plugged into $|f'(x_F)|$. The resulting value (if a suitably small enough ϵ is used) will be just larger than 1, which further implies that the fixed point is now unstable.

The analysis for the stability of a limit cycle is nearly identical except for the fact that a limit cycle will oscillate between, say, m points in an orbit: x_{F_1}, x_{F_2}, $\cdots$, x_{F_m}. Taking the absolute value of the product of $f'(x)$ evaluated at each of these points gives us

$$\left| \prod_{i=1}^{m} f'(x_{F_i}) \right| = |f'(x_{F_1}) \times f'(x_{F_2}) \times \cdots \times f'(x_{F_m})|,$$

which has the same stability properties listed above for fixed points.

10.3 Bifurcations and Universality

Recall that the junction point where the system moves from a fixed point to a period-2 limit cycle occurs at exactly $\frac{3}{4}$. This period doubling is known as a *bifurcation*. Let's see what happens as we increase r further. When r is set to just larger than $(1 + \sqrt{6}) \div 4$, the logistic map will bifurcate a second time, giving a period-4 limit cycle. Figure 10.5 shows the period-4 cycle in time-series and state-space forms.

If we increase r even more, we will eventually force the system into a period-8 limit cycle, then a period-16 cycle, and so on. The amount that we have to increase r to get another period doubling gets smaller and smaller for each new bifurcation. This cascade of period doublings is reminiscent of the race between Achilles and the tortoise, in that an infinite number of bifurcations (or time steps in the race) can be confined to a local region of finite size. At a very special critical value, the dynamical system will fall into what is essentially an infinite-period limit cycle. This is chaos. Figure 10.6 shows the time series and state space of the logistic map for $r = 1$, which is in the chaotic regime.

Figure 10.7 illustrates the transition from order to chaos with two bifurcation diagrams. The x-axis represents a value for r, and the y-axis shows the period of the limit cycle. Notice that the diagram clearly shows how each period doubling spans a shorter amount of space than the previous doubling. Eventually the period doublings converge to what looks like an infinite-period attractor. Notice also that

