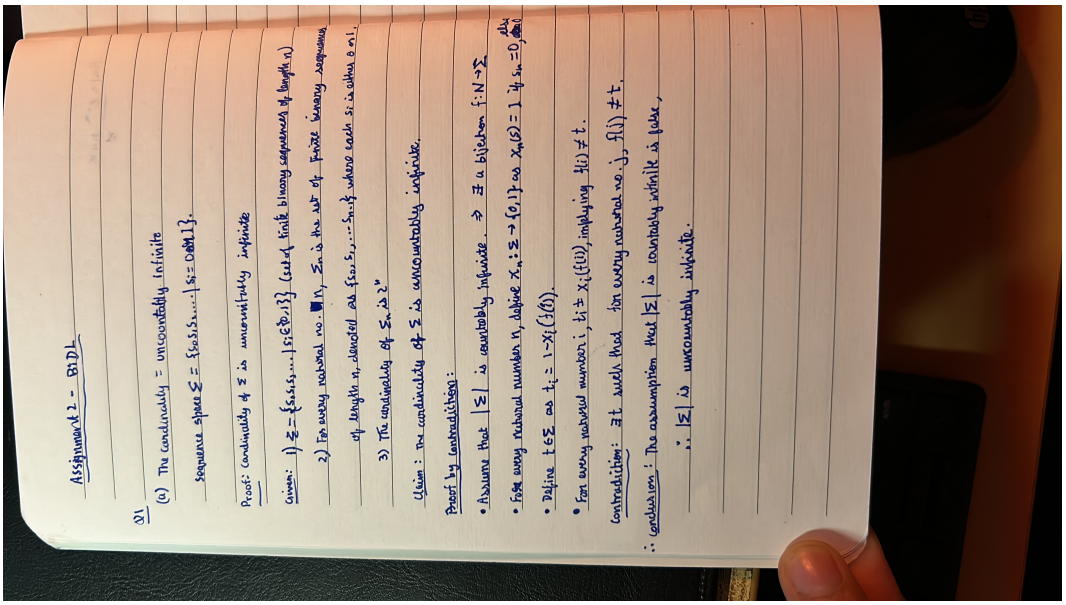
Assignment2

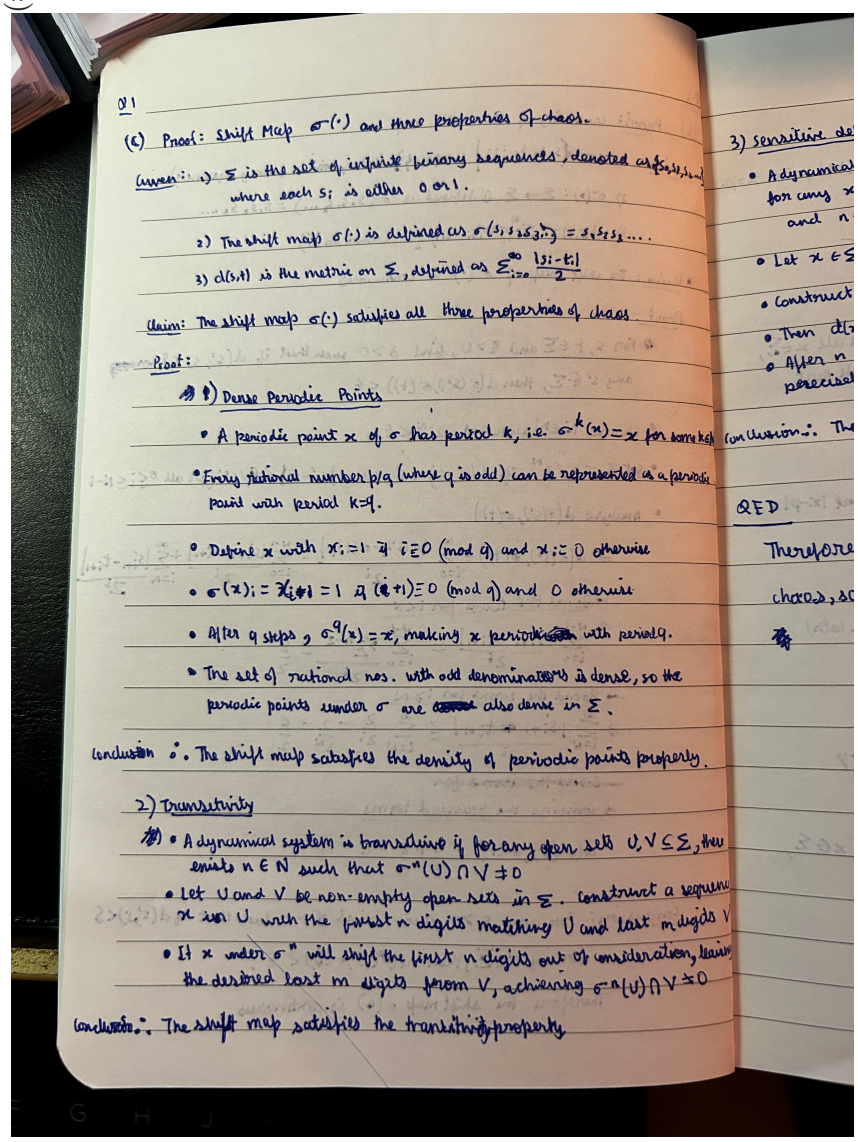
February 21, 2024

Q1.

a)



c)



(c) Proof: Shift Map $\sigma(\cdot)$ and three properties of chaos.

Given: 1) Σ is the set of infinite binary sequences, denoted as $\{s_0, s_1, s_2, \ldots\}$ where each s_i is either 0 or 1.

2) The shift map $\sigma(\cdot)$ is defined as $\sigma(s_0, s_1, s_2, s_3, \ldots) = s_1, s_2, s_3, \ldots$.

