

# Deterministic Chaos in the Logistic Map

## Chaos in Classical Dynamics

The equations of classical dynamics are systems of ordinary differential equations with time as the independent variable. Newton's equations

$$m \frac{d^2 \vec{r}}{dt^2} = \vec{F}(\vec{r}, t) \quad (1)$$

for a single particle, for example, are three second order differential equations. These equations are perfectly deterministic, which means that given initial conditions determine the time evolution of the system uniquely. If the equations are nonlinear, their solutions can exhibit surprisingly complex behavior for some initial conditions and particular values of physical parameters. This behavior can be so complex that it becomes impossible to predict the time evolution of the system given finite computing resources. Such behavior is called "chaotic". It is important to note that chaos in classical dynamics does not imply true randomness: a given set of initial conditions determines a unique solution. Chaos sets in when tiny changes in the initial conditions get amplified exponentially and lead to very different final states. Tiny errors introduced when one makes numerical approximations also get amplified in an uncontrollable way.

## The Logistic Equation

Chaotic motion in classical dynamics is difficult to study numerically. We will therefore start with a simpler equation, which has applications in population biology. Let  $N(t)$  be the number of members of a species. The equation

$$\frac{dN}{dt} = aN, \quad (2)$$

where  $a$  is a constant growth rate, provides a simple model for exponential population growth or decay. This is a linear equation, and it is exactly solvable:

$$N(t) = N(0)e^{at}. \quad (3)$$

This solution, although not chaotic, shares one of the characteristics of chaotic behavior, namely extreme sensitivity to initial conditions: if  $a > 0$ , then a single additional individual at the  $t = 0$  leads to an exponentially large number  $e^{at}$  of additional individuals at later times. In this case, this extreme sensitivity is not considered chaotic because it can be predicted exactly: chaotic equations exhibit similar exponential divergence of solutions which have almost the same initial values, but it is impossible in practice to predict such solutions exactly.

The logistic equation is a nonlinear generalization of exponential growth equation:

$$\frac{dN}{dt} = aN - bN^2, \quad (4)$$

where  $b$  is a positive constant. The quadratic term  $-bN^2$  models environmental pressure on the population which tend to suppress growth if  $N$  gets too large. This equation does not have chaotic solutions. It can be shown that at least 3 coupled nonlinear equations are required for chaos to occur.

# The Logistic Map

This is a discrete-time approximation to the logistic equation:

$$\frac{dN}{dt} \simeq \frac{N(t_{n+1}) - N(t_n)}{\Delta t} = aN(t_n) - bN^2(t_n), \quad t_n = n\Delta t, \quad n = 0, 1, \dots, \quad (5)$$

where  $\Delta t$  is a fixed time step. Some such approximation must be made to solve the equation numerically using a digital computer: such a machine can only perform a finite sequence of operations, while physical time is essentially an infinite continuum. This approximation is also appropriate for a biological population with non-overlapping generations, if  $\Delta t$  is taken to be the time between successive generations of individuals. If we define

$$x_n \equiv \frac{b\Delta t N(t_n)}{(1 + a\Delta t)},$$

the logistic map takes the simple form:

$$x_{n+1} = 4rx_n(1 - x_n) \equiv f(x_n), \quad r = \frac{(1 + a\Delta t)}{4}. \quad (6)$$

$N$ , being the number of individuals, is necessarily non-negative. If  $a$  and  $b$  are positive,  $N(t_n) \geq 0$  implies that  $x_n \geq 0$ . If  $a$  is positive, so is  $r$ , and then  $x_{n+1}$  will be non-negative if  $0 \leq x_n \leq 1$ . The “logistic function”  $f$  has a maximum at  $x = 0.5$ , where  $f(0.5) = r$ . We can ensure that  $0 \leq x_{n+1} \leq 1$  if  $r$  is restricted to the interval  $0 \leq r \leq 1$ .