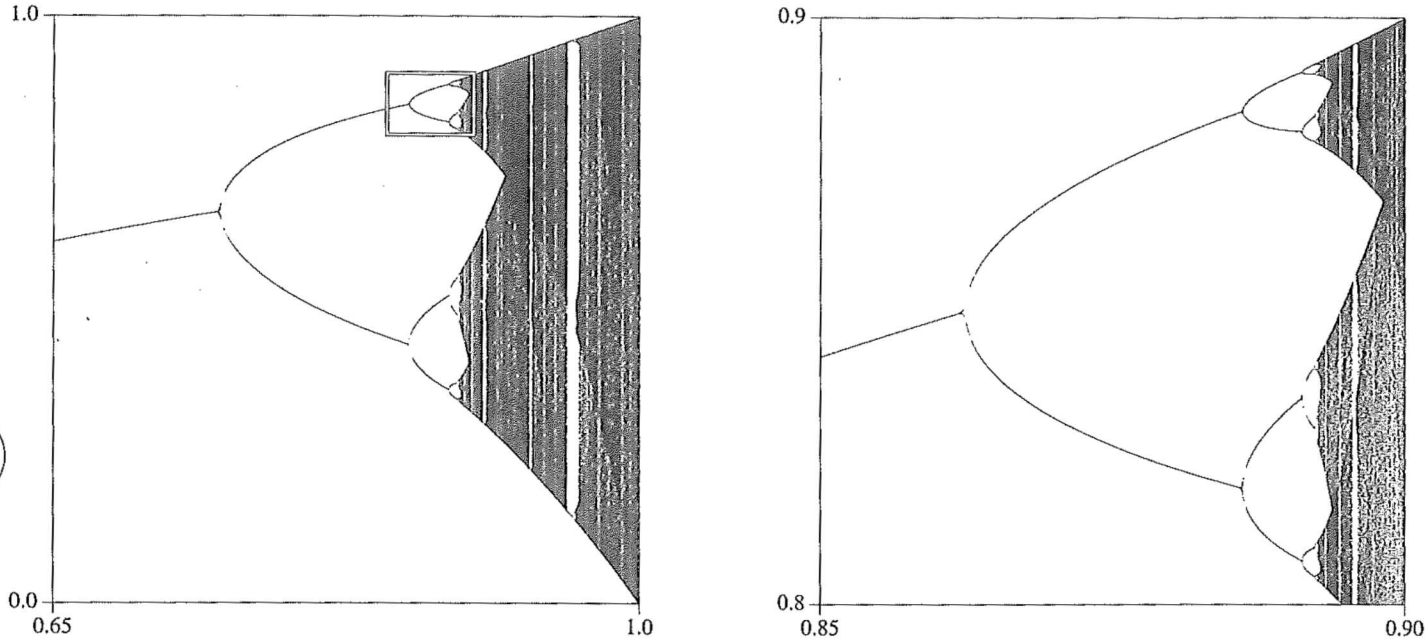


Figure 10.7 Bifurcation diagrams for the logistic map: (a) This image has values of r such that fixed points, limit cycles, and chaos are all visible. (b) This image shows the detail of the boxed section of (a).

the two images are very self-similar in that we could enlarge a portion of Figure 10.7b to get another self-similar section, and so on.

Bifurcations of this sort are seen in many different types of phenomena. What is truly astonishing is that for a very broad class of bump-like functions, such as the logistic map, the cascade of bifurcations behaves in accordance with a universal number known as the *Feigenbaum constant*. For notational convenience let's refer to the value of r at which the logistic map bifurcates into a period-2^n limit cycle as a_n, that is, $a_1 = \frac{3}{4}$. Thus, we would know that for $a_{k-1} < r \leq a_k$ the logistic map would have a stable period-2^k limit cycle. Mitchell Feigenbaum considered the properties of the series of numbers generated by

$$d_k = \frac{a_k - a_{k-1}}{a_{k+1} - a_k},$$

for $k \geq 2$. In the equation above, the numerator and denominator represent the distance between successive bifurcation points, as is shown in Figure 10.8. Therefore, the whole expression, being the ratio of the two distances, quantifies how fast the next bifurcation occurs relative to the previous one. Feigenbaum showed that $d_\infty = 4.669202\cdots$, not just for the logistic map but for all one-dimensional maps that have a single hump. Thus, every chaotic system that falls into this class bifurcates at the same rate.

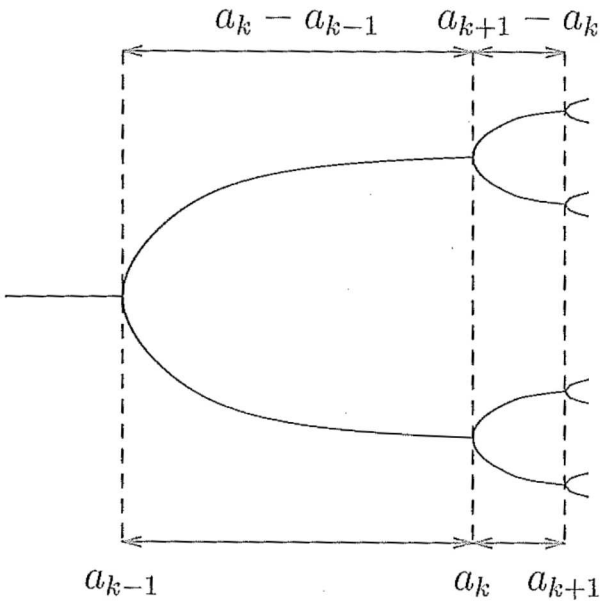


Figure 10.8 Detail of a bifurcation diagram to show the source of the Feigenbaum constant

By itself, this is an amazing result, yet it turns out that there is a very practical application for this knowledge. Suppose you are interested in a chaotic process that has a single tunable parameter like r for the logistic map. By empirically noting where the first few bifurcations occur, it is possible to get an accurate estimate for a_∞, which is the point where the system becomes chaotic. Thus, it is possible to estimate when a system will become chaotic before it ever happens.

