Iterated Maps

Iterated maps are probably the simplest display of non-linear systems. Iterated maps are produced by putting a number through a function then taking the result and putting it through the equation again, then repeating. Try this with the cos button on your calculator (you must have a calculator that has radians as an angle measure, the scientific calculator under windows will work fine.) Just pick any starting value and keep pressing the cos button. The series of numbers will eventually approach a stable point and there will be no more changes. The stream of numbers is called an 'orbit', the final point is called a stable fixed point.

This exercise shows what an iterated map is, however it is a very dull example of an iterated map. Iterated maps are what many fractals are made of. The Mandelbrot set is just an iterated map in the complex number plane. Here we will deal with simple one dimensional maps, because they are easier to study. If you wish to study more complex maps, the links page has several fractal links which may or may not attack the subject from a mathematical standpoint. Many fractal sited are dedicated to fractal art.

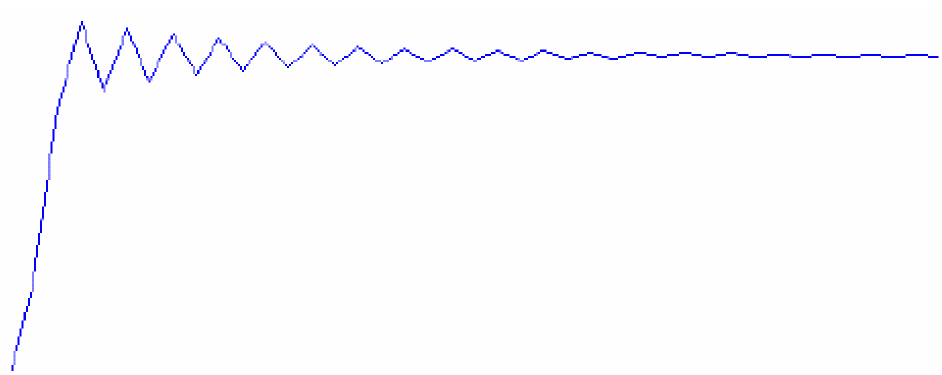
The Logistic Equation

A particularly interesting, and popular iterated map is the logistic map. This map shows many of the features that we will see appearing later on in continuous systems. The logistic equation is actually a simple model for species population with no predators, but limited food supply. It is given by the following equation:

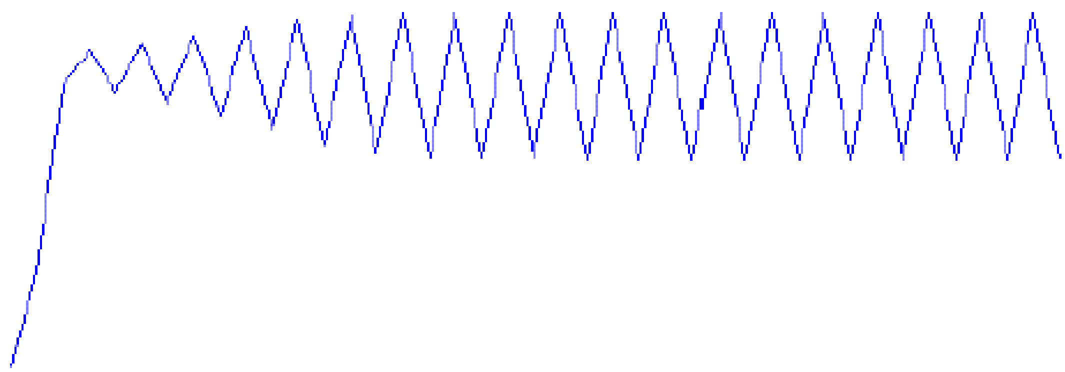
$$x_{n+1} = r x_n (1 - x_n)$$

where r is a parameter to be set anywhere from 0 to 4. The initial x must be from the region 0 to 1. This equation is much more difficult to analyze on a calculator, so we will be using a Java applet to help with the analysis.

To start, we will be setting r to 2.9. To see what happens, we plot a time series plot of the orbit. As the function is iterated it approaches a stable point. This is similar to the exercise performed earlier. There is actually an unstable fixed point at 0, try plugging zero into the equation for x and see what you get. This point is unstable because if the initial conditions do not start exactly on zero, then they will go to the stable point. The origin is called a repeller, while the stable point is an attractor. In the population example, the origin corresponds to a zero population. Life does not spring from nothing. The parameter r is the amount of food supply. For this amount of food supply the population grows to a point, then settles down to a steady state.



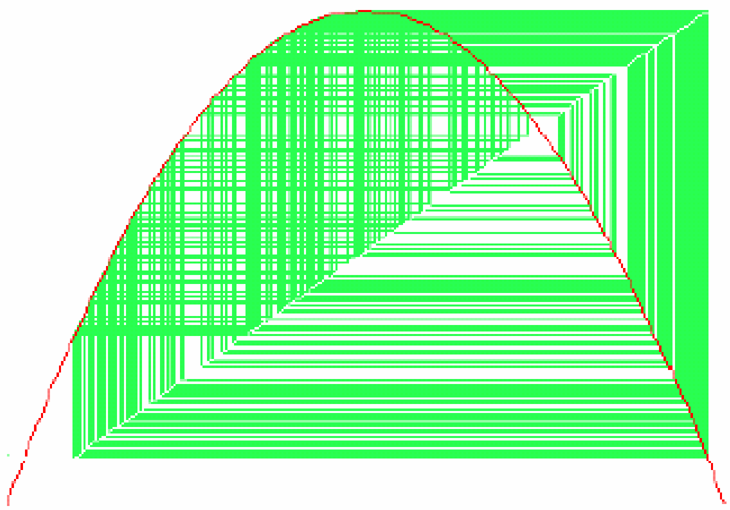
If we increase r past 3.0 then something more interesting happens. The orbit does not settle down to a fixed point. The fixed points that were there before have lost stability, now the system will cycle between two points. This is called a stable cycle, in this case, a stable 2-cycle. In our population, the food has been increased. Now a small generation has so much food that it makes a rapid growth spurt, however, in the next generation, there are too many in our population and not enough food, so the population dies off a bit. This is actually stable behavior, and is seen in some bacteria cultures!