3) $d(s,t)$ is the metric on Σ, defined as $\sum_{i=0}^{\infty} \frac{|s_i - t_i|}{2}$

Claim: The shift map $\sigma(\cdot)$ satisfies all three properties of chaos.

Proof:

1) Dense Periodic Points

- A periodic point x of σ has period k, i.e. $\sigma^k(x) = x$ for some k
- Every rational number p/q (where q is odd) can be represented as a periodic point with period $k = q$.
- Define x with $x_i = 1$ if $i \equiv 0 \pmod q$ and $x_i = 0$ otherwise
- $\sigma(x)_i = x_{i+1} = 1$ if $(i+1) \equiv 0 \pmod q$ and 0 otherwise
- After q steps, $\sigma^q(x) = x$, making x periodic with period q.
- The set of rational nos. with odd denominators is dense, so the periodic points under σ are also dense in Σ.

Conclusion $\therefore$ The shift map satisfies the density of periodic points property.

2) Transitivity

- A dynamical system is transitive if for any open sets $U, V \subseteq \Sigma$, there exists $n \in \mathbb{N}$ such that $\sigma^n(U) \cap V \neq 0$
- Let U and V be non-empty open sets in Σ. Construct a sequence x in U with the first n digits matching U and last m digits V
- If x under σ^n will shift the first n digits out of consideration, leaving the desired last m digits from V, achieving $\sigma^n(U) \cap V \neq 0$

Conclusion $\therefore$ The shift map satisfies the transitivity property

3) Sensitive de

- A dynamical for any x and n
- Let $x \in \Sigma$
- Construct
- Then $d(x$
- After n precisel

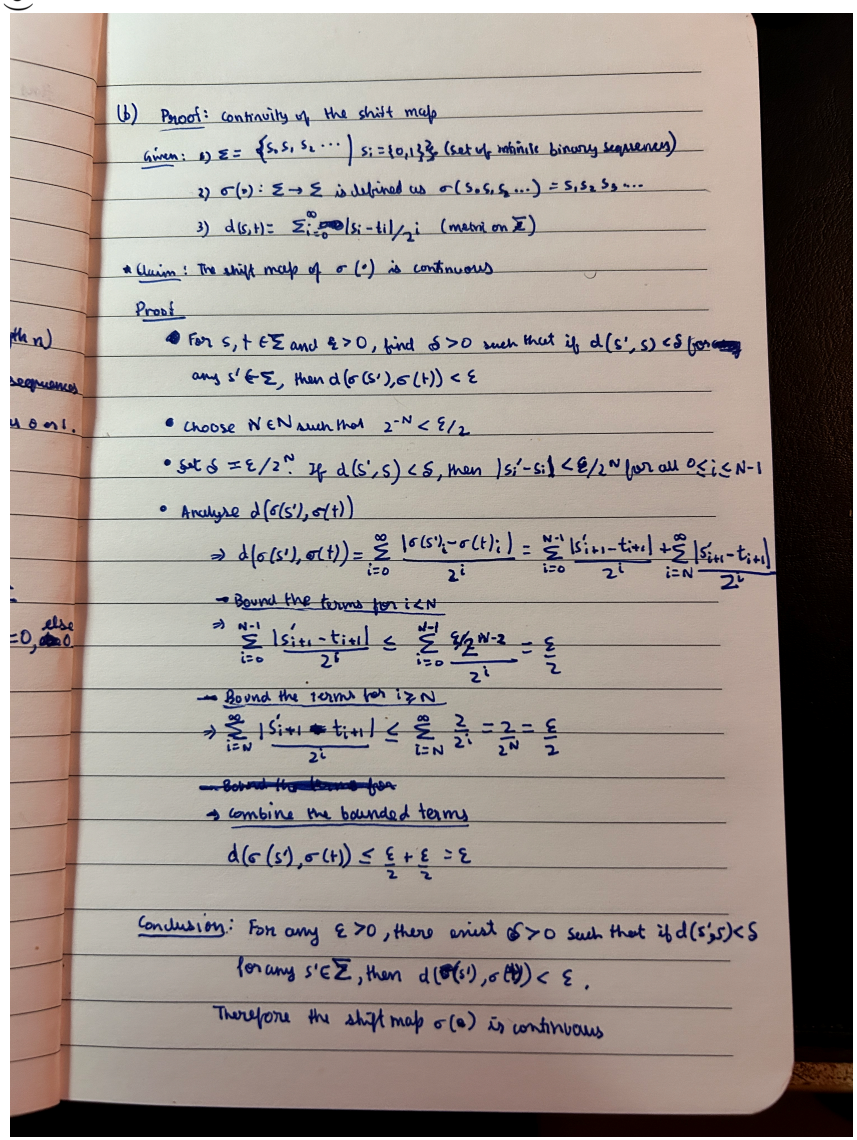
Conclusion $\therefore$ The

QED

Therefore

chaos, Δ

b)



(b) Proof: Continuity of the shift map

Given: 1) $\Sigma = \{s_0, s_1, s_2 \cdots \mid s_i = \{0,1\}\}$ (set of infinite binary sequences)

2) $\sigma(\cdot) : \Sigma \to \Sigma$ is defined as $\sigma(s_0, s_1, s_2 \ldots) = s_1, s_2, s_3 \ldots$

3) $d(s,t) = \sum_{i=0}^{\infty} |s_i - t_i| / 2^i$ (metric on Σ)

* Claim: The shift map of $\sigma(\cdot)$ is continuous

Proof

- For $s, t \in \Sigma$ and $\varepsilon > 0$, find $\delta > 0$ such that if $d(s', s) < \delta$ for any $s' \in \Sigma$, then $d(\sigma(s'), \sigma(t)) < \varepsilon$
- Choose $N \in \mathbb{N}$ such that $2^{-N} < \varepsilon/2$
- Set $\delta = \varepsilon / 2^N$. If $d(s', s) < \delta$, then $|s'_i - s_i| < \varepsilon / 2^N$ for all $0 \le i \le N-1$
- Analyse $d(\sigma(s'), \sigma(t))$

$$\Rightarrow d(\sigma(s'), \sigma(t)) = \sum_{i=0}^{\infty} \frac{|\sigma(s')_i - \sigma(t)_i|}{2^i} = \sum_{i=0}^{N-1} \frac{|s'_{i+1} - t_{i+1}|}{2^i} + \sum_{i=N}^{\infty} \frac{|s'_{i+1} - t_{i+1}|}{2^i}$$