10.4 Prediction, Layered Pastry, and Information Loss

One of the most important differences between chaotic processes and truly stochastic processes is that the future behavior of a chaotic system can be predicted in the short term, while stochastic processes can be characterized only statistically. For example, in a fair coin toss, knowing the results from earlier coin tosses does absolutely nothing for you if you are trying to predict what the next coin toss will yield. The best you could do, in terms of being able to predict the future, would be to generalize over an entire history, for instance, we could safely say that about half of our future coin tosses will result in heads; however, we are powerless to make any accurate assertions about any particular random event, such as the very next coin toss. Chaotic processes, on the other hand, can be predicted, since chaos always has hidden order within it. For many chaotic processes, the order becomes apparent when one looks at state space plots, as was done earlier. In contrast to chaos, plotting x_t against x_{t+1} for a stochastic process will reveal no hidden structure whatsoever.

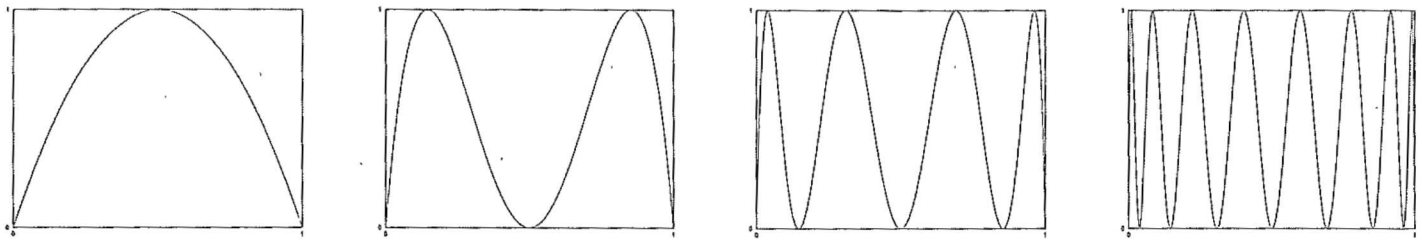


Figure 10.9 Functional mappings with $r = 1$: (a) $f^1(x)$, (b) $f^2(x)$, (c) $f^3(x)$, (d) $f^4(x)$

The flipside of this is that long-term prediction of a chaotic process becomes more difficult the farther into the future we try to predict. To predict one time step into the future, we would need to have an idea of what $f(x)$ looked like. If we wanted to predict two time steps into the future, we would have to examine $f(x)$ recursively applied to itself as $f(f(x))$. For the general case, let's agree that $f^m(x)$ denotes the mapping function of $f(x)$ recursively applied to itself a total of m times. Figure 10.9 illustrates how the mapping functions become more and more complicated as we increase the number of time steps into the future that we are looking at.

The pathological nature of chaos can be better appreciated with an analogy. Suppose you are a pastry chef who wants to make a pastry crust that has many layers in it. One way to accomplish this would be to stretch out your dough, fold one half over the other, and then repeat as often as needed. Notice that each step doubles the number of layers. It turns out that the motion of a chaotic system is very similar to the way we would form the pastry crust. For the logistic map, the $4rx$ portion of the function "stretches" out the input surface, while the $-4rx^2$ term "cuts" and "folds" the input. In this way, two grains of sugar that were originally close to one another in the pastry dough would gradually move away from one another, becoming decorrelated from one another as time went on. The sensitivity of chaotic systems is a result of these mixing actions.

Looking again at Figure 10.9, we can see that $f^m(x)$ has 2^{m-1} "humps" in it, which is similar to the way our pastry has an exponential number of layers. We will now see how all of this relates to a firm theoretical limitation on how far into the future one can predict. To make the analysis a bit simpler, let's assume that you only need to distinguish x_t values that are greater than or equal to $\frac{1}{2}$ from those that are less than $\frac{1}{2}$. In other words, if you are predicting m time steps into the future and I supply you with a value for x_t, your prediction scheme should return 1 bit of information: either "Yes, $x_{t+m} \leq \frac{1}{2}$" or "No, $x_{t+m} > \frac{1}{2}$."

