Chaos and Strange Attractors

Olga Niyibizi

February 18, 2016

1 Introduction

Chaos theory encompasses a wide variety of topics. One of those topics is *attractors*, which are dynamical systems that exhibit attracting behavior. Attractors have initial states that evolve towards a single state or a set of final states. The study of attractors is one that combines fractal geometry and chaos theory. In examining how fractals and chaotic systems are related, we will be able to differentiate chaotic attractors from other attractors. First, we will examine what characteristics are needed in order for a system to be considered chaotic. Then, we will look at fractal geometry and examine what it means for an object to have a fractal dimension. Finally, we will use the characteristics of a chaotic system and fractal geometry to classify various attractors.

2 Characteristics of a Chaotic System

There are things that cannot be modeled by simple linear equations such as changes in the weather, medical epidemics, and population dynamics. At some point, there are too many variables introduced in an unpredictable way for the system to be linear. While the definition of chaos has changed in the past, today, *chaos theory* is study of nonlinear dynamic systems that are highly sensitive to initial conditions [5]. There are four characteristics that make up a chaotic system. In this section we will look at these four characteristics in depth and see how together these characteristics create a chaotic system.

2.1 Dynamical

One of the characteristics of a chaotic system is that it is dynamic. A *dynamical system* evolves over time and has two properties [6]. The first property of a dynamical system involves identifying what is evolving over time. To do this, we can create a set of variables that give a description of the system at any particular time.

Definition 1. *State variables* are the set of variables that describe the system. The set of all possible values of the state variable is called the *state space*.

The second property of a dynamical system is the evolution rule.

Definition 2. The *evolution rule* of a dynamical system is the function that determines the state that follows from the current state. The evolution rule can involve discrete or continuous time.

One example of a discrete dynamical system is the growth of a bacteria population. For our model, we will assume that the bacteria will divide into two and that each bacterium divides at the same time. Furthermore, in order for our system to be discrete, we define time so that one unit of time is the time between each division, effectively ignoring time lapse. The bacteria population growth can be modeled by the following equation:

$$b_{(t+1)} = 2b_t. \tag{1}$$

In this model b_t,the state variable, is the number of bacteria in the population at time t. Additionally, we define our state space, the allowed variables of b_t, to be the set of non-negative integers. The evolution rule for this is simply that the number of bacteria doubles over time, as given by Equation (1). The evolution rule, in addition to some initial population size, $b_0 = 1$, ensures that our system is well defined at all future times.

In addition to discrete systems, there also are continuous dynamical systems such as an undamped pendulum. An undamped pendulum is one that oscillates with no friction and thus it oscillates infinitely. This pendulum consists of a rigid rod with a ball fastened at the end and swings back and forth about the pivot point. The position of the pendulum is specified by the angle, θ, that the rod makes with a straight line pointing down. When the pendulum is straight down, $\theta = 0$. When the pendulum is pointing to the right, as in Figure 1, $0 < \theta < \pi$; and $-\pi < \theta < 0$ when the pendulum is pointing to the left.

In looking at Figure 1, there is not enough information to determine the next position of the pendulum. Our prediction of the next position of the pendulum would depend on whether the pendulum in Figure 1 was going clockwise or counterclockwise. As a result, in order to determine the next state of the pendulum, we must also consider the angular velocity, ω. For an undamped pendulum, our state variables are θ and ω where the state space is periodic in θ. A positive angular velocity denotes a counterclockwise rotation, while a negative angular velocity denotes a clockwise rotation. Since the system is two-dimensional, the current state of the system is represented by the point $(\theta(t), \omega(t))$ in the state space. The time evolution rule of an undamped pendulum is dependent by the angular velocity and rate of change of the angular velocity. Because θ and ω evolve continuously, the system is a continuous dynamical system.

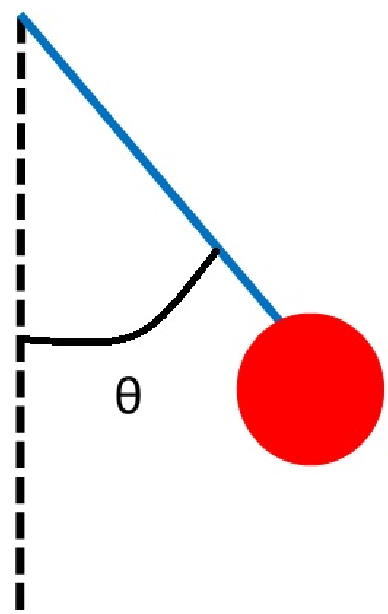

Figure 1: Undamped Pendulum *An undamped pendulum with the position of the ball given by θ.*

2.2 Deterministic

In addition to being dynamic, chaotic systems are also deterministic. Often times because the final state is not easily predictable, chaotic systems are thought to be random. However, the difference between a random system and a chaotic system is that a random system will not reproduce the same outcome. For example, a coin toss is random, but not deterministic. A coin toss could be deterministic if the conditions of one coin toss could remain unchanged in between tosses. This would mean that variable such as the air currents, the way in which the coin was flipped and the coin would have to measured and repeated in order to get the same result. As it is, some of those condition cannot be replicated. As a result, a coin toss of a standard coin generally has an equal chance of landing head or tails. Even though we know what the possible outcomes could be, we can not say with certainty what it will be. With a coin toss you know the possible outcomes, but the results of a series of five coin tosses cannot realistically be replicated. Therefore, a coin toss is considered random event. In contrast, if we look at the simple model of bacteria growth, as in Equation (1), we expect it to behave the same way, given the same initial condition, regardless of how many times we were to simulate the model. Given the same set of initial conditions, a deterministic model behaves the same way every time, and the results can be replicated.

2.3 Nonlinearity

Nonlinear systems are sometimes known to display chaotic behavior. Nonlinear systems are ones in which the output is not directly proportional to the input. Long term chaotic systems are difficult to model, since chaotic systems display nonlinear behavior. In a nonlinear system the simple changes in one part of the system can produce complex effects throughout the system. One such example of a nonlinear system is the P.F. Verhulst's model for population growth which is a slight modification from our bacteria growth model.

Let us consider a model for population growth similar Equation (1):

$$X_{n+1} = BX_n \tag{2}$$

For this model, X_{n+1} is the size of the population for next year and X_n is the population size for this year. The value of X_n, is a number between 0 and 1 that is normalized and thus allows us to compare the population of the previous year with the current year. Furthermore, B is the birthrate of the population and can be any number greater than or equal to 1. Anything less than 1 and the population will settle to 0. The problem with this model, however, is that it is unrealistic because it simulates exponential growth. If, for example, we were to apply this model to an animal population, say squirrels, then after multiple generations squirrels would cover the entire globe. We know this is not the case, so we need to modify the equation so that the population grows and shrinks accordingly. In order to balance populations Verhulst added an additional factor of $(1 - X_n)$ to Equation (2) resulting in a growth modeled by the following equation:

$$X_{(n+1)} = BX_n(1 - X_n). \tag{3}$$

Verhulst's model is also referred to as the *logistic growth model* [1].

This addition by Verhulst creates two competing factors, one that attempts to grow the population, while the other shrinks the population. For a small X_n, the factor $(1 - X_n)$ is close to 1, making the model close to the original growth equation. When is X_n increases towards 1, we see that $(1 - X_n)$ decreases towards 0 causing the right-side of the equation to shrink. Since X_n is being multiplied by itself, it produces feedback and nonlinearity. The growth of the population from in a given year depends nonlinearly on what comes from the previous year. Figure 2 shows the bifurcation diagram for the logistic growth model where X_n is the value for which the population