## Numerical Algorithm for the Dynamics

It is very simple to write a computer program to generate the sequence of values  $x_n$  for a given value of the parameter  $r$ . The following code, which is written in the Java programming language, is a simple way of “iterating” the map:

\_\_\_\_\_ Program 1: <http://www.physics.buffalo.edu/phy410-505/topic4/logistic.cpp> \_\_\_\_\_

```
#include <iostream>
#include <fstream>
using namespace std;

int main() {
    cout << " Logistic map trajectory\n"
         << " -----\n"
         << " Enter 0 < r < 4: ";
    double r;
    cin >> r;
    cout << " Enter initial x in [0, 1]: ";
    double x;
    cin >> x;
    cout << " Enter number of steps N: ";
    int N;
    cin >> N;
    ofstream dataFile("logistic.data");
    for (int i = 0; i <= N; i++) {
        dataFile << i << '\t' << x << '\n';
    }
}
```

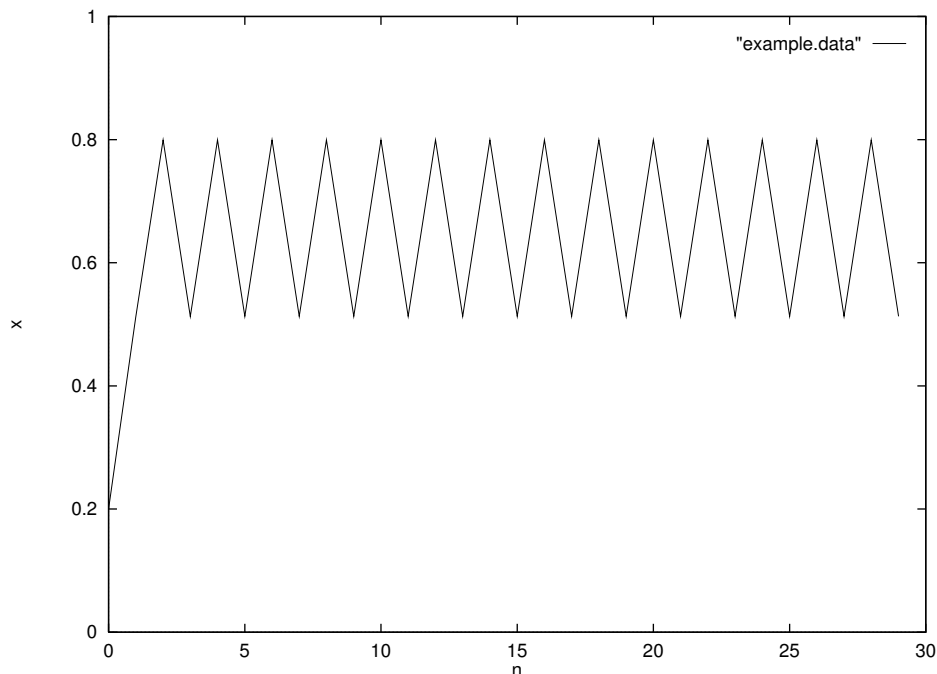


Figure 1: Behavior of the logistic map with parameter value  $r = 0.8$ . Note the initial transient behavior, which is followed by a periodic oscillation between two values of  $x$ .

```

    x = 4 * r * x * (1 - x);
}
cout << " Trajectory in file logistic.data" << endl;
dataFile.close();
}

```

## Fixed Points of $f$

The fixed points of the logistic map are determined by the equation

$$x^* = f(x^*), \quad (7)$$

which has solutions

$$x^* = 0, \quad \text{and} \quad x^* = 1 - \frac{1}{4r}. \quad (8)$$

If  $x_n = x^*$ ,  $x_{n+1} = x_{n+2} = \dots = x^*$ . This sequence of equal values is also called a 1-cycle because it is a periodic solution of the logistic map with period equal to one generation. It is easy to show that for any starting value  $x_0$  such that  $0 < x_0 < 1$ ,

$$\lim_{n \rightarrow \infty} x_n = \begin{cases} 0, & \text{for } 0 < r \leq 0.25, \\ 1 - \frac{1}{4r}, & \text{for } 0.25 < r \leq 0.75. \end{cases} \quad (9)$$

After some initial “transient” behavior, the limiting fixed point values are approached rather rapidly. In the parameter range  $0 < r < 0.25$ , any initial  $x_0 \neq 1 - 1/(4r)$  approaches  $x^* = 0$ , which is called an