To represent the starting value of x_t, we must ultimately encode the information in binary form. First, consider the case of $m = 1$. To determine if x_{t+1} is greater than (or less than) $\frac{1}{2}$, we essentially need to partition the input space into at least

t	x_t^a	x_t^b	$\lvert x_t^a - x_t^b \rvert$
0	0.987654321	0.987654320	0.000000001
1	0.048773053	0.048773057	0.000000004
2	0.185576969	0.185576983	0.000000014
3	0.604552629	0.604552665	0.000000035
4	0.956274991	0.956274961	0.000000030
5	0.167252531	0.167252639	0.000000108
6	0.557116488	0.557116776	0.000000288
7	0.986950827	0.986950696	0.000000132
8	0.051515568	0.051516080	0.000000512
9	0.195446856	0.195448694	0.000001839
10	0.628989529	0.628994009	0.000004480
11	0.933446806	0.933442183	0.000004623
12	0.248495467	0.248511497	0.000016030
13	0.746981880	0.747014131	0.000032252
14	0.755999804	0.755936076	0.000063729
15	0.737856401	0.737986901	0.000130500
16	0.773697331	0.773448940	0.000248390
17	0.700359085	0.700902708	0.000543623
18	0.839424949	0.838552408	0.000872541
19	0.539162817	0.541529069	0.002366252
20	0.993865095	0.993101346	0.000763749
21	0.024389072	0.027404252	0.003015180
22	0.095176980	0.106613035	0.011436055
23	0.344473289	0.380986782	0.036513494
24	0.903245768	0.943343416	0.040097648
25	0.349571401	0.213786462	0.135784940
26	0.909484947	0.672327242	0.237157705
27	0.329288313	0.881213286	0.551924973

Table 10.1 Exponentially divergence: With two slightly different starting positions, two chaotic trajectories will exponentially diverge from one another.

three sections: the middle section, where the hump is greater than $\frac{1}{2}$, and the left and right extremes, where the function descends below $\frac{1}{2}$. For arbitrary values of m, we need to partition the input space into $2^m + 1$ sections. Consequently, we need at least $m + 1$ bits of accuracy to encode the initial value for x_t. If our floating-point representation for numbers uses any number of bits less than $m + 1$, then the accuracy of the predictions can be no better than a random guess. The situation is such that in a very real way, we lose 1 bit of information for each time step into the future that we try to predict.

To put all of this in perspective, the most advanced computers today use 128 bits for floating-point numbers. So even though the logistic map is an extremely simple system, modern computers can make only marginally accurate predictions for less than 128 time steps into the future. You may be thinking that a clever programmer could work around this by using more bits for the floating-point numbers. However, since a computer's memory is finite in size, there will always be a value m for which all computers are helpless to predict what x_{t+m} looks like.

The problem gets even worse when we consider two other factors. First, for all of this section we have been assuming that our initial measurement of x_t was accurate. In the real world there is always some sort of measurement error. Any error, even something unbelievably infinitesimal, will grow at an exponential rate, breaking any prediction scheme we can think of. The second factor relates to some of the discussion from Chapter 2. More specifically, what happens when the state falls on an irrational number? Since there are infinitely many more irrational than rational

numbers, any encoding scheme that attempts to represent all numbers by rational numbers must have—by definition—at least some error in the storage technique.

We will now demonstrate all of this with an experiment that you can perform on a simple calculator. We are going to compute two chaotic trajectories of the logistic map for $r = 1$. One trajectory, which we will label x_t^a, will have an initial value of $x_0^a = 0.987654321$. The second trajectory will be labeled x_t^b, and use an initial value of $x_0^b = 0.987654320$. Therefore, the two starting positions differ only by about 1 part in a billion, which would be regarded as an amazingly accurate measurement in the real world.

To represent a decimal number requires about three bits of information per decimal place. Since our numbers have nine digits, there is a total of approximately twenty-seven bits of information per number. Therefore, let's see what happens to the two trajectories after twenty-seven time steps. For the first few steps the error seems quite tolerable. In fact, the two trajectories are very similar for about the first twenty steps. It is not until the very last step that x_t^a and x_t^b fall on different sides of the $\frac{1}{2}$ divider; however, from this point on, the two trajectories will have no correlation with each other at all, since, as can be seen in Table 10.1, the error roughly doubles in size for each additional time step.