→ Bound the terms for $i < N$

$$\Rightarrow \sum_{i=0}^{N-1} \frac{|s'_{i+1} - t_{i+1}|}{2^i} \le \sum_{i=0}^{N-1} \frac{\varepsilon/2^{N-2}}{2^i} = \frac{\varepsilon}{2}$$

→ Bound the terms for $i \ge N$

$$\Rightarrow \sum_{i=N}^{\infty} \frac{|s'_{i+1} - t_{i+1}|}{2^i} \le \sum_{i=N}^{\infty} \frac{2}{2^i} = \frac{2}{2^N} = \frac{\varepsilon}{2}$$

→ Combine the bounded terms

$$d(\sigma(s'), \sigma(t)) \le \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon$$

Conclusion: For any $\varepsilon > 0$, there exist $\delta > 0$ such that if $d(s', s) < \delta$ for any $s' \in \Sigma$, then $d(\sigma(s'), \sigma(t)) < \varepsilon$.

Therefore the shift map $\sigma(\cdot)$ is continuous

3) Sensitive dependance on initial conditions:

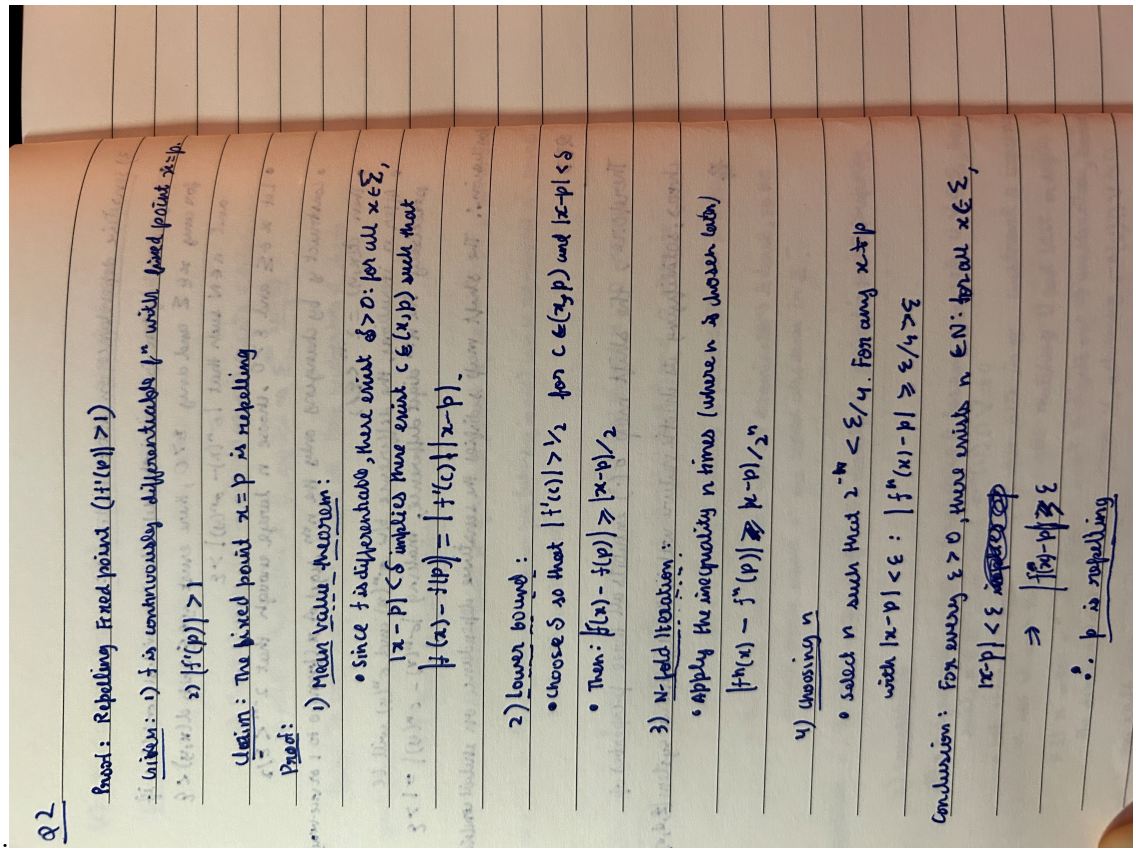
- A dynamical system is sensitively dependant on initial conditions if for any $x \in \Sigma$ and any $\varepsilon > 0$, there exist $y \in \Sigma$ with $d(x,y) < \varepsilon$ and $n \in \mathbb{N}$ such that $|\sigma^n(x) - \sigma^n(y)| > \varepsilon$
- Let $x \in \Sigma$ and $\varepsilon > 0$. Choose n large enough that $2^{-n} < \varepsilon/2$
- Construct y by changing only the n^{th} digit of x (from 0 to 1 or vice-versa)
- Then $d(x,y) = 2^{-n} < \varepsilon/2$
- After n iterations, the difference b/w $\sigma^n(x)$ and $\sigma^n(y)$ will be precisely the n^{th} digit difference, making $|\sigma^n(x) - \sigma^n(y)| = 1 > \varepsilon$

Conclusion:. The shift map satisfies the sensitive dependance on initial conditions

QED

Therefore, the shift map $\sigma(\cdot)$ exhibits all three properties of chaos, solidifying its chaotic nature within the dynamical system (Σ, d, σ)