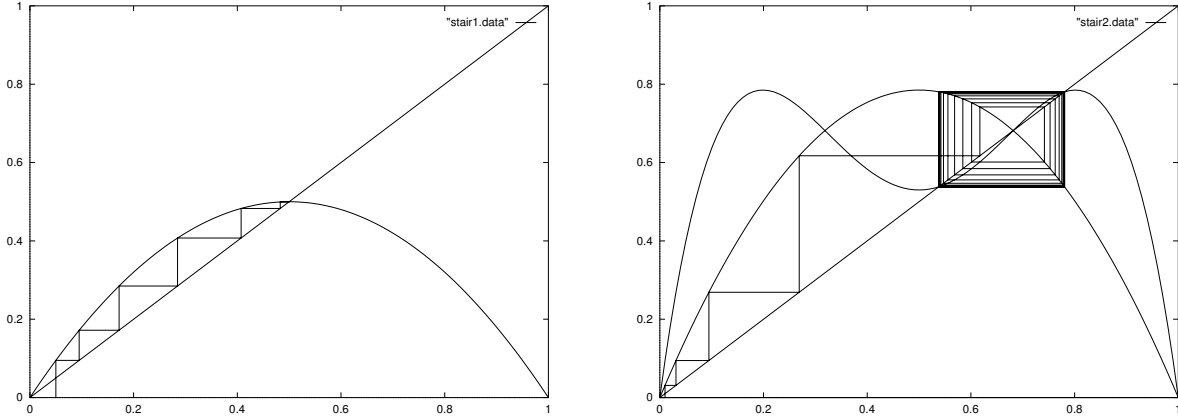


Figure 2: (a) The graph on the left shows the staircase construction for  $r = 0.5$  and  $x_0 = 0.05$ . Also plotted on this graph are the map function  $f(x)$  and the fixed-point line  $x_{n+1} = x_n$ . Notice that the staircase converges to the fixed point  $x^* = 0.5$ . (b) The graph on the right shows the staircase construction for  $r = 0.785$  and  $x_0 = 0.01$ . The double-hump curve is the second iterate  $f(f(x))$  of the map function. Note that the staircase oscillates asymptotically between two stable fixed points of the second iterate.

“attractor”. The set of initial values which tend to the attractor is called its “basin of attraction”. If  $x_0 = 1 - 1/(4r)$  exactly, then the solution remains stuck at this point if  $0 < r < 0.25$ , but even the slightest deviation from  $1 - 1/(4r)$  causes the solution to run away to  $x^* = 0$ .  $x^* = 1 - 1/(4r)$  is therefore called an “unstable” fixed point, whereas  $x^* = 0$  is a “stable” fixed point. In the parameter range  $0.25 < r \leq 0.75$ ,  $x^* = 0$  becomes unstable, and  $x^* = 1 - 1/(4r)$  becomes stable.

The approach to a fixed point can be demonstrated graphically by plotting  $x_{n+1}$  as a function of  $x_n$ . The fixed points of the map are given by the intersections of this function with the diagonal line  $x_{n+1} = x_n$ . Given an initial value  $x_0$ , the sequence generated by the map can be constructed graphically by repeating the following pair of steps:

1. Move in the vertical direction to the curve  $f(x_n)$ ,
2. Move in the horizontal direction to the diagonal line.

This “staircase” construction is illustrated in Fig. 2. It is easy to see that if one is close to a fixed point, then this staircase construction will move towards the fixed point if the angle between the tangent to the curve  $f$  at the fixed point and the horizontal is less than  $45^\circ$ , and away from the fixed point if it is greater than  $45^\circ$ . Thus, the criterion for stability of a fixed point  $x^*$  is  $|f'(x^*)| \leq 1$ , where  $f' \equiv df/dx$ .

The slope criterion helps explain these results, and shows that both fixed points are unstable for  $r > 0.75$ .

## Period Doubling

The fixed points (2.8) become unstable if  $r > 0.75$ . The limiting behavior of the map when the parameter  $r$  is increased a little beyond 0.75 is a 2-cycle in which  $x_{n+1}$  oscillates between two fixed values as  $n$  increases. These fixed values are the fixed points of the second iterate  $f^2$  of the function  $f$ :

$$f^2(x_n) \equiv f(f(x_n)) = x_{n+2} = f(x_{n+1}) = (4r)^2 x_n (1 - x_n) [1 - 4r x_n (1 - x_n)]. \quad (10)$$

Note that  $f^2(x)$  is *not* the square of  $f(x)$ . The second iterate  $f^2$  is a fourth order polynomial, and it will therefore have 4 fixed points in general. Two of these must coincide with (2.8), because a fixed point of  $f$  is also a fixed point of  $f^2$ . In a 2-cycle, the solution oscillates back and forth between the other two fixed points. This 2-cycle will be stable if  $|f^{2'}| < 1$  at each of these two points, and this condition is satisfied in the range  $0.75 < r \leq 0.862\dots$