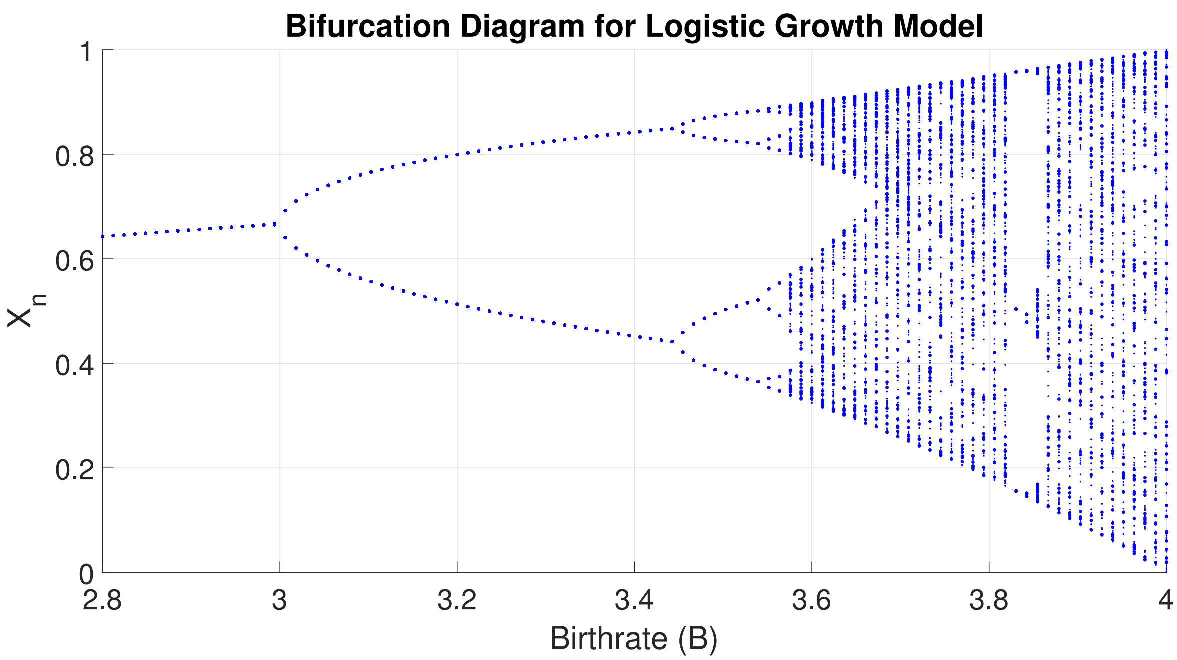

Figure 2: Bifurcation Diagram for Non-linear Growth *As the birthrate increases, the limiting value of the population also increases. As the birthrate passes 3, the line splits into populations, one high and the other low.* Appendix A

population asymptotically approaches. For $2.8 \leq B < 3$, Figure 2 shows that as the birthrate increases, the equilibrium point for the population size also increases. However, for $3 < B < 3.5$ we see that the population oscillates between two stable values. Each of the two points have to be examined, and upon further examination we notice that the points split again at about $B = 3.5$. The bifurcation diagram allows us to see where the system splits, and follow each system accordingly. The period-doubling that occurs is modeling population as it alternates between the two limiting values of the population. One cycle has a high population, leaving an overpopulated system. The overpopulated system causes much of the population to die-off in the next cycle in attempt to stabilize the system. The reduction of the population produces a large population in the next cycle. As a result, the system has an equilibrium cycle which alternates between high and low populations. The result being that the system continues to double as B increases, resulting in a chaotic system that is complex yet regular. The bacteria growth model given by Equation (1) is one that had a strictly linear relationship. In the logistic growth model, because X_n is being multiplied by itself, it is being iterated and thus the growth from year to year depend nonlinearly on what the population size was before it.

2.4 Sensitivity to Initial Conditions

Lastly, a chaotic system, in addition to being dynamical, nonlinear and deterministic; is sensitive to initial conditions. This means that a small change in the initial state can lead to drastically different final states. This further adds to the unpredictability of chaotic systems. If, for example, there is an error, even something as seemingly small as a rounding error, made in observing the present state, predictions of the instantaneous state becomes impossible for future sates. Furthermore, since our system is nonlinear, the difference between systems with different initial condition is also nonlinear. Let us use our logistic growth model, Equation (3), with $B = 4$ to consider two initial values, $X_0 = .506$ and $X_0 = .506127$.

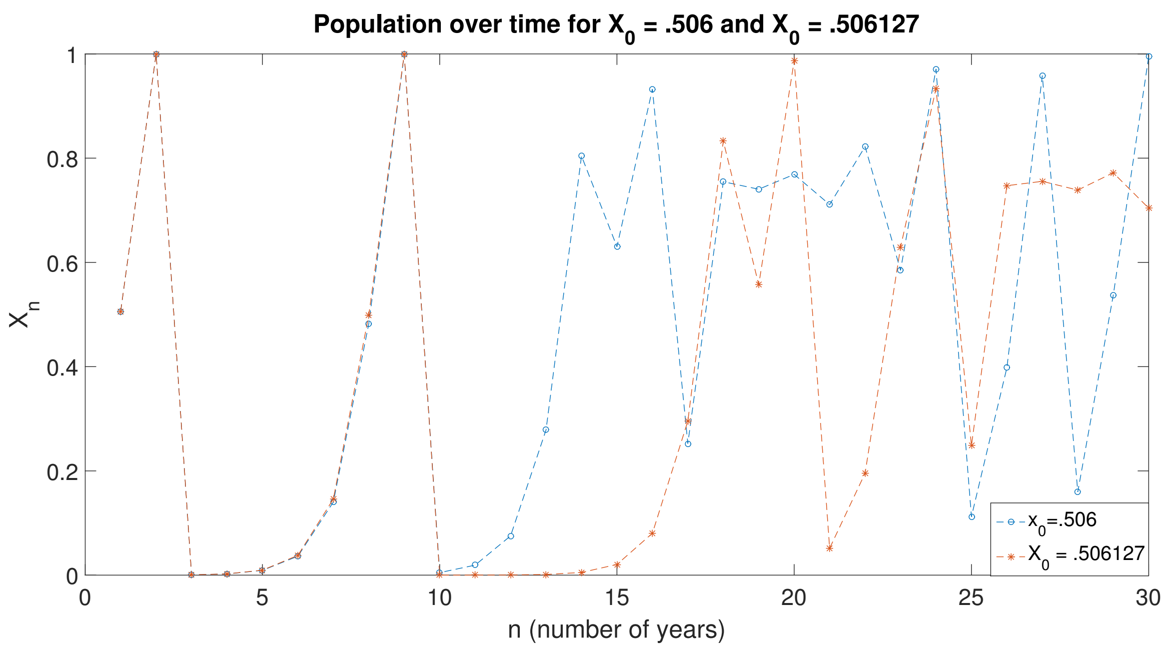

Figure 3: Population over time for the logistic growth model $B = 4$ for $X_0 = .506$ and $X_0 = .506127$ *The open circle denotes* $X_0 = .506$ *and the asterisk denotes* $X_0 = .506127$*. This plot was generated using MATLAB. The source code for this graph and the values for Table 1 can be found in Appendix A*

Figure 3 shows the change of the population over 30 years. Both systems rise and fall as we would expect them to, but because of the initial conditions, the populations vary from one another, particularly between $10 < N < 25$. Since the system is nonlinear the initial difference between the two systems also changes over the years. Table 1 shows that even though the initial difference between the values is 0.000127, it changes over time and with the final difference in populations being 0.290539. The rounding of the initial population resulted in two very different systems.

Table 1: Population size at various years for different initial condition

n	0	1	2	3	28	29	30
$X_{1(0)} = .506$	0.506000	0.999856	0.000576	0.002302	0.159626	0.536581	0.999465
$X_{2(0)} = .506127$	0.506127	0.999849	0.000600	0.002400	0.738757	0.771979	0.704107
$\lvert X_{1(n)} - X_{2(n)}\rvert$	0.000127	0.000006	0.000024	0.000098	0.579131	0.235398	0.290539

3 Fractal Dimensions

The study of attractors combines chaos theory and fractal geometry. Fractal geometry is the geometry of chaos since it can be used to visually depict the behavior of chaotic systems. Fractals allow us to measure qualities that have no clear definition such as brokenness or irregularity an object [5]. In addition to chaotic characteristics, one thing that sets a chaotic attractors apart from other attractor are their fractal properties.

3.1 Fractal Geometry

A *fractal* is a geometric object that is infinitely complex and self-similar on all scales. A simple figure can become complex when specific rules are imposed on integrating figures in a feedback loop. One example of a fractal is the Sierpinski triangle shown in Figure 4. Fractals are defined by the following properties [2]:

1. fractals are defined recursively,
2. fractals have a fine structure,
3. fractals have some form of self-similarity,
4. fractals are too irregular to be described in traditional geometric language and,
5. fractals have fractal dimension.