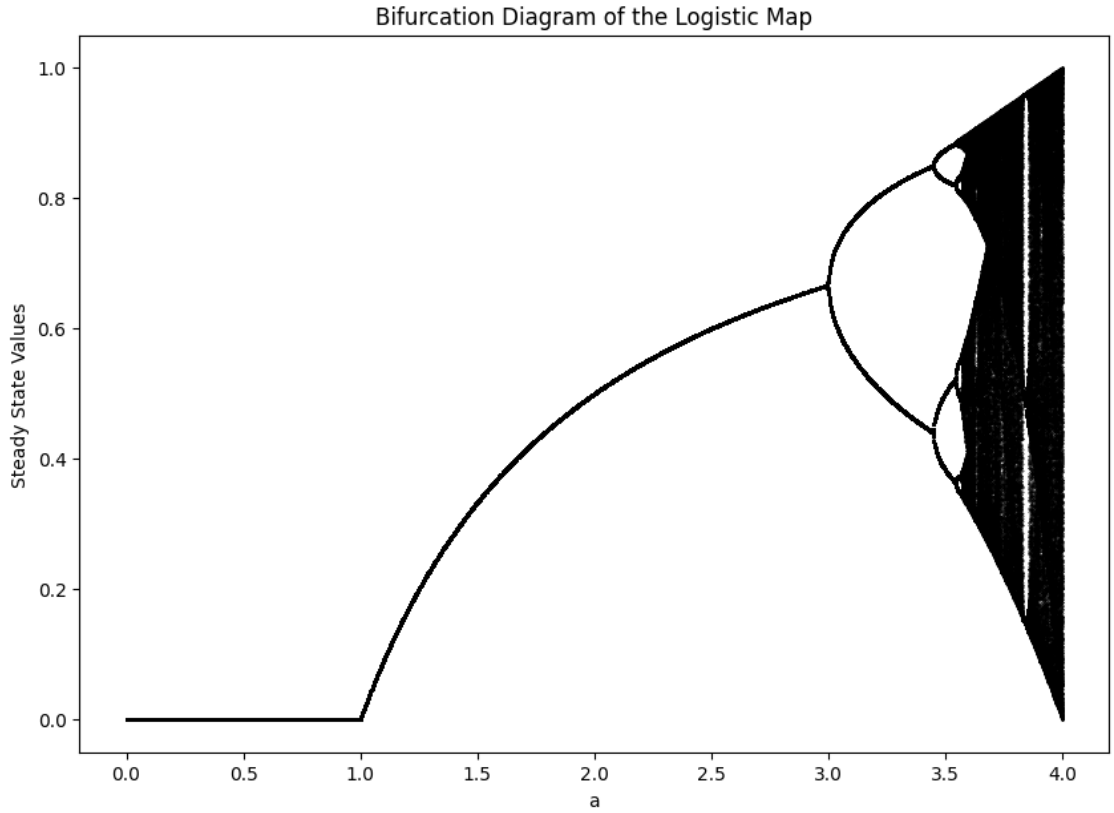
Q2.



Q3.

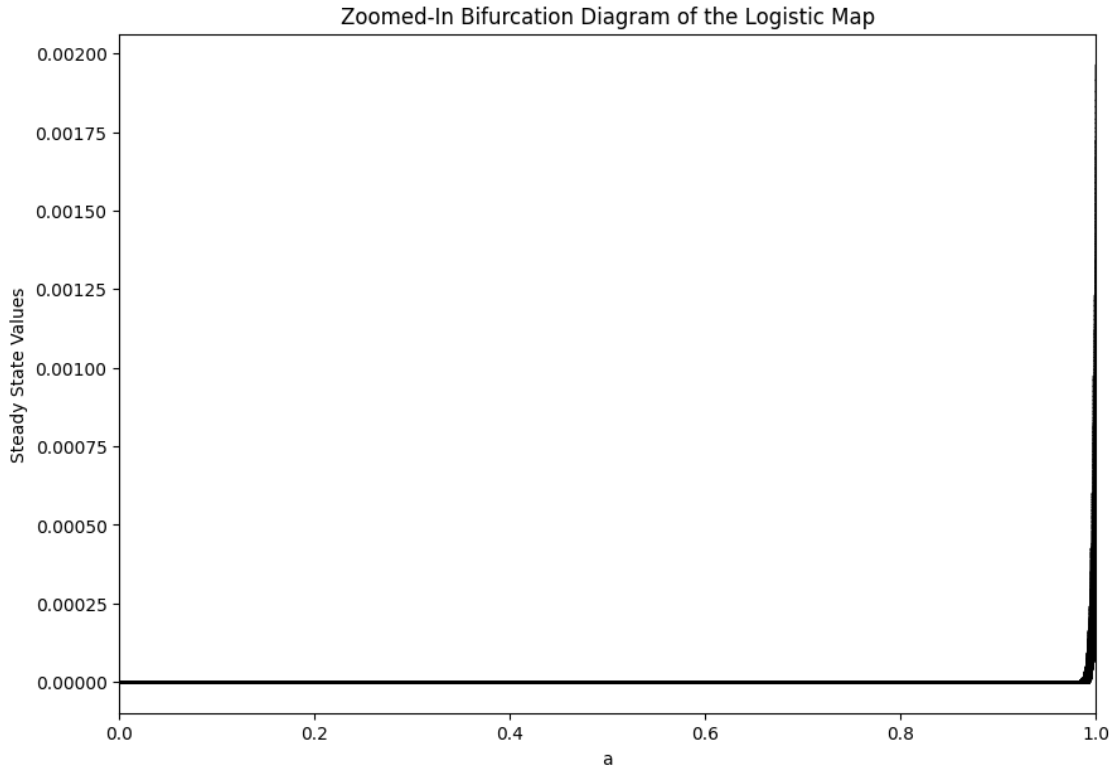
[1]:
```
import numpy as np
import matplotlib.pyplot as plt

# Define the logistic map function
def logistic_map(x, a):
    return a * x * (1 - x)

# Generate different values of a
a_values = np.linspace(0, 4, 2000)

# Initialize an empty list to store the steady state values
steady_states = []

# Iterate over the a values
for a in a_values:
    # Start from a random initial value
    x = np.random.rand()

    # Perform 1000 iterations
    for i in range(1000):
        x = logistic_map(x, a)
```

```
        # After the first 500 transient values, store the steady state values
        if i >= 500:
            steady_states.append((a, x))

# Convert the list of steady state values into a numpy array
steady_states = np.array(steady_states)

# Plot the bifurcation diagram
plt.figure(figsize=(10, 7))
plt.scatter(steady_states[:, 0], steady_states[:, 1], s=0.1, color='black')
plt.title('Bifurcation Diagram of the Logistic Map')
plt.xlabel('a')
plt.ylabel('Steady State Values')
plt.show()
```