It is easy to see that the two new fixed points of  $f^2$  are given by

$$\frac{1}{2} \left[ 1 + \frac{1}{4r} \pm \sqrt{\left(1 + \frac{1}{4r}\right) \left(1 - \frac{3}{4r}\right)} \right].$$

Consider the behavior of this formula near the bifurcation point  $r = 0.75$ . The two new fixed points lie on opposite sides of the fixed point at  $1 - (1/4r)$ . The slope of the second iterate is given by  $f^{2'}(x_{n+2}) = f'(x_{n+1})f'(x_n)$ . Note that  $f^2$  has the same slope at the two new fixed points, which are a 2-cycle of  $f$ .

The two fixed points of  $f^2$  which give rise to the 2-cycle of  $f$  become unstable at the same value of  $r \simeq 0.862$ . If the stable fixed points are plotted as a function of  $r$ , the diagram in the vicinity of  $r = 0.75$  resembles a pitchfork with a handle branching into two tines of equal extent in  $r$ : this phenomenon is therefore referred to as a “pitchfork bifurcation”.

It can be shown that each of the two new fixed points of  $f^2$  undergoes a pitchfork bifurcation when it becomes unstable. The four tines of the two pitchforks are new stable fixed points of the fourth-order iterate  $f^4(x_n) = f(f(f(f(x_n))))$ , which is a polynomial of order 16 and can have as many as 16 fixed points. Four of these fixed points are stable: they constitute a 2-cycle of  $f^2$ , and a 4-cycle of  $f$ . The fixed points of  $f^4$  simultaneously become unstable at  $r_2 \simeq 0.880$ , where each of them undergoes a pitchfork bifurcation. This period doubling behavior repeats itself *ad infinitum* over a finite range of  $r$ , to give rise to a sequence of 2-, 4-, 8-, 16-,  $\dots$ , cycles. Let  $r_k$  denote the value of  $r$  at which the stable  $2^k$ -cycle of  $f$  first appears. It is found that  $r_\infty = 0.8925\dots$

This behavior of the logistic map can be summarized in a “bifurcation plot” such as that shown in Fig. 3. This is a plot of the limiting or asymptotic behavior of the population as a function of  $r$ .

\_\_\_\_\_ Program 2: <http://www.physics.buffalo.edu/phy410-505/topic4/bifurcate.cpp> \_\_\_\_\_

```
#include <fstream>
#include <iostream>
using namespace std;

double x0 = 0.5;           // starting x
double r1;                 // starting r
double r2;                 // ending r
int rSteps;                // steps in r
int stepsToDiscard = 20;   // discard transient behavior
int stepsToPlot = 50;     // number of steps to plot

double r;                  // current parameter value
double x;                  // current map point

void map() {
    x = 4 * r * x * (1 - x);
}

int main() {
```

Figure 3: For each value of  $r$ , the logistic map was started with  $x_0 = 0.5$ . The first 100 points were discarded, and the next 300 points retained.

```

cout << " Logistic Map Bifurcation Diagram\n"
    << " -----\n"
    << " Enter start and end r in [0,1]: ";
cin >> r1 >> r2;
cout << " Enter number of steps in r: ";
cin >> rSteps;
ofstream dataFile("bifurcate.data");
cout << " Initial x = " << x0 << "\tDiscarding " << stepsToDiscard
    << " steps, retaining " << stepsToPlot << " for each r" << endl;
for (int i = 0; i <= rSteps; i++) {
    r = r1 + (r2 - r1) * i / double(rSteps);
    x = x0;
    for (int j = 0; j < stepsToDiscard; j++)
        map();
    for (int j = 0; j < stepsToPlot; j++) {
        map();
        dataFile << r << '\t' << x << '\n';
    }
}
cout << " Data in file bifurcate.data" << endl;
dataFile.close();
}

```

## Feigenbaum's universal numbers

By experimenting with a pocket calculator, Feigenbaum discovered a remarkable property of the period doubling sequence of values  $r_k$ . He showed that they approach  $r_\infty$  in a geometric fashion:

$$r_\infty - r_k = \text{const} \times \delta^{-k}, \quad (11)$$

where

$$\delta = \lim_{k \rightarrow \infty} \left( \frac{r_{k+1} - r_k}{r_{k+2} - r_{k+1}} \right) = 4.6692016 \dots \quad (12)$$

It turns out that period doubling behavior is found in a large variety of systems. It occurs for example in any map  $f(x)$  which has a hump or maximum like the logistic map. The particular sequence of parameter values at which the doublings occur will depend on the form of the function  $f$ , but the constant  $\delta$  computed from these parameter values is precisely the same as for the logistic map! More remarkably, there are a large number of much more complex systems ranging from the nonlinear pendulum in mechanics, to fluids exhibiting turbulent flow in hydrodynamics, in which period doubling phenomena have been observed, and in which the constant  $\delta$  is measured to have the same value (within computational or experimental errors) as it does in the logistic map!