The first property defines the repeating pattern that is visible in fractals. The recursive definition of a figure gives an initial stage and a recursive rule that determines how the next stage is created. For the Sierpinski triangle in Figure 4, we start with an equilateral triangle. Then for each triangle, we join the midpoints of of the sides and remove the triangle that is formed in the middle as a result. This recursive rule is one that can be applied in finer detail. As a result, if we were to magnify the structure, we could see more detail at a smaller scale. This leads to the third property of fractals. If a structure contains an exact replica of the whole structure on a smaller scale it is considered to be self-similar. In stage 3 of Figure 4 any of the three triangular corners could be considered self-similar. Of all of the properties of a fractal, the one we are most interested in is the the last property. There are many different ways to compute a dimension, the simplest of which for self-similar figures is called the similarity dimension. Using the similarity dimension, we will be able to determine if a figure is a fractal.

Definition 3. A figure is said to be *fractal* if its dimension is non-integer.

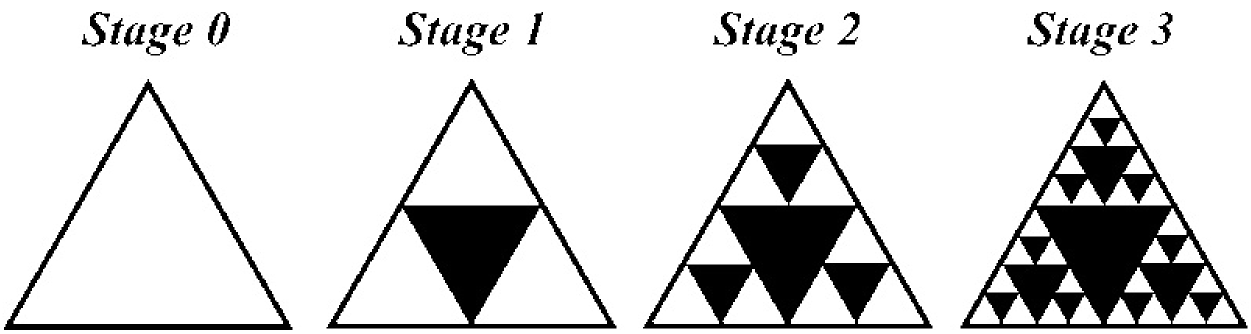

Figure 4: The Sierpinski Triangle *The stages of the Sierpinski triangle.* [4]

Similarity dimension allows us to look parts of figures that are reduced copies of the whole figure. For example, if we reduce a single-segmented line by a factor of 3, we are left with 3 self-similar lines. If, however we were to reduce a single-segmented line by a factor of 4, we get 4 similar lines. In fact, if we reduce a line segment by a factor of r were are left with n similar pieces, thus $n = r$. The same can not be said about a square. If we reduce a square by a factor of 3, the result is 9 self-similar pieces, and reducing a square by a factor of 4 yields 16 pieces. Here we see that reducing the square by a factor of r results in n^2 self-similar pieces, thus $n = e^2$. Similarly, a cube is a non fractal figure because we get n^3 self-similar pieces if we reduce it by a factor of r, here $n = r^3$. This is why line segments are one-dimensional, squares are two-dimensional, and cubes are three-dimensional. Furthermore, since the dimension is an integer, we know that the are not fractals. If we let r be the reduction factor, n be the resulting number of self-similar pieces, and D be the dimension, then the relationship can be described by the following equation [2]:

$$r^D = n. \tag{4}$$

If we take the logarithm of both sides, we can use the properties of logs to solve for D:

$$\log\left(r^D\right) = \log\left(n\right), \tag{5}$$

which reduces to

$$D * \log\left(r\right) = \log\left(n\right), \tag{6}$$

and finally

$$D = \frac{\log\left(n\right)}{\log\left(r\right)}. \tag{7}$$

We can use this equation to determine how the Sierpinski triangle has a fractal dimension, and thus is a fractal. If we look at Figure 4, we notice that stage 1 is made up of 3 smaller triangles from stage 0. In the next iteration, stage 2 is made of 9 triangles from stage 0, and 3 triangles from stage 2. So it stands that each time we iterate a function, we are left with 3 times as many triangles as before, thus $n = 3$. Also given that the triangle is equilateral, and looking between stage 0 and stage 1, we see that the base is half of the original. In going from stage 0 to stage 1, we have $r = 2$ and $n = 3$. We can then calculate the dimension using Equation (7):

$$D = \frac{\log\left(3\right)}{\log\left(2\right)} \approx 1.585$$

Similarly, if we go from stage 1 to stage 2, the dimension is the same because the n remains equal to 3, and $r = 2$.The result is a non-integer, therefore we know that the Sierpinski triangle is a fractal. Though this shows that the similarity dimension is helpful in calculating the dimension of a figure, it only works for a small subset of self-similar figures that are strictly self-similar.

Definition 4. A figure is *strictly self-similar* if is self similar at every point.

This means that a figure can be broken into a number of disjointed pieces, each of which is an exact copy of the whole figure [2]. Looking at the Sierpinski triangle, if we isolate one of the the three triangles we see that it is a smaller copy of the whole figure. This not only true of the three large triangles in the whole figure, but it is also true of all the triangles in the figure. Attractors are not structures that are strictly self-similar. As a result, we must formulate a different way to measure the dimension of an attractor.

3.2 Lyapunov Spectrum

The Lyapunov exponent of a dynamical system measures the sensitivity of the system to changes in the initial conditions. Essentially, if two orbits start close to each other in a chaotic system, they will move away from each other at an exponential rate. That is,

$$\frac{d}{d_0} = e^{\lambda(t-t_0)} \tag{8}$$

were d_0 is the initial displacement between a starting point and a nearby point at initial time, t_0. Furthermore, d is the displacement at time t where $t > t_0$ [3]. Solving the above equation for the Lyapunov exponent, λ, we get

$$\lambda = \frac{1}{t-t_0} \ln \left| \frac{d}{d_0} \right| . \tag{9}$$

The equation above gives us a λ value from two specific points. In order to determine the Lyapunov exponent of the whole system we must take the average of many pairs of points.

Definition 5. If the displacement between the i-th point and a neighboring point at time t_i, is d_i and the initial displacement between the two points is d_{0i} at time t_{0i}, then the *Lyapunov exponent* for the system is defined as

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \frac{1}{t_i - t_{0i}} \ln \left| \frac{d_i}{d_{0i}} \right| . \tag{10}$$

Dynamical system have orbits have orbits that fold and stretch into one another, meaning that they diverge and merge, therefore the equation to determine the Lyapunov exponent must be adjusted. When considering two orbits, we only want to look at the initial divergence and stop before folding occurs. This new Lyapunov equation follows one reference orbit and its nearby orbits and calculates their average logarithmic rate of separation.

To begin calculating the Lyapunov exponent of an attractor, we first select a random point. This first point will be the reference point on the reference orbit. Then we pick a random point on a nearby orbit, and calculate the distance, d_0, between the two orbits. Then we move forward in the same direction on both orbits. In the case of a discrete system we advance by one iteration, and in the continuous system we advance by a some time interval, t_{step}. We then determine this new distance, d_1. Then a new point is chosen on a nearby orbit that is in the same direction, but displaced from, the reference orbit of d_0. This process is repeated n times to find an approximation of λ [3]. Figure 5 gives a visual depiction for calculating the Lyapunov exponent of a system. The result is the following equation for a continuous system:

$$\lambda = \frac{1}{n(t_{step})} \sum_{i=1}^{n} \ln \left| \frac{d_i}{d_0} \right|, \tag{11}$$

and similarly a discrete system:

$$\lambda = \frac{1}{n} \sum_{i=1}^{n} \log_2 \left| \frac{d_i}{d_0} \right|. \tag{12}$$