[15]:
```
# Generate different values of a between 0 and 1
a_values = np.linspace(0, 1, 2000)

# Initialize an empty list to store the steady state values
steady_states = []
```

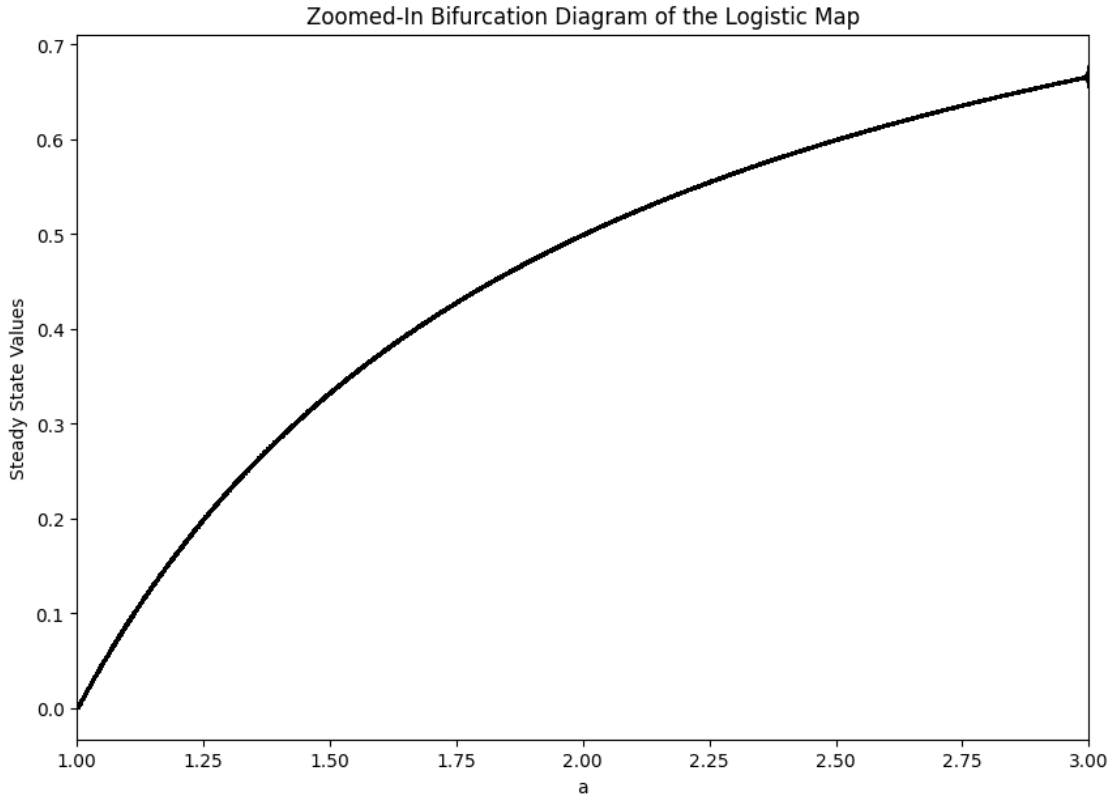
```
# Iterate over the a values
for a in a_values:
    # Start from a random initial value
    x = np.random.rand()

    # Perform 1000 iterations
    for i in range(1000):
        x = logistic_map(x, a)

        # After the first 500 transient values, store the steady state values
        if i >= 500:
            steady_states.append((a, x))

# Convert the list of steady state values into a numpy array
steady_states = np.array(steady_states)

# Plot the zoomed-in bifurcation diagram
plt.figure(figsize=(10, 7))
plt.scatter(steady_states[:, 0], steady_states[:, 1], s=0.1, color='black')
plt.title('Zoomed-In Bifurcation Diagram of the Logistic Map')
plt.xlabel('a')
plt.ylabel('Steady State Values')
plt.xlim([0, 1])
plt.show()
```



As for the mathematical explanation, the logistic map equation is defined as xn=axn−1(1−xn−1)

When a is between 0 and 1, the logistic map will always converge to 0 regardless of the initial value of x. This is because for 0 a < 1, the term ax(1 - x) will always be less than x for any x in the interval [0, 1]. Therefore, the sequence {x_n} generated by the logistic map will be a decreasing sequence that converges to 0. This is why you see a concentration of values at zero in the bifurcation diagram for 0 a < 1.