10.5 The Shadowing Lemma

There are a few troubling facts about the results from the last section. As touched on briefly before, since computers can represent only digital quantities and approximate real numbers with finite precision, any computer simulation of a chaotic system is doomed to degrade increasingly the farther into the future one tries to predict. Another facet of this problem involves the fact that if we simulate a chaotic system in a computer, at some point in the future the simulated system must start to repeat itself because of the finite precision available. In other words, computers cannot really generate aperiodic trajectories, but only limit cycles with, admittedly, very long periods.

How, then, do we know if computer simulations of chaos are valid in the sense that they yield true characterizations of real chaos? Worse still, is it possible that "chaos" is nothing more than a computer artifact that results from trying to represent a stochastic world with digital numbers? The *shadowing lemma* is a remarkable result that has an answer to these questions, at least for certain types of chaotic systems.

Figure 10.10 contains two graphs that illustrate the shadowing lemma in action. The graph on the left shows an exact trajectory (from a chaotic system) plotted alongside a simulated one. For the reasons outlined earlier, the two trajectories inevitably drift apart.

The graph on the right of Figure 10.10 shows the same computed trajectory as in the first graph, but this time a *shadow trajectory* is plotted alongside it. For

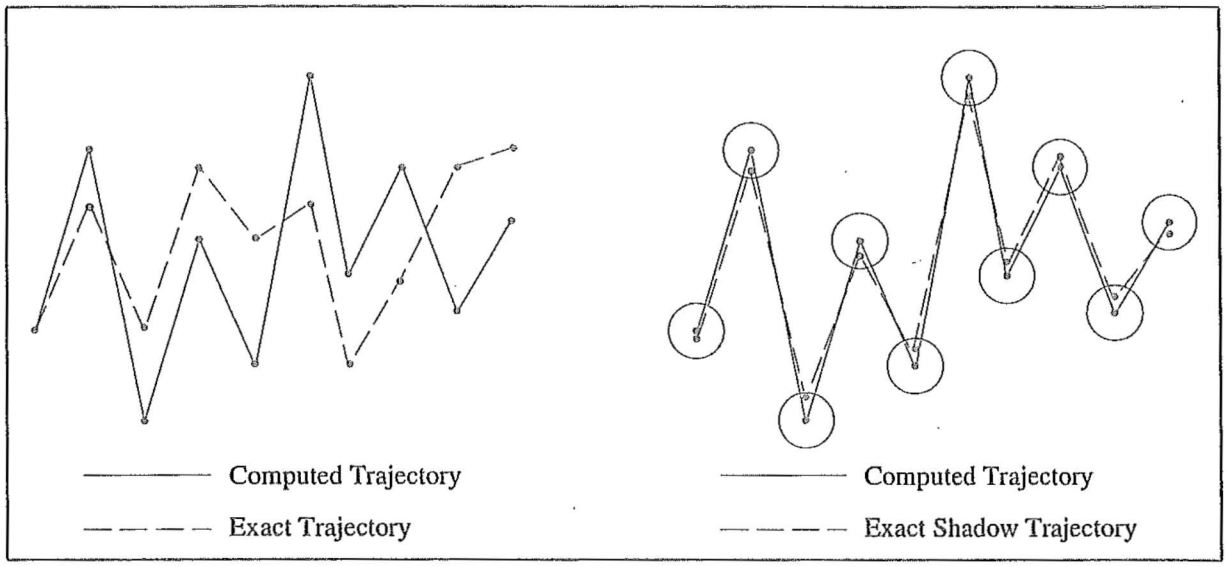


Figure 10.10 Shadow trajectories: On the left, the computed trajectory deviates from the exact (real) trajectory because of accumulated errors. On the right is a shadow of the computed trajectory that is arbitrarily close to the computed trajectory.

any computed trajectory, the shadowing lemma posits the existence of a shadow trajectory that follows arbitrarily close to the computed trajectory. In other words, the shadowing lemma tells us that a computer simulation of chaos does, in fact, provide an accurate picture of the motion in a chaotic system.

