UCSB, Physics 129AL, Computational Physics: Problem Set 4

Zihang Wang (UCSB), zihangwang@ucsb.edu

Due: Feb 28, 11:59PM

GitHub Submission Guideline

We will use GitHub for problem set submissions. By the due date, you should have a single public repository on GitHub containing all the work you have done for this problem set. Finally, upload a screenshot or a .txt file to Canvas with your GitHub username and repository name so the TA knows who you are and which repository you are using for the problem set.

Remember: talk to your fellow students, work together, and use GPTs. You will find it much easier than working alone. Good luck! All work should be done in the Docker container, and don't forget to commit it to Git!

Question 1: Bifurcation and Chaos in the Logistic Map

The logistic map is a discrete dynamical system given by the recurrence relation:

$$x_{n+1} = rx_n(1 - x_n), \tag{1}$$

where $0 \leq x_n \leq 1$ represents a normalized population, and r is a growth parameter.

a) fixed points

Consider the logistic map with an initial condition $x_0 = 0.2$. For a given r, write a Python program that find the fixed points of the following quadratic equation,

$$f(x)rx\,(1-x) \quad \to \quad x = rx\,(1-x). \tag{2}$$

For $r = 1, 2, 3, 4$, determine the stability of these fixed points as a function of r by analyzing the derivative $f'(x) = r(1 - 2x)$ at the fixed points.

b) dynamic programming

Using dynamic programming we discussed previously, (you can use recursion or iteration), write a Python program that iterate the logistic map for a given r and a convergence threshold. Numerically iterate the logistic map for $r = 2, 3, 3.5, 3.8, 4.0$ with an initial condition $x_0 = 0.2$.

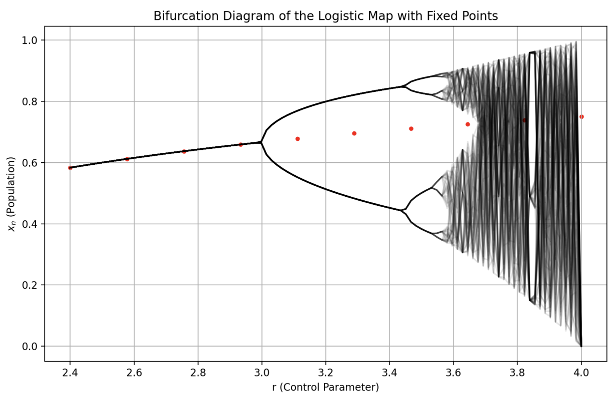
c) Different initial condition

Using the above iterative program you made, test different initial condition, $x_0 = 0.1, 0.3, 0.5$. Do they produce the same result? Plot the time series x_n for different values of r labeled by initial conditions.

d) Bifurcation Condition

With an initial condition $x_0 = 0.2$, analyze the system for values of r in the range $0 < r \leq 4$. Identify critical values of r_i where bifurcations occur, leading to period-doubling behavior.

- $0 < r \leq r_1$: The population dies out ($x_n \to 0$).
- $1 < r \leq r_2$: The system stabilizes at a single fixed point.
- $r \approx r_3$: A bifurcation occurs, and the system oscillates between two values.
- $r \approx r_4$: A second bifurcation, leading to a period-4 cycle.
- $r \approx r_5$: Period-8 cycle.
- $r \approx r_6$: Chaos.
- $r \approx r_7$: Small windows of periodicity within chaos.
- $r = r_8$: Chaotic behavior.



e) Scaling in Bifurcation

Let's consider a modified logistic map of a discrete dynamical system with the recurrence relation,

$$x_{n+1} = r x_n (1 - x_n^{\gamma}), \tag{3}$$

where $0.5 \leq \gamma \leq 1.5$. Write a program that plot the first bifurcation point with respect to the function γ.

Question 2: Julia Set, Area, and Fractal dimension.