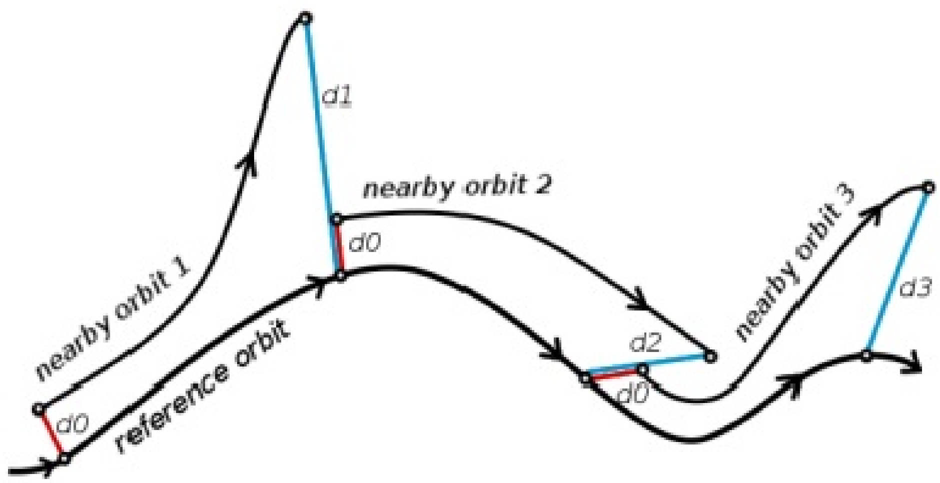

Figure 5: Steps for calculating a Lyapunov exponent *A reference orbit is chosen and nearby orbits are used to calculate the local Lyapunov exponent.* [3]

The Lyapunov spectrum for a system is determined by repeating the above process for each dimension of the system. Thus a 2-dimensional system will have 2 Lyapunov exponents, one in the x-axis and one in the y-axis. So it follows that an n-dimensional system will have n Lyapunov exponents. The set of all the Lyapunov exponents, $\{\lambda_1, \lambda_2, ..., \lambda_n\}$, is the Lyapunov spectrum where the exponents are arranged from largest to smallest. The largest Lyapunov exponent is used then used to determine if a system is chaotic. If the largest Lyapunov exponent is larger than zero, then the trajectories diverge exponential, thus indicating a chaotic system. If the exponent is equal to zero, the system generally has a periodic motion. If the largest exponent is less than zero, than the system converges to a fixed point. In the even that the exponent is less than or equal to zero, the system is non-chaotic.

$$\lambda_1 > 0 \Leftrightarrow chaotic \tag{13}$$

$$\lambda_1 \leq 0 \Leftrightarrow non - chaotic \tag{14}$$

For a continuous 3-dimensional system attractors can be classified in various ways based on the pattern of the Lyapunov spectrum. For example, $(+, 0, -)$ denotes a strange attractor, $(0, 0, -)$ is a two-tours attractor, $(-, -, -)$ is a fixed point attractor, and $(0, -, -)$ is a limit cycle attractor [3].

Recall that we are trying to develop a method to calculate the dimension of an attractor. With the Lyapunov exponents, we can use the Kaplan-Yorke dimension.

Definition 6. If $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$ are the Lyapunov exponents for a dynamical system in $\mathbb{R}^n$ and j is the largest integer for which $\lambda_1 + \lambda_2 + ... + \lambda_j \geq 0$, then the *Kaplan-Yorke dimension* is given by

$$D_{KY} = j + \frac{\lambda_1 + \lambda_2 + ... + \lambda_j}{|\lambda_{j+1}|}. \tag{15}$$

With the Kaplan-Yorke dimension, if D_{KY} is a non-integer, the system is a fractal. However, it is possible for a set to have integer dimensions and still have fractal properties. The Lyapunov exponent and the Kaplan-Yorke dimension can be used to categorize various attractors by determining if they are chaotic and fractal in nature.

4 Attractors

With the appropriate tools we can now examine various attractors. In order to define what an attractor is, we must first define an attracting set [3].

Definition 7. An *attracting set*, A, is a subset of $\mathbb{R}^n$ that the initial points of the system evolve towards. Furthermore, an attracting set must satisfy two conditions:

1. The basin of attraction B(A), which consists of all points whose orbits converge to A, has a positive measure.

2. For any closed subset $A' \subset A$, the set difference B(A)\B(A') has to also be of positive measure.

There are further technicalities to these definition that are outside of the scope of this paper. It is important to note however, that the attracting set cannot be a single point or a set of discontinuous points. In addition to the previous two conditions, an attracting set must satisfy an additional condition to be considered an attractor.

Definition 8. An attracting set that contains a dense orbit is an *attractor*.

A dense orbit on an attracting set means that there is at least one orbit that passes through, or gets close to, every point of the attracting set. In the following sections we will examine various attractors and determine which of them are strange chaotic attractors.

Definition 9. An attractor is *strange* if its attracting set is fractal in nature.

We will use the Lyapunov exponent to determine if an attractors is chaotic, and we will use the Kaplan-Yorke dimension, in addition to other knowledge we have about attractors, to determine if an attractor is chaotic.

4.1 Nonstrange Nonchaotic Attractors

For our first system let us consider the Van der Pol oscillator. Van der Pol first discovered stable oscillations in electrical circuits employing vacuum tubes [7]. The Van der Pol oscillator can be defined by the following set of differential equations:

$$\frac{dx}{dt} = y \tag{16}$$

$$\frac{dy}{dt} = \frac{-1}{cd}(x + by^3 - ay). \tag{17}$$

Here x is the current, y is the voltage, c is the capacitance, d is the the inductance, and a and b are constants. Figure 6 shows the system when $a = 1, b = 1, c = 1$, and $d = 1$. This system is considered a limit cycle attractor, with the trajectories that spiral towards a ring-like attracting set. The MATLAB code used to generate the plot for the limit cycle attractor and its corresponding Lyapunov values can be found in Appendix C and Appendix E repectively.

We can determine the Lyapunov spectrum by plotting Equation (11) as n increases , as shown in Figure 7. Figure 7 shows the Lyapunov exponents converge, resulting in $\lambda_1 = 0$ and $\lambda_2 = -1.06$. The Lyapunov exponents follow a $(0, -)$ pattern which is also indicative of a limit cycle. Additionally, because there are no positive Lyapunov exponents, we can determine that the system is nonchaotic. Furthermore, we can use the Kaplan-Yorke dimension given by Equation (15) to determine that the attractor does not have a fractal nature. Here,

$$D_{KY} = 1 + \frac{0}{|-1.06|} = 1.$$

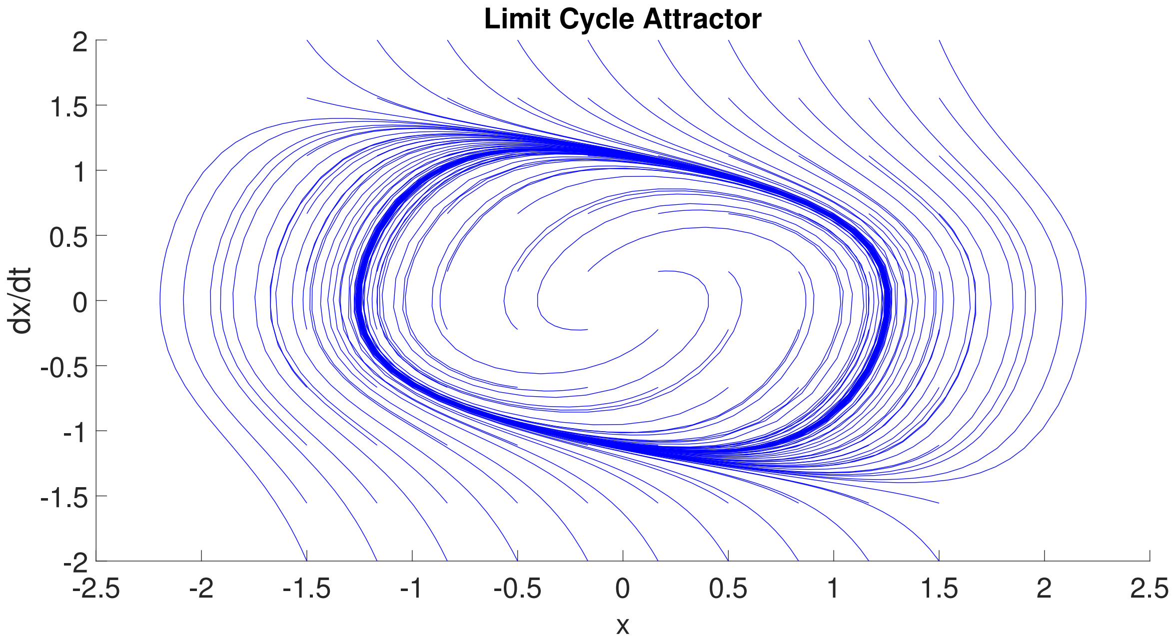

Figure 6: Phase plane for Van der Pol Oscillator *The phase plane shows the rate of change in currents as a function of the current.*

Thus we know that this system is a nonstrange, nonchaotic attractor. Similar attractors, such as fixed point attractors, will also be nonstrange nonchaotic attractors. Nonstrange nonchaotic attractors are predictable and have orbits that tend to one attracting set. As a result, if we change the initial conditions of our system, the trajectory would orbit toward the same attracting set.