As an example of what the shadowing lemma means, consider the problem of weather forecasting. We know that all weather forecasts are doomed to be inaccurate in the long term. However, if we look at a weather simulation as it evolves over time, it is quite possible that the simulation captures all of the richness and complexity of the real weather system as it could happen.

10.6 Characteristics of Chaos

In this section we will summarize the characteristics of chaos that are common to all chaotic systems. Most of the issues that we will discuss in this section simply summarize material from the earlier portions of the chapter. The other topics that we will discuss will serve as motivation for the latter chapters in this part.

The goal of this section is to provide a working definition of chaos. This task is somewhat difficult in that many scientists do not agree on a single definition of chaos. Nevertheless, the characteristics listed in this section should go a long way toward showing you what chaos looks like. Also, keep in mind that these characteristics are necessary but not sufficient, that is, all chaotic systems will have all of these characteristics, but just because a dynamical system possesses one of these attributes does not make it chaotic.

Deterministic Chaotic systems are completely deterministic and not random. Given a previous history of a chaotic system, the future of the system will be completely determined; however, this does not mean that we can compute what the future looks like.

Related to this distinction between chaotic and random processes are the topological properties of their respective state spaces. The motion of a random process is, by definition, uncorrelated with the previous states of the system. As such, if you looked at the state space of a random process, you would see only a "blob-like" structure with no order whatsoever. Complementary to this, for all chaotic systems there is always a way of showing structure in the system's state space. We may need to look at a higher-dimensional state space in order to see this structure, but the structure is there, nevertheless.

Sensitive Chaotic systems are extremely sensitive to initial conditions, since any perturbation, no matter how minute, will forever alter the future of a chaotic system. This fact has sometimes been referred to as the "butterfly effect," which comes from Edward Lorenz's story of how a butterfly flapping its wings can alter global weather patterns. In his book *Chance and Chaos*, David Ruelle gives an even better example of sensitivity to initial conditions, which I will paraphrase here. Suppose that by some miracle the attractive effect of a single electron located at the limit of the known universe could be suspended momentarily. How long do you think it would take for this slight perturbation to change the future on a macroscopic scale? Since the motion of air causes individual molecules to collide with one another, it would be interesting to know how long it would take for these collisions to be altered. Amazingly, after only about fifty collisions the molecules in Earth's atmosphere will have collided in a different manner than they would have originally. If we wait another minute or two, the motion in a turbulent portion of the atmosphere will be altered at the macroscopic level. And, if we wait another week or month, the motion of the entire weather system will be measurably altered.

Thus, the next time you hear your local weather forecaster tell you what next week's weather will look like, you can recognize it for the nonsense it is. But this is not to say that all prediction is hopeless, since chaotic systems can be predicted over the short-term with a fair amount of accuracy. Another facet of the sensitivity of chaotic systems is that in many ways chaotic systems are more susceptible to control. You can look at it this way: Since any perturbation causes an exponential divergence in the state space trajectory of a chaotic system, with a minute alteration to the system we could profoundly alter the system's future behavior.

Ergodic Chaotic motion is ergodic, which means that the state space trajectory of a chaotic system will always return to the local region of a previous point in the trajectory. For example, we can define a local region of interest by a single point in a state space and a distance measure. Every point in the state space that is within

the specified distance from the point will be considered to be in the local region of the point. If a system is ergodic, then no matter how small we make the local region, as long as it's nonzero, we are guaranteed that the system will eventually return to this local region. Using the weather as an example, ergodicity means that it is very likely that someday in the future you will experience weather almost—but not exactly—identical to today's weather.

Embedded Chaotic attractors are embedded with an infinite number of unstable periodic orbits. Looking back at Figure 10.9, recall that the function $f^m(x)$ got more and more complicated as we increased m. For any of the plots in Figure 10.9, it is easy to see that the identity line, $y = x$, will intersect $f^m(x)$ at multiple locations. For all points, x_p, where $f^m(x)$ intersects the identity line, x_p will be part of a period-2^{m-1} unstable limit cycle. Since we can do this for any positive value of m, it follows that there must be an infinite number of limit cycles embedded within the chaotic attractor.