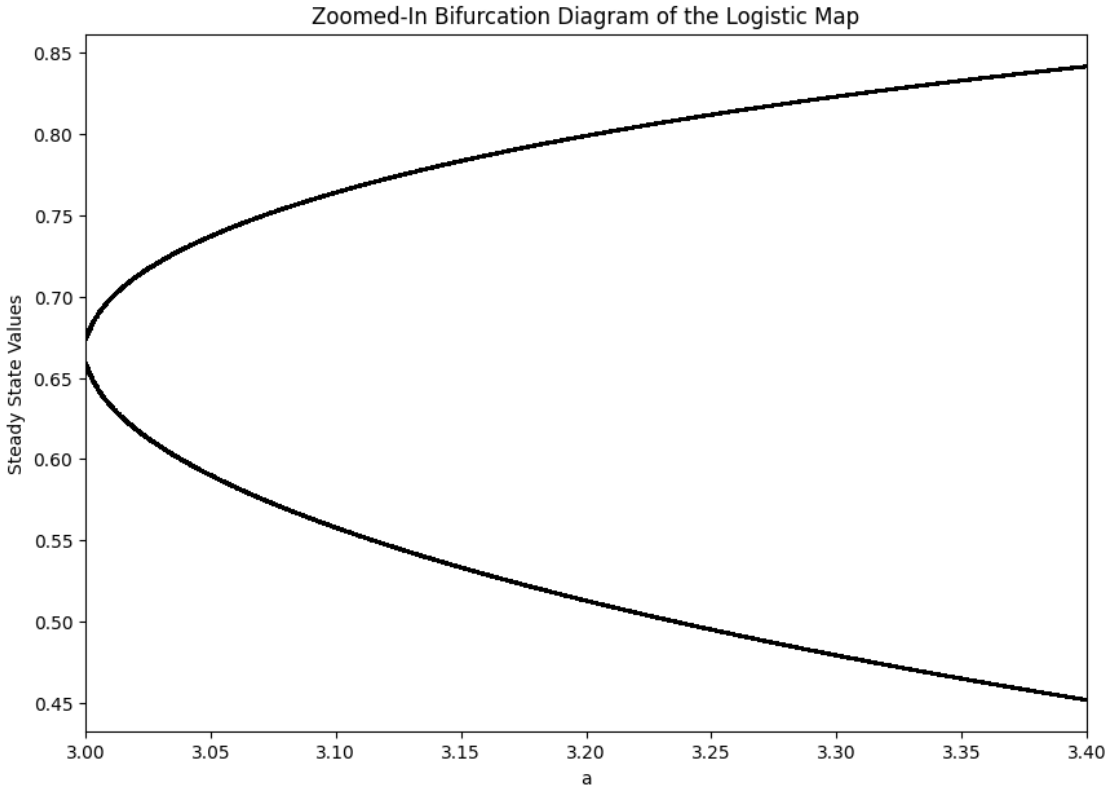
[16]:
```
# Generate different values of a between 1 and 3
a_values = np.linspace(1, 3, 2000)

# Initialize an empty list to store the steady state values
steady_states = []

# Iterate over the a values
for a in a_values:
    # Start from a random initial value
    x = np.random.rand()

    # Perform 1000 iterations
    for i in range(1000):
        x = logistic_map(x, a)

        # After the first 500 transient values, store the steady state values
        if i >= 500:
            steady_states.append((a, x))

# Convert the list of steady state values into a numpy array
steady_states = np.array(steady_states)

# Plot the zoomed-in bifurcation diagram
plt.figure(figsize=(10, 7))
plt.scatter(steady_states[:, 0], steady_states[:, 1], s=0.1, color='black')
plt.title('Zoomed-In Bifurcation Diagram of the Logistic Map')
plt.xlabel('a')
plt.ylabel('Steady State Values')
plt.xlim([1, 3])
plt.show()
```



mathematical explanation, when a is between 1 and 3, the logistic map will converge to a single value that is not zero. This is because for 1 a < 3, the term ax(1 - x) will be less than x for x > 1/a, and greater than x for x < 1/a. Therefore, the sequence {x_n} generated by the logistic map will oscillate around 1/a and eventually converge to this value. This is why there is a spread of values in the bifurcation diagram for 1 a < 3, and not a concentration at zero.

[23]:
```
# Generate different values of a between 3 and 3.4
a_values = np.linspace(3, 3.4, 2000)

# Initialize an empty list to store the steady state values
steady_states = []

# Iterate over the a values
for a in a_values:
    # Start from a random initial value
    x = np.random.rand()

    # Perform 1000 iterations
    for i in range(1000):
        x = logistic_map(x, a)

        # After the first 500 transient values, store the steady state values
```

```
        if i >= 500:
            steady_states.append((a, x))

# Convert the list of steady state values into a numpy array
steady_states = np.array(steady_states)

# Plot the zoomed-in bifurcation diagram
plt.figure(figsize=(10, 7))
plt.scatter(steady_states[:, 0], steady_states[:, 1], s=0.1, color='black')
plt.title('Zoomed-In Bifurcation Diagram of the Logistic Map')
plt.xlabel('a')
plt.ylabel('Steady State Values')
plt.xlim([3, 3.4])
plt.show()
```