4.2 Strange Nonchaotic Attractors

There is a common misconception that strange attractors and chaotic attractors are the same. However, we know from our earlier definition of strange that, this is not the case. Chaos focuses on the loss of predictability, while strangeness describes unfamiliar geometric structures [3]. In a strange system, orbits in an attractor are non-periodic, meaning that any point in the attracting set is never visited more than once, and results in a set that is fractal in nature. While strange is often used as a name for attractors that are chaotic, strangeness is not dependent on the existence of chaos. Therefore, there exist strange attractors that are also nonchaotic.

Greborgi, Ott, Pelkian and Yorke were the first to show that attractors could be strange and nonchaotic. The GOPY model, unofficially named after the authors, is a model that was created

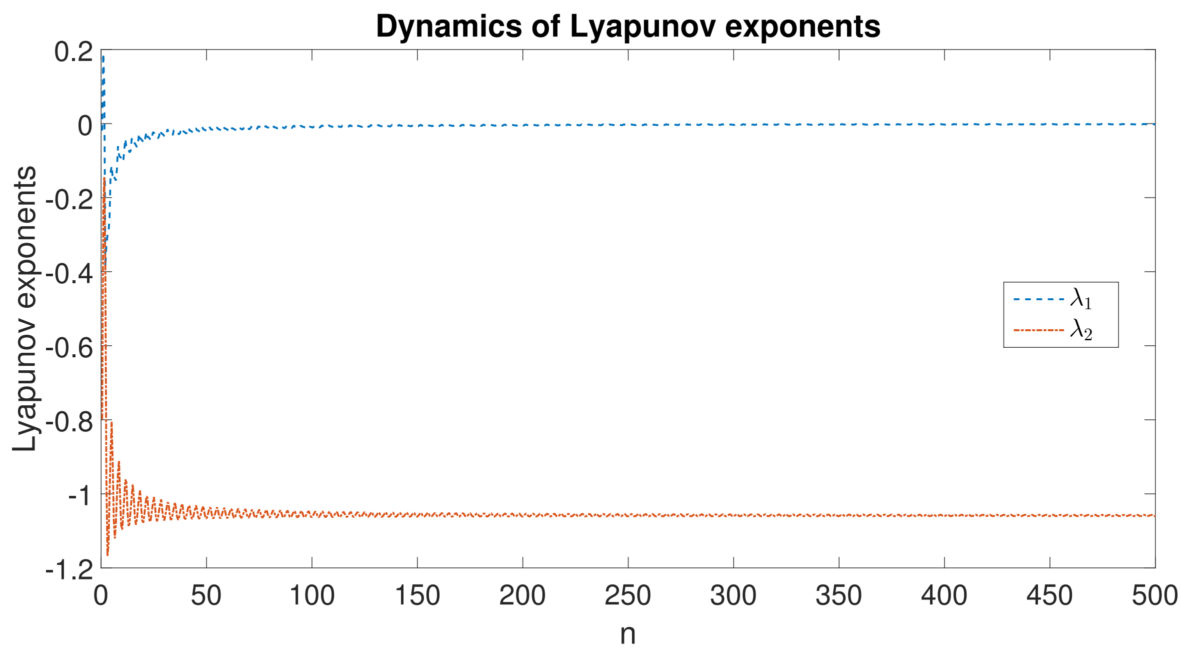

Figure 7: Lyapunov Exponents for Van der Pol Oscillator *As n increases, the two-dimensional system yields two Lyapunov exponents;* $\lambda_1 = 0$ *and* $\lambda_2 = -1.06$*.*

to study strange nonchaotic attractors. It is a discrete system defined by the following equation:

$$x_{n+1} = 2\sigma \tanh(x_n)\cos(2\pi\theta_n) \tag{18}$$

$$\theta_{n+1} = (\theta_n + \omega) \mod 1 \tag{19}$$

Figure 8 shows the attractor for $\sigma = 1.5$ and $\omega = \frac{\sqrt{5}-1}{2}$ with the initial conditions of $x_0 = 1$ and $\theta_0 = 0$. Note that ω is the golden ratio which is an irrational number. Calculating the Lyapunov spectrum of the attractor we get that $\lambda_1 = 0$ and $\lambda_2 = -1.53$ [3]. Like the limit cycle, the GOPY model has no positive exponents, so it is not chaotic.

When computing the Kaplan-Yorke dimension we get that $D_{KY} = 1$. However, despite the non-fractal appearance of the structure and the integer dimension, the system still has a fractal nature. This is because upon closer examination, we notice that the system is actually non-periodic. We know that system is non-periodic because θ_{n+1} is the modulo 1 of the sum of θ_n and an irrational number, thus producing an irrational number. As the function is iterated, θ_{n+1} will always result in an irrational number, resulting in a system that is non-periodic overall. This will hold true for any ω that is irrational. Furthermore, if we look closely at the graph, we also notice some discontinuity. Furthermore, while Figure 8 seems to display a curve, it is actually a collection of iterated values

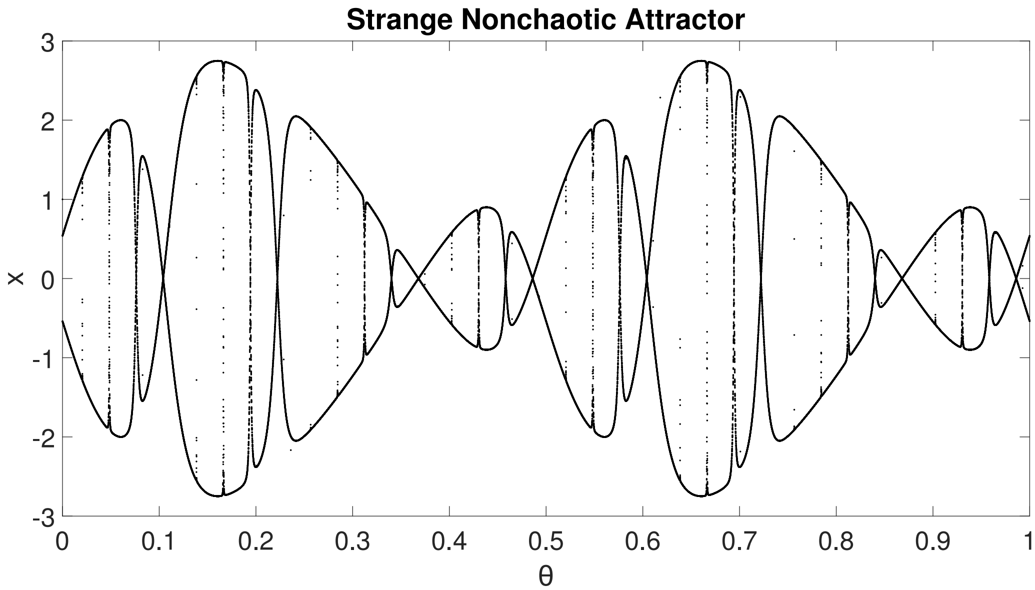

Figure 8: Strange nonchaotic attractor *One orbit of the discrete GOPY model where* $x_0 = 1$ *and* $\theta_0 = 0$ Appendix F

of the discrete functions. Taking that into consideration, we notice that discontinuity happening at irregular intervals. Consider the points on the attractor at $\theta = .25$ and $\theta = .75$. The cos for those θ values are 0, so there must exits the points ($\theta = .25 + \omega$, $x = 0$) and ($\theta = .75 + \omega$, $x = 0$) on the attractor. Thus $x = 0$, $\theta \in [0, 1]$ is not the attractor, but there is a dense set of points in the attractor that are also in $x = 0$, $\theta \in [0, 1]$. These sets of points are fractal in nature and have non-integer dimension. This, in addition to the system being non-periodic, makes the GOPY model a fractal structure. Thus the GOPY model is an example of a system that is strange but not chaotic.

4.3 Chaotic Attractors

It is important to note while a system can be strange but nonchaotic that the converse cannot be true. This is to say a system cannot be nonstrange but chaotic. This is because of the fractal nature of non-periodic orbits. In a chaotic attractor, the orbits are non-periodic so if the system is chaotic it is also strange [3].