Over the next four chapters we will examine these and other properties of chaotic systems. Chapter 11 will introduce us to chaos in multidimensional systems, strange attractors, as well as some important tools of the trade for visualizing chaos. In Chapter 12 we will examine a form of chaos that is commonly found in chemistry, biology, ecology, and economics. In Chapter 13 we will take a closer look at how chaos can be exploited to control the future. Finally, in Chapter 14, we will see how incomputability, randomness, and chaos can be seen as multiple facets of a single phenomenon.

10.7 Further Exploration

There are four programs that I used to generate all of the computed figures in this chapter. In this section, I will briefly describe how to use these programs. The command-line options for all of the programs are summarized in Table 10.2. As with just about every other program listed in this book, most of the programs for this chapter use the `-term` option to specify how images are plotted.

Another option that is common to all of the programs for this chapter is the `-func` option, which can be used to specify what one-dimensional map to use. The logistic map is the default map to use. Other possibilities include the tent map, the sine map, and the Gaussian map.

The first program that you should try is `gen1d`, which will generate a one-dimensional time series. You may wish to use the `-x0` option to specify the initial point, and the `-r` option to specify the value of r to use.

The next program is `phase1d`, which can be used to generate state space plots. As with `gen1d`, the `-points` option can be used to specify the number of points

Option Name	Option Type	Option Meaning
Options Common to All Featured Programs		
`-points`	INTEGER	number of points to plot
`-skip`	INTEGER	number of points to skip
`-aux`	INTEGER	value for auxiliary map parameter
`-x0`	INTEGER	initial value, x_0
`-func`	STRING	which map function to use
Options Only for `gen1d`, `phase1d`, and `phase1d2`		
`-r`	INTEGER	value for r
Options Only for `phase1d`, `phase1d2`, and `bifur1d`		
`-width`	INTEGER	width of the plot in pixels
`-height`	INTEGER	height of the plot in pixels
`-inv`	SWITCH	invert colors?
`-xmag`	INTEGER	magnification factor for X Windows
`-term`	STRING	how to plot points
Options Only for `bifur1d`		
`-rmin`	INTEGER	smallest value for r
`-rmax`	INTEGER	largest value for r
`-factor`	INTEGER	multiplicative factor for iterates
`-ymin`	INTEGER	smallest value for y-range
`-ymax`	INTEGER	largest value for y-range
`-box`	INTEGER	line width for a box
`-brmin`	INTEGER	smallest r-value for the box
`-brmax`	INTEGER	largest r-value for the box
`-bymin`	INTEGER	smallest value for box y-range
`-bymax`	INTEGER	largest value for box y-range

Table 10.2 The command-line options for `gen1d`, `phase1d`, `phase1d2` and `bifur1d`

in the series. You may also change the initial value with `-x0`. You can skip the first few initial points with the `-skip` option. This is useful if you only want to see where a map will converge to, and not how it got there.

The program `phase1d2` is identical to `phase1d` except that you may use the `-dx` option to specify where a second trajectory will start relative to the first trajectory. This is useful if you want to see how rapidly two nearby points will diverge.

The last program for this chapter is `bifur1d`, which you can use to plot bifurcation diagrams. You need to supply values with `-rmin` and `-rmax` to specify the

first and final values of r to be contained in the bifurcation plot. Similarly, you can change the y-ranges with `-ymin` and `-ymax`. If you wish to use this program to generate a PostScript image, generating a `PGM` file and converting it to PostScript with `pnmtops` or `xv` is far more efficient.

Since the source code for all of the programs in this chapter includes the file `maps1d.c`, to extend the capabilities of all of the programs, you only need to modify `maps1d.c`. Contained in this file are the function definitions for the various one-dimensional maps that you can use. The file is documented with an example of how to add your own functions.

10.8 Further Reading

Gleick, J. (1987). *Chaos.* New York: Viking.

Ruelle, D. (1993). *Chance and chaos.* Princeton: Princeton University Press.

Stewart, I. (1990). *Does God play dice?: The mathematics of chaos.* Oxford: Blackwell.