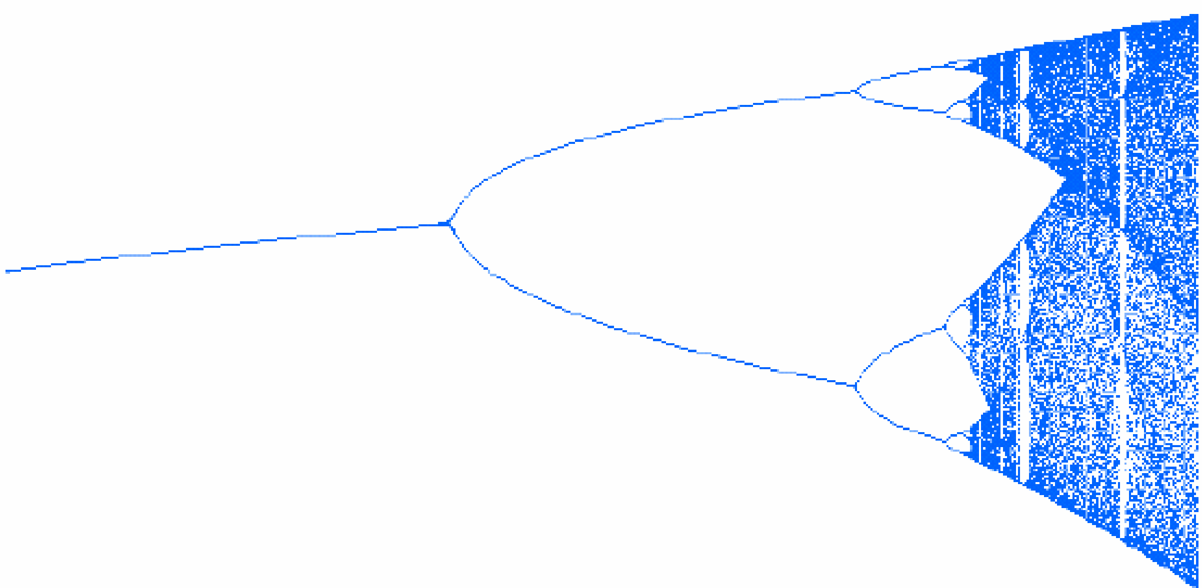
If we keep increasing r, this two cycle becomes a four cycle, then an 8 cycle and so on. Before we examine this, lets first take a look at a nice way of seeing this visually.

What we are doing here is taking a point x1, evaluating x2 = f(x1), then x3 = f(x2), and so on. If we plot (x1,x2), this is a point on the logistic curve. Drawing a horizontal line to (x2,x2) gives a point on the diagonal line. To get back onto the logistic curve we draw a line to (x2,x3), then back to the diagonal line at (x3,x3). This probably seems like a strange way to see the logistic orbit, but if you experiment with it, you can see stable fixed points, stable cycles and anything else this equation may hide very easily. Here is a Java applet that allows you to do just that. These plots are called cobwebs, (for reasons you will see shortly). Experiment with r=2.9, and r=3.2. You will see the fixed point and the two cycle that were covered earlier. The applet is used by setting the value of r in the 'r' text box. The other two text boxes, start cycle number and end cycle number, allow you to calculate a number of initial cycles, to let any messy parts of the cycle die out before the applet starts drawing. If start is 300 and end is 600 you should get a pretty neat picture of the cobweb. If you want to see the orbit right from the beginning, set start to 0. Initial x is the starting point of the cobweb.

Now that we have cobwebs under our belt, we can increase r further. If you didn't try increasing r past 3.2, try it now, try r=3.5 then r=3.565. You will see that the cycle has changed to a 4-cycle then an 8-cycle. These changes are called bifurcations. At a bifurcation the system undergoes a massive change in long term behavior. As r is increased, the bifurcations come faster and faster, until finally at about 3.5699 the cycle length becomes infinite. If r is further increased from 3.5699 up (but still below 4.0) then the system no longer has a cycle, it bounces about forever, but never repeats itself. Here is a cobweb for r=3.9.



This behavior is chaos. There is another way to easier see these bifurcations. If the stable points, or stable cycles are plotted as a function of r, then each of the cycles can be seen bifurcating into a cycle twice as long. After r is increased past 3.5699, chaos appears, but there are windows of periodic behavior interspersed with the chaos.



This route to chaos is called a period doubling cascade. It appears in many real life systems and very closely resembles this map. This is known as universality, the same simple map appears over and over again in just about every type of chaotic system.