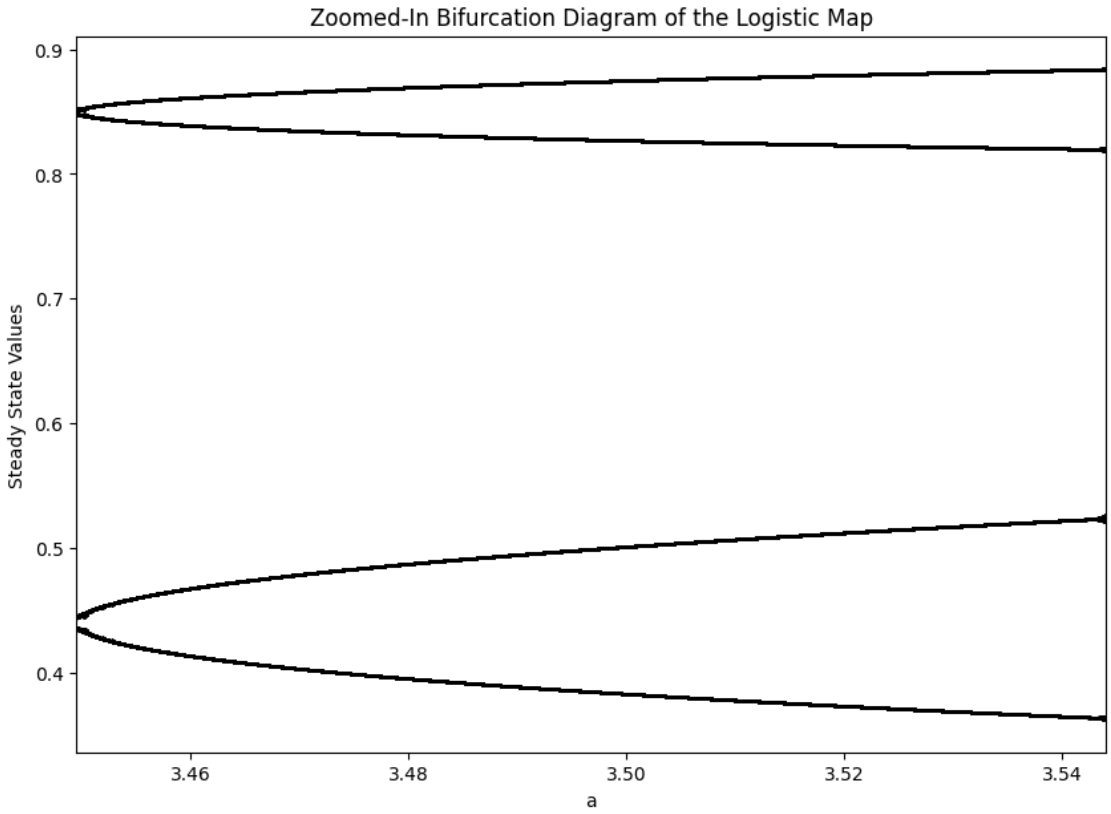


mathematical explanation, when a is between 3 and 3.4, the logistic map will exhibit period-doubling bifurcation. This means that the system will oscillate between two values instead of converging to a single value. The two values are called period-two fixed points.

The period-two fixed points are stable if the absolute value of the derivative of the logistic map at these points is less than 1. The derivative of the logistic map is a(1 - 2x). So, the stability condition for the period-two fixed points x1 and x2 is $|a(1 - 2x1)| < 1$ and $|a(1 - 2x2)| < 1$.

The initial conditions leading to period-two orbits can be any value in the interval [0, 1]. This is because the logistic map is a chaotic system, which means that its behavior is highly sensitive to initial conditions. Even a small change in the initial value can lead to drastically different trajectories.

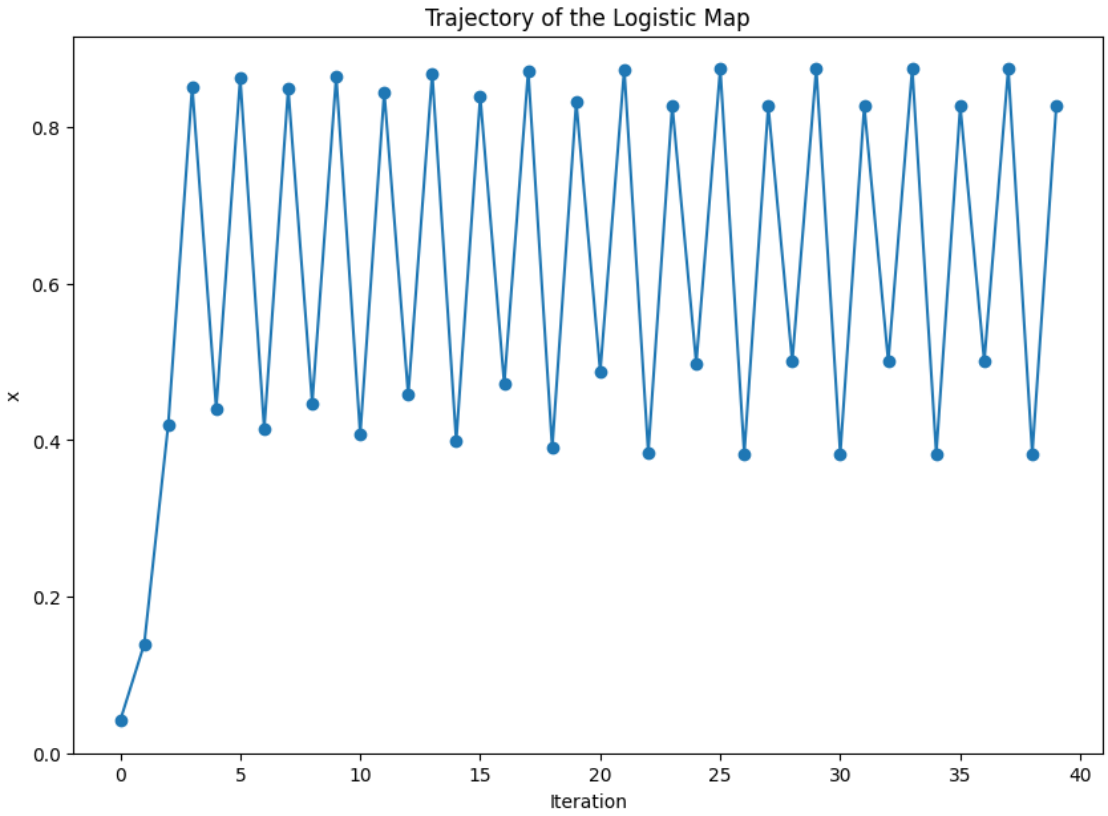
[18]:
```
# Generate different values of a between 3.44949 and 3.54409
a_values = np.linspace(3.44949, 3.54409, 2000)

# Initialize an empty list to store the steady state values
steady_states = []

# Iterate over the a values
for a in a_values:
    # Start from a random initial value
    x = np.random.rand()

    # Perform 1000 iterations
    for i in range(1000):
        x = logistic_map(x, a)

        # After the first 500 transient values, store the steady state values
        if i >= 500:
            steady_states.append((a, x))

# Convert the list of steady state values into a numpy array
steady_states = np.array(steady_states)

# Plot the zoomed-in bifurcation diagram
plt.figure(figsize=(10, 7))
plt.scatter(steady_states[:, 0], steady_states[:, 1], s=0.1, color='black')
plt.title('Zoomed-In Bifurcation Diagram of the Logistic Map')
plt.xlabel('a')
plt.ylabel('Steady State Values')
plt.xlim([3.44949, 3.54409])
plt.show()
```



observations, when a is between 3.44949 and 3.54409, the logistic map exhibits period-doubling bifurcation. This means that the system will oscillate between four values instead of converging to a single value or oscillating between two values. The four values are called period-four fixed points. This is a characteristic behavior of chaotic systems, where a small change in the parameter a can lead to drastically different behaviors.