One example of a chaotic attractor is the Lorenz system. Edward Lorenz developed a model for atmospheric convection using the equations [3]:

$$\frac{dx}{dt} = \sigma(y - x) \tag{20}$$

$$\frac{dy}{dt} = x(\rho - z) - y \tag{21}$$

$$\frac{dy}{dt} = xy - \beta z, \tag{22}$$

where σ, ρ and β are parameters. The plot with the parameters $\rho = 28, \sigma = 10,\ \beta = \frac{8}{3}$ and an initial vector of (10, 10, 10) can be seen in Figure 9. As the orbits converge and diverge, they create a butterfly shape that has a fractal nature. We can also generate a plot, as in Figure 10, to determine the Lyapunov exponents. Since the Lorenz attractor is a 3-dimensional system, we get 3 Lyapunov exponents. For this attractor the Lyapunov spectrum,$\{\lambda_1, \lambda_2, \lambda_3\}$, is the set $\{.81, .02, -14.50\}$. We determine that the attractor is strange by calculating the D_{KY} for this Lorenz attractor as follows:

$$D_{KY} = 2 + \frac{.81 + .02}{|-14.50|} = 2.06$$

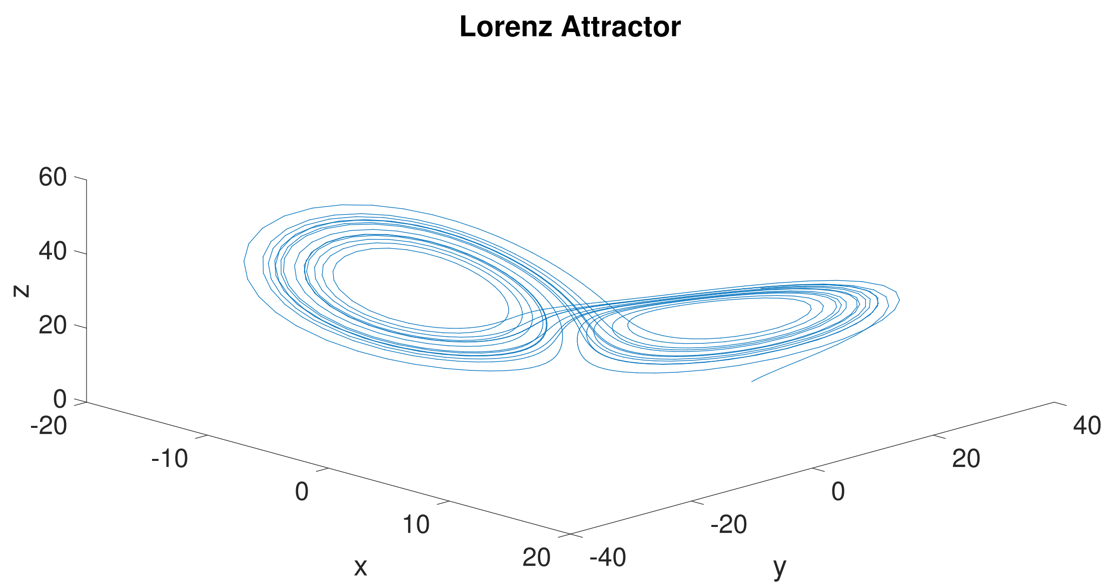

Figure 9: Lorenz attractor *The single orbit of the Lorenz attractor with an initial vector of (10 10 10).* Appendix D

Here the Kaplan-Yorke dimension is a non-integer, meaning that the Lorenz attractor is a fractal and thus is strange. We also know that this attractor is strange since the spectrum follows the $(+, 0, -)$ pattern previously discussed in Section 3.2. Though $_2 \approx .02$, the spectrum still fits the overall pattern for a strange attractor. The Lyapunov exponent λ_1 is greater than 1, indicating a

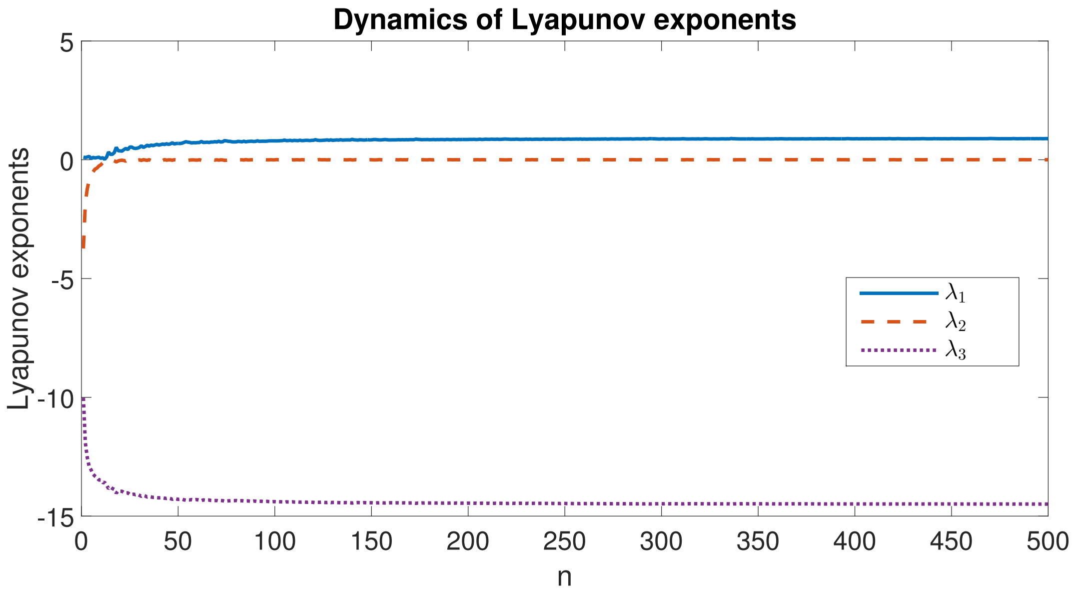

Figure 10: Lorenz attractor *The corresponding Lyapunov exponents for the three-dimensional Lorenz attractor are* $\lambda_1 = 0.81$, $\lambda_2 \approx .02$ *and* $\lambda_3 = -14.50$. Appendix E

chaotic system. If we consider the characteristics of a chaotic system as outlined in Section 1 we have further proof that the system is chaotic.

The Lorenz attractor is a system of differential equations modeling the rate of change of position over time. Our state space is defined by the variables x, y, and z over time t. Our evolution rule is given by the systems of differential equations, in addition to our initial vector (10, 10, 10). The system is three-dimensional and x, y, and z evolve continuously. As a result the Lorenz attractor is a continuous dynamical system. Given the same initial conditions, the attractor will look the same. Given that there are no random numbers or probabilities involved, we know that our system is deterministic.

Like the logistic growth model, the Lorenz attractor is sensitive to initial conditions. If we change our initial vector from (10, 10, 10) to (9.99, 9.99, 9.99) we get two different systems. If we look at the mapping of the different initial conditions for Equation (20) we see how the systems differ. We can also use Figure 11 to see that the system is nonlinear. In a linear system, we would expect the graph to shift down by 0.01 along the y-axis. Instead, we have two different trajectories of the system, which indicates that our system is nonlinear. Given that the Lorenz attractor has the four characteristics of a chaotic system and has $\lambda_1 > 0$, we know it is chaotic. Given that a chaotic system must also be strange, we know that the attractor also has a fractal nature, which