Feigenbaum discovered a second universal constant that occurs in the period doubling cascade of the logistic map. A  $2^k$ -cycle corresponds to the variable  $x_n$  cycling through the  $2^k$  stable fixed points of the

$2^k$ -th iterate of  $f$ . Let  $d_k$  be distance of that fixed point in the cycle which is closest to the maximum of the function  $f(x)$  at  $x = 0.5$ . The sequence of closest points oscillates about  $x = 0.5$  and approaches it at an exponential rate:

$$d_{k+1} \simeq -\left(\frac{1}{\alpha}\right) d_k, \quad (13)$$

where  $\alpha = 2.502907875\dots$  is a universal constant characteristic of period-doubling behavior.

## The chaotic region

The behavior of the map for  $0.8925 \leq r < 1$  is so complicated that it is termed chaotic. Chaos implies an absence of periodicity, since periodic behavior is perfectly predictable. Indeed, there are continuously many values of  $r$  in the chaotic region for which the map never returns to the same value of  $x$ . However, there is some order embedded in this chaos: there are infinitely many values of  $r$  at which the behavior of the map is periodic. As  $r$  increases beyond 0.8925, these cycles at first all have periods of even order. Periods of odd order begin to appear at  $r \simeq 0.9196$ , with cycles of large odd order appearing first.

A 3-cycle first appears at  $r \simeq 0.9571$  and persists over a window extending to  $r \simeq 0.9624$ . Its appearance can be understood by considering the third iterate  $f^3$  of  $f$ , which is a polynomial of order 8. At  $r = 0.9$ ,  $f^3(x)$  intersects the fixed point line at two points in the interval  $0 \leq x \leq 1$ , one of them at  $x = 0$ . Both of these are unstable. At the onset of the 3-cycle, the curve  $f^3$  becomes tangent to the fixed point line at three distinct values of  $x$ , two of them below the unstable fixed point, and one above it. As  $r$  increases, each tangent point crosses the fixed point line and thus gives rise to two new fixed points, one of which is stable and the other unstable. Thus there are three stable fixed points of  $f^3$ , which give rise to a 3-cycle of  $f$ . Just below the onset of the 3-cycle the map is non-periodic: the transition from this chaotic region to a stable 3-cycle is referred to as a ‘‘tangent bifurcation’’. The 3-cycle window ends when the three stable fixed points of  $f^3$  simultaneously become unstable. Each of these points undergoes a pitchfork bifurcation into two stable fixed points of  $f^6$ . This is followed by a cascade of pitchfork bifurcations into  $3 \times 2^k$ -cycles,  $k = 2, 3, \dots \infty$ . Above this region an infinite number of cycles of all possible integer periods can occur, as well as a continuously many points at which the trajectories are aperiodic.

## Lyapunov exponents

The sequence of values  $x_n$  generated by the logistic map is perfectly determined in principle. A chaotic sequence has the property that it is very difficult to compute in practice because small numerical errors tend to grow in an uncontrollable fashion. To understand this phenomenon, it is useful to consider two slightly different starting values of  $x$ , say  $x_0$  and  $x_0 + \epsilon$ , where  $\epsilon$  might typically be a roundoff error in a numerical calculation or an experimental uncertainty in a measurement. These two starting values generate two sequences of trajectories. If the trajectories are periodic, the separation between them typically grows linearly with the number of steps. In the chaotic region, the two trajectories tend to diverge from one another at an exponential rate. After  $n$  iterations of the map, the distance between the trajectories can be parametrized as follows:

$$|f^n(x_0 + \epsilon) - f^n(x_0)| \simeq \epsilon e^{n\lambda}, \quad (14)$$

where  $\lambda$  is known as the Lyapunov exponent. The Lyapunov exponent can depend on  $n$  in general. If the divergence is exponential, then  $\lambda$  will be a positive constant. Linear divergence is obtained if  $n\lambda \simeq \ln(n)$ . This equation can be solved for  $\lambda$ :