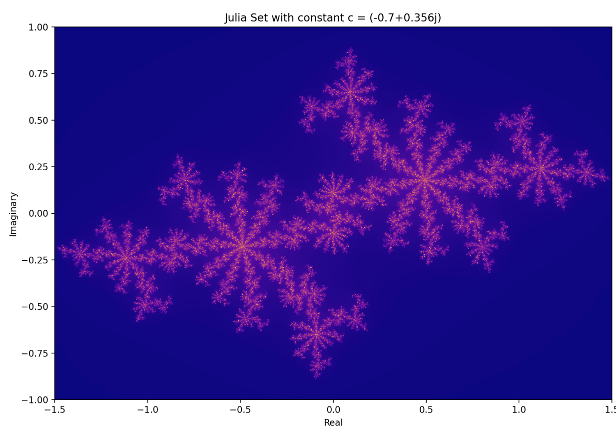
The Julia set is a set that arises from iterating a simple complex function, and the boundary of the set often exhibits fractal behavior. It is defined as the quadratic map,

$$f(z) = z^2 + c, \rightarrow z_{n+1} = f(z_n) = z_n^2 + c,$$

where z is a complex number and c is a constant complex parameter.

a) Julia set generation

Write a Python program that generate the Julia set over the following region, x density = 800pt, y density = 800pt, $\mathrm{x_{min}} = -1.5, \mathrm{x_{max}} = 1.5$, $\mathrm{y_{min}} = -1, \mathrm{y_{max}} = 1$, $c = -0.7, 0.356i$, iteration=256. You should see something like this,

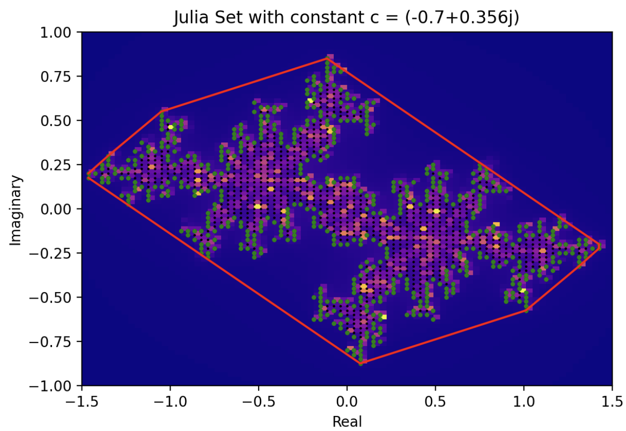


b) Convex Hull

For a given Julia set, write a Python program that find the area enclosed by the convex hull. You should have something below.

c) Contour

For a given Julia set, write a program that find the area enclosed by its contour. You should have something below.



Some information: Fractional Dimension

Fractional dimension is defined via scaling laws that measure how the number of self-similar pieces of an object grows as the scale decreases.

One common method to determine the fractional dimension of an object is the **box-counting method**, which follows these steps: We first divide the space into smaller boxes, and the size of each box is denoted by ϵ. Then, let $N(\epsilon)$ be the number of boxes that contain part of the object. The **fractal dimension** D, which is defined as,

$$D = \lim_{\epsilon \to 0} \frac{\log N(\epsilon)}{\log(1/\epsilon)} \tag{4}$$

You may find the the above box-counting method very similar to the concept of renormalization group (RG), where both focus on the scaling properties of a system. The box-counting method observes how the number of boxes required to cover an object scales as the box size is reduced. In RG, one observes how the system's properties (such as correlations or other physical observables) change as the scale is changed. In the context of fractals, this scaling behavior reveals the fractal dimension, which characterizes how an object behaves at different scales.

d) box-counting method

Write a program that numerically estimate the fractional dimension using the box-counting method. Use this program to find the fractional dimension of the Julia set you generated previously.

Question 3: Lorenz system

The Lorenz system is a set of three coupled, nonlinear differential equations that describe the motion of a fluid layer under thermal convection used in atmospheric studies, and it is a classic example of deterministic chaos.

1 Equations of the Lorenz System

The system is defined by the following equations:

$$\frac{dx}{dt} = \sigma(y - x), \ \frac{dy}{dt} = x(\rho - z) - y, \ \frac{dz}{dt} = xy - \beta z. \tag{5}$$

Here, the variables x, y, and z represent the **state of the system**, and the parameters σ, ρ, and β determine its behavior. You should note that x, y, z are not necessarily the positions.

a) Real systems

Find a physical system that can be described by the above Lorenz system. Explain the physical meanings of states x, y, z and σ, ρ, β.

b) Lorenz attractors

For certain values of σ, ρ, and β, the Lorenz system exhibits chaotic behavior. For a given σ, ρ, and β, write a Python program that find all coordinates of the Lorenz attractors x_a, y_a, z_a over a time range $t \in (0, 12)$.

You can take, $\sigma = 10$, $\rho = 48$, and $\beta = 3$ as your starting point.

c) Trajectories

Write a Python program that produces a video that shows the time evolutions of the Lorenz system.

Upload the video to Github using the following conditions, $\sigma = 10$, $\rho = 48$, and $\beta = 3$ $t \in (0, 12)$.