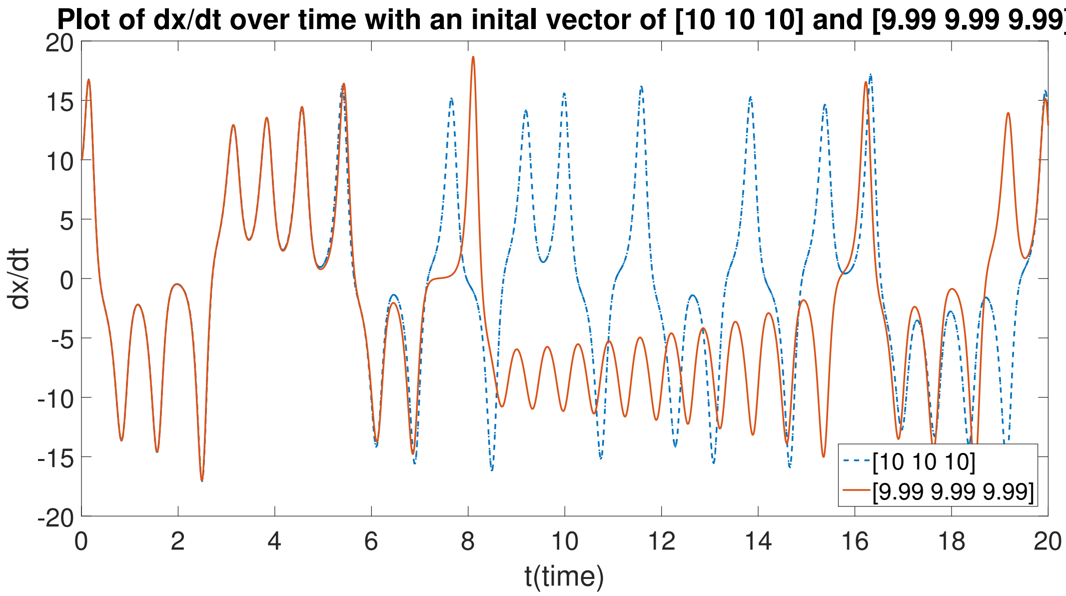

Figure 11: Plot of $\frac{dx}{dt}$ for the Lorenz Function *This graph shows the rate of change in the x direction over time for different initial conditions. The dotted line is the initial vector (10,10,10), the solid line is the initial vector (9.99, 9.99, 9.99)* See Appendix D for MATLAB source code for the plot of the graph.

confirms our results from the Kaplan-Yorke dimension. Thus, the Lorenz attractor is a strange chaotic attractor.

5 Conclusion

Looking at attractors allows one to closely examine the relationship between fractals and chaotic systems. There is a common misconception in which people think that strange attractors are also chaotic attractors but we were able to see that this is not always the case.

By using the Lyapunov exponent we are able to identify if a system is chaotic or nonchaotic. In understanding the characteristic of chaotic systems, we had additional information to support the largest Lyapunov exponent. We can also use the Lyapunov spectrum to determine the dimensions of a given system. Using the Lyapunov spectrum we are able to calculate the Kaplan-Yorke dimension for a system. While a system with fractal dimensions always indicated a fractal structure, there can be fractals that have an integer D_{KY}. We are then able to build up from a nonstrange nonchaotic attractor to a strange chaotic attractor. By studying properties of various attractors, we have been able to see how chaotic theory and fractal geometries are related. Understanding the two helped separate chaotic attractors from other attractors.

Acknowledgement

I want to thank Dr. Elizabeth Wolf for inspiring this topic and for her help with population growth modeling. Also many thanks to my adviser Dr. Ian Bentley for all his time and guidance in helping me work through this topic.

References

[1] Briggs, John, and F. David Peat,*Turbulent Mirror: An Illustrated Guide to Chaos Theory and the Science of Wholeness*, New York: Harper Row, 1989.

[2] Bedford, Crayton W.*Introduction to Fractals and Chaos: Mathematics and Meaning.* Andover, MA: Venture Pub., 1998.

[3] Taylor, Robert ”Attractors: Nonstrange to Chaotic”*SIAM Undergraduate Research Online* Volume 4, July 21 2011 `<https://www.siam.org/students/siuro/vol4/S01079.pdf>`

[4] ”CK-Foundation” *Self-Similarity* `<https://dr282zn36sxxg.cloudfront.net/datastreams>`

[5] Sardar, Ziauddin, Iwona Abrams, and Ziauddin Sardar *Introducing Chaos*, Icon Books, 2009.

[6] Nykamp DQ *Math Insight* `<http://mathinsight.org/dynamical_system_idea>`

[7] Abraham, Abraham, Shaw *Basic Principles of Dynamical Systems* Plenum Press, New York, 1992

[8] MATLAB version 8.10.0. Natick, Massachusetts: The MathWorks Inc., 2014.

A Appendix

All codes executed using MATLAB [8]

```
function mat = yue_bifur_(x0,a0,a1,N,L,p_siz)
% --------------------------------------------------
% By Yue Wu
% ECE Dept, Tufts University
% 03/03/2010
% All copyrights reserved
% --------------------------------------------------
% Function yue_bifur: plots 1D bifurcation figure
% Input: fun = some function @(x,para)
%        x0 = initial value of x
%        a0 = the start value of parameter a
%        a1 = the end value of parameter a
%         N = the number of intervals for parameter
%         L = the number of iterations for each initial pair
%               of (x0,parameter a)
%        p_siz = the marker size, default 1
% Output: mat = bifucation matrix with size N by L
%               which stores a length L sequence
%               for each pair of (x0,parameter a)
%
% --------------------------------------------------
% Demo:
 fun = @(x,r) r*x*(1-x);
% x0 = .4; a0 = 0; a1 = 4; N = 100; L = 50;
% mat = yue_bifur(fun,x0,a0,a1,N,L);
% ---------------------------------------------------
```

```
% default settings
if ~exist('p_siz','var')
    p_siz = 1;
end

% initialization
mat = zeros(N,L);
a = linspace(a0,a1,N);

% main loop
format long
for i = 1:N
    ca = a(i); % pick one parameter value at each time
    for j = 1:L % generate a sequence with length L
        if j == 1
            pre = x0; % assign initial value
            for k = 1:500 % throw out bad data
               nxt = fun(pre,ca);
               pre = nxt;
            end
        end
        nxt = fun(pre,ca); % generate sequence
        mat(i,j) = nxt; % store in mat
        pre = nxt; % use latest value as the initial value for the next round
    end
end

% plot
dcolor = [0,0,1]; % Marker color setting: blue
[r,c] = meshgrid(1:L,a); % associated cooridate data
```

```
surf(r,c,mat,'Marker','.','MarkerSize',p_siz,'FaceColor',...
...'None','MarkerEdgeColor', dcolor,'EdgeColor','None')
view([90,0,0]) % change camera direction
ylim([a0,a1]) % fit to data
```

```
%Olga Niyibizi
%The following code is used to invoke the function(yue_bifur) that generates a 1-D
%bifurcation plot.

x0 = .4; a0 = 2.8; a1 = 4; N = 100; L = 50; %initial conditions
mat = yue_bifur(x0,a0,a1,N,L,10);
%In order to create a dense and through graph, additional initial values are added
 x1=.5;
hold on
 mat1 = yue_bifur(x1,a0,a1,N,L,5);
 x2=.8
mat2 = yue_bifur(x2,a0,a1,N,L,5);
```

B

```
%Olga Niyibizi
%This code graphs population using the logistic growth model

clc
clear
years = 30; %number of years mapped
b = 4 %birthrate
format long
%Initialize vectors
outone = zeros(1,years);
```

```
outtwo = zeros(1,years);
x = zeros(1,years);
%initial conditions
x1 = .506;
x2 = .506127;

%Main loop to determine the population size at each year
for i=1:years
    outone(i) = x1;
    outtwo(i) = x2;
    x1 = b*x1*(1-x1);
    x2 = b*(x2)*(1-x2);
    x(i)=i;
end

diff = abs(outone - outtwo)%calculate the difference of each year
%Plot the different years
plot(x,outone,'--o');
hold on
plot(x, outtwo,'--*');
```

C

```
function dx = limit(t,x)
%Olga Niyibizi
%This function is the Right-Hand-Side of the equation for the Van der Pol oscillator

%Initial parameters
a=1;
b=1;
c=1;
```

```
d=1;

dx = [0; 0]; %initial array

%System of equations for Van der Pol oscillators

dx(1) = x(2);

dx(2) = (-1/(c*d))*(x(1)+b*x(2)^3-a*x(2));


%Olga Niyibizi

%This code plots various trajectories of the solution for the systems of

%equations for Van der Pol oscillator

%It invokes the function for the solution to the equation


clc

clear

%Initial vectors for various initial conditions

lenx = linspace(-1.5,1.5,10);

leny = linspace(-2,2,10);


%Main loop that solves the differential equations and plots the

%trajectories and various initial conditions

for i = 1:10

    for j = 1:10

    %ODE45 is a solver that solves the differential equation given the RHS of the equation

        [t, y] = ode45('limit',[0,100],[lenx(i);leny(j)]);

        hold on

        plot(y(:,1),y(:,2), 'b')

    end

end
```