$$\lambda \simeq \frac{1}{n} \ln \left| \frac{f^n(x_0 + \epsilon) - f^n(x_0)}{\epsilon} \right| \simeq \frac{1}{n} \ln \left| \frac{df^n}{dx} \right|, \quad (15)$$

where we have approximated the difference by the derivative, assuming that  $\epsilon$  is small. One can use the chain rule for differentiation to show that

$$\frac{df^n(x_0)}{dx} = f'(x_{n-1})f'(x_{n-2})\cdots f'(x_0),$$

and hence,

$$\lambda \approx \frac{1}{n} \sum_{i=0}^{n-1} \ln |f'(x_i)|. \quad (16)$$

It is straightforward to compute  $\lambda$  as a function of  $r$  using this formula. A plot of  $\lambda$  versus  $r$  can then be compared with a bifurcation diagram: trajectories in regions where  $\lambda$  is positive exhibit extreme sensitivity to initial conditions and numerical errors, and generally correspond with chaotic regions.

\_\_\_\_\_ Program 3: <http://www.physics.buffalo.edu/phy410-505/topic4/lyapunov.cpp> \_\_\_\_\_

```
#include <cmath>
#include <fstream>
#include <iostream>
using namespace std;

double x0 = 0.5;           // starting x
double r1;                // starting r
double r2;                // ending r
int rSteps;               // steps in r
int stepsToDiscard = 20;  // discard transient behavior
int stepsToRetain = 50;   // number of steps to retain

double r;                 // current parameter value
double x;                 // current map point

void map() {
    x = 4 * r * x * (1 - x);
}

int main() {
    cout << " Logistic Map Lyapunov Exponents\n"
         << " -----\n"
         << " Enter start and end r in [0,1]: ";
    cin >> r1 >> r2;
    cout << " Enter number of steps in r: ";
    cin >> rSteps;
    ofstream dataFile("lyapunov.data");
    cout << " Initial x = " << x0 << "\tDiscarding " << stepsToDiscard
         << " steps, retaining " << stepsToRetain << " for each r" << endl;
    for (int i = 0; i <= rSteps; i++) {
        r = r1 + (r2 - r1) * i / double(rSteps);
        x = x0;
        for (int j = 0; j < stepsToDiscard; j++)
            map();
        double lambda = 0;
        for (int j = 0; j < stepsToRetain; j++) {
            map();
```

```

        lambda += log(abs(4 * r * (1 - 2 * x)));
    }
    lambda /= stepsToRetain;
    dataFile << r << '\t' << lambda << '\n';
}
cout << " Data in file lyapunov.data" << endl;
dataFile.close();
}

```

---

## The entropy of a trajectory

An interesting criterion for whether a sequence  $x_n$  is chaotic is to compute its entropy, which is a measure of disorder. Let us divide the interval  $0 \leq x \leq 1$  into  $M$  subintervals or bins, and ask for the probability  $p_m$  that a particular point in the trajectory will lie in the  $m$ -th bin. If  $n_m$  is the number of members of the sequence that lie in the  $m$ -th bin and  $n$  is the number of points in the sequence, then  $p_m = n_m/n$ . In analogy with the definition of entropy in statistical physics, the entropy of the sequence can then be defined using the formula

$$S = - \sum_{m=1}^M p_m \ln(p_m) . \quad (17)$$

It is easy to see that the entropy will be zero if all the points in the sequence lie in a single bin. This will happen if the sequence tends to a fixed point. In computing the entropy, it is therefore important to ignore the initial transient behaviour of the sequence. The entropy will attain a maximum value  $\ln(M)$  if all the intervals are equally probable. The entropy, like the Lyapunov exponent, can easily be computed as a function of  $r$ , and its behavior can be compared with the bifurcation diagram.

## Homework Problem

Use the Lyapunov program to make a plot of the exponent as a function of the parameter  $r$  in the period doubling and chaotic regions. Modify the program to calculate the entropy  $S$  and plot it in this range of  $r$ . Compare and comment on the significance of the two plots.

## References

- [1] P. Cvitanović, R. Artuso, R. Mainieri, G. Tanner, G. Vattay, N. Whelan and A. Wirzba, “Chaos: Classical and Quantum”, <http://chaosbook.org>.