D

```
function dx=lor(t,x)
%Olga Niyibizi
%This is a function for the RHS of the Lorenz Attractor

% Initial Parameters
s=10;
r=28;
b=8/3;

dx=[0; 0; 0];%Initial Vector

% System of equations for the Lorenz Attractor
dx(1) = s*(x(2)-x(1));
dx(2) = x(1)*(r-x(3))-x(2);
dx(3) = x(1)*x(2)-b*x(3);

%Olga Niyibizi
%This code creates a 3-D plot of the Lorenz attractor. It also plots the
%dx/dy function of the code in 2-D

[t, y] = ode45('lor', [0 20], [10 10 10]);
plot3(y(:,1),y(:,2),y(:,3))
view(45,30) %Rotates view of the graph

%Lorenz attractor at different initial vector and the plot of the graph
%against time
[t1, y1] = ode45('lor', [0 20], [9.99 9.99 9.99]);
plot(t(:,1), y(:,1),'.-');
```

```
hold on
plot(t1(:,1), y1(:,1));
```

E

```
%Olga Niyibizi
%Code that executes the function that calculates the local
%Lyapunov exponents and graphs it

    [T,Res]=lyapunov(2,@vdp,@ode45,0,0.5,500,[1 1],10);

plot(T,Res);
title('Dynamics of Lyapunov exponents');
xlabel('Time'); ylabel('Lyapunov exponents');
h = legend('$\lambda_1$','$\lambda_2$');
set(h,'Interpreter','latex');

function f=vdp(t,X)
%Olga Niyibizi
%RHS for Van der Pol equations
%We use the Jacobian here because output must be a vector

%Inital parameters
a=1;
b=1;
c=1;
d=1;

x=X(1);
y=X(2);
%Idenity matrix
```

```
Y= [X(3), X(5);
    X(4), X(6)];

%RHS of Van der Pol
dx=y;
dy=(-1/(c*d))*(x+(b*y^3)-(a*y));

dx1=[dx;dy];

%Jacobian of the solution

 Jac=[0, 1;
      ((-1/(c*d))), -((3*c*b*y^2)-a)/(c*d)];
 F=Jac*Y;

f=[dx1; F(:)];

%Olga Niyibizi
%Code that executes the function that calulates the local
%Lyapunov exponents and graphs them
[T,Res]=lyapunov(3,@lorenz_ext,@ode45,0,1,500,[1 1 1],10);

plot(T,Res);
title('Dynamics of Lyapunov exponents');
xlabel('n'); ylabel('Lyapunov exponents');
h = legend('$\lambda_1$','$\lambda_2$','$\lambda_3$');
set(h,'Interpreter','latex');

function f=lorenz_ext(t,X)
%Olga Niyibizi
```

```
%RHS for Lorenz equations
%We use the Jacobian here because output must be a vector

%Inital parameters
s = 10;
r = 28;
b = 8/3;

x=X(1);
y=X(2);
z=X(3);
%Idenity matrix
Y=[X(4), X(7), X(10);
    X(5), X(8), X(11);
    X(6), X(9), X(12)];

%RHS of Lorenz equation
dx=s*(y-x);
dy=-x*z+r*x-y;
dz=x*y-b*z;

dx1=[dx;dy;dz];

%Jacobian of the solution
 Jac=[-s, s,     0;
     r-z,    -1,    -x;
         y,     x, -b];
 F=Jac*Y;

f=[dx1; F(:)];
```

```
function [Texp,Lexp]=lyapunov(n,rhs_ext_fcn,fcn_integrator,tstart,stept,tend,ystart,ioutp);
%
%    Lyapunov exponent calcullation for ODE-system.
%
%    The alogrithm employed in this m-file for determining Lyapunov
%    exponents was proposed in
%
%         A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano,
%        "Determining Lyapunov Exponents from a Time Series," Physica D,
%        Vol. 16, pp. 285-317, 1985.
%
%    For integrating ODE system can be used any MATLAB ODE-suite methods.
% This function is a part of MATDS program - toolbox for dynamical system investigation
%    See:    http://www.math.rsu.ru/mexmat/kvm/matds/
%
%    Input parameters:
%      n - number of equation
%      rhs_ext_fcn - handle of function with right hand side of extended ODE-system.
%              This function must include RHS of ODE-system coupled with
%              variational equation (n items of linearized systems, see Example).
%      fcn_integrator - handle of ODE integrator function, for example: @ode45
%      tstart - start values of independent value (time t)
%      stept - step on t-variable for Gram-Schmidt renormalization procedure.
%      tend - finish value of time
%      ystart - start point of trajectory of ODE system.
%      ioutp - step of print to MATLAB main window. ioutp==0 - no print,
%              if ioutp>0 then each ioutp-th point will be print.
%
```

```
%    Output parameters:
%      Texp - time values
%      Lexp - Lyapunov exponents to each time value.
%
%    Users have to write their own ODE functions for their specified
%    systems and use handle of this function as rhs_ext_fcn - parameter.
%
% --------------------------------------------------------------------
% Copyright (C) 2004, Govorukhin V.N.
% This file is intended for use with MATLAB and was produced for MATDS-program
% http://www.math.rsu.ru/mexmat/kvm/matds/
% lyapunov.m is free software. lyapunov.m is distributed in the hope that it
% will be useful, but WITHOUT ANY WARRANTY.
%

%       n=number of nonlinear odes
%       n2=n*(n+1)=total number of odes

n1=n; n2=n1*(n1+1);

%  Number of steps

nit = round((tend-tstart)/stept);

% Memory allocation

y=zeros(n2,1); cum=zeros(n1,1); y0=y;
gsc=cum; znorm=cum;

% Initial values
```

```
y(1:n)=ystart(:);

for i=1:n1 y((n1+1)*i)=1.0; end;

t=tstart;

% Main loop

for ITERLYAP=1:nit

% Solutuion of extended ODE system

  [T,Y] = feval(fcn_integrator,rhs_ext_fcn,[t t+stept],y);

  t=t+stept;
  y=Y(size(Y,1),:);

  for i=1:n1
      for j=1:n1 y0(n1*i+j)=y(n1*j+i); end;
  end;

%       construct new orthonormal basis by gram-schmidt

  znorm(1)=0.0;
  for j=1:n1 znorm(1)=znorm(1)+y0(n1*j+1)^2; end;

  znorm(1)=sqrt(znorm(1));

  for j=1:n1 y0(n1*j+1)=y0(n1*j+1)/znorm(1); end;
```

```
  for j=2:n1
      for k=1:(j-1)
          gsc(k)=0.0;
          for l=1:n1 gsc(k)=gsc(k)+y0(n1*l+j)*y0(n1*l+k); end;
      end;

      for k=1:n1
          for l=1:(j-1)
              y0(n1*k+j)=y0(n1*k+j)-gsc(l)*y0(n1*k+l);
          end;
      end;

      znorm(j)=0.0;
      for k=1:n1 znorm(j)=znorm(j)+y0(n1*k+j)^2; end;
      znorm(j)=sqrt(znorm(j));

      for k=1:n1 y0(n1*k+j)=y0(n1*k+j)/znorm(j); end;
  end;

%       update running vector magnitudes

  for k=1:n1 cum(k)=cum(k)+log(znorm(k)); end;

%       normalize exponent

  for k=1:n1
      lp(k)=cum(k)/(t-tstart);
  end;
```

```
% Output modification

  if ITERLYAP==1
     Lexp=lp;
     Texp=t;
  else
     Lexp=[Lexp; lp];
     Texp=[Texp; t];
  end;

  if (mod(ITERLYAP,ioutp)==0)
     fprintf('t=%6.4f',t);
     for k=1:n1 fprintf(' %10.6f',lp(k)); end;
     fprintf('\n');
  end;

  for i=1:n1
      for j=1:n1
          y(n1*j+i)=y0(n1*i+j);
      end;
  end;
end;
```

F

```
%Olga Niyibizi
%This code graphs the GPOY model
clc
clear
it = 100000; %number of iterations
%Initialize vectors
```

```
x = zeros(1,it);
theta = zeros(1,it);

%initial conditions
x1 = 1;
angle = 0;

%Main loop to determine the population size at each year
for i=1:it
    x(i) = x1;
    theta(i) = angle;
    x1 = 2*(1.5)*tanh(x1)*cos(2*pi*angle);
    angle = mod((angle +(sqrt(5)-1)/2),1) ;
end

%Stem is a MATLAB function that plots discrete systems
stem(theta,x);
```