[20]:
```
# Define the logistic map function
def logistic_map(x, a):
    return a * x * (1 - x)

# Initialize the parameter a and the initial value x0
a = 3.5
x0 = 0.012

# Initialize an empty list to store the trajectory
trajectory = []

# Perform 40 iterations
for i in range(40):
    x0 = logistic_map(x0, a)
    trajectory.append(x0)
```

```
# Plot the trajectory
plt.figure(figsize=(10, 7))
plt.plot(range(40), trajectory, 'o-')
plt.title('Trajectory of the Logistic Map')
plt.xlabel('Iteration')
plt.ylabel('x')
plt.show()
```

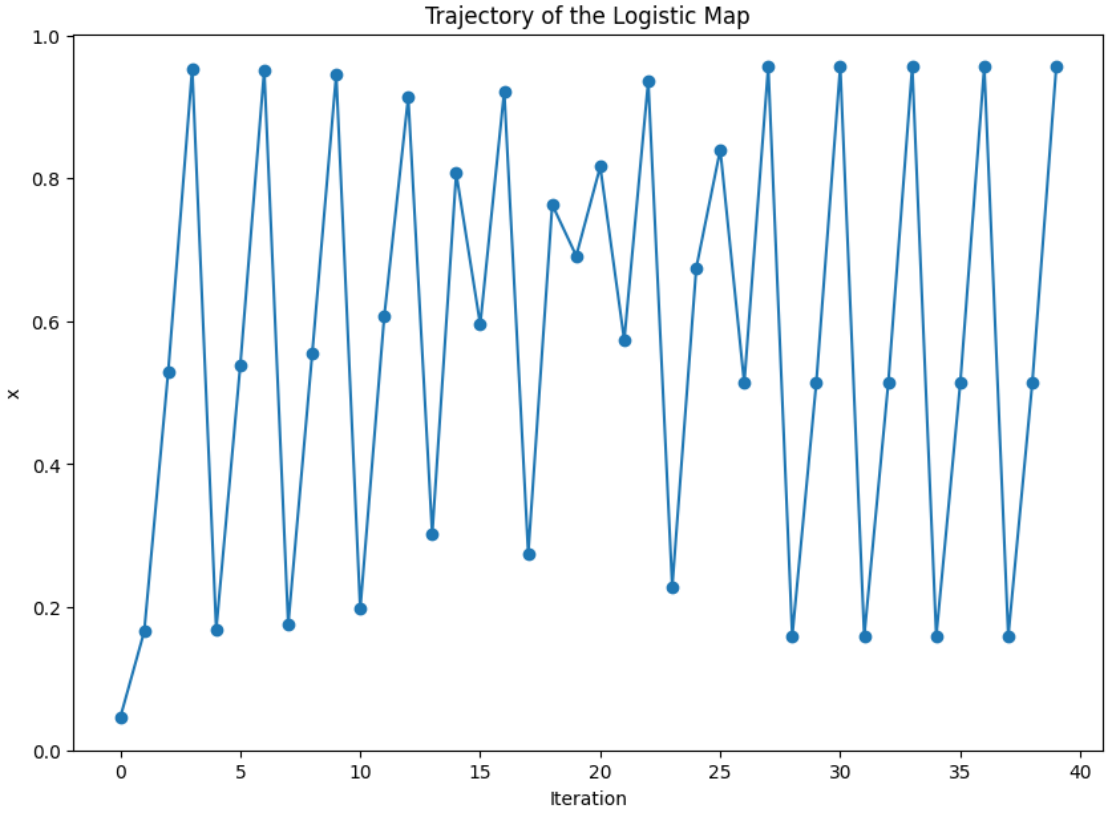


observations, the trajectory does not settle down to a fixed point. Instead, it appears to oscillate between four different values. This is a characteristic behavior of the logistic map when a is in the regime of $3 < a < 3.57$, where it exhibits period-doubling bifurcations leading to chaos.

The value $a = 3.5$ is in this regime, and the logistic map exhibits complex behavior, despite being a simple deterministic system. The trajectory's behavior is highly sensitive to the initial condition x0, and even a tiny change in x0 can lead to drastically different trajectories. This sensitivity to initial conditions is one of the hallmarks of chaos.

[21]:
```
# Define the logistic map function
def logistic_map(x, a):
    return a * x * (1 - x)
```

```python
# Initialize the parameter a and the initial value x0
a = 1 + 2*np.sqrt(2)
x0 = 0.012

# Initialize an empty list to store the trajectory
trajectory = []

# Perform 40 iterations
for i in range(40):
    x0 = logistic_map(x0, a)
    trajectory.append(x0)

# Plot the trajectory
plt.figure(figsize=(10, 7))
plt.plot(range(40), trajectory, 'o-')
plt.title('Trajectory of the Logistic Map')
plt.xlabel('Iteration')
plt.ylabel('x')
plt.show()
```



Observations, the trajectory does not settle down to a fixed point or a small cycle. Instead, it

appears to jump around chaotically among many different values. This is a characteristic behavior of the logistic map when a is in the chaotic regime.

The value a = 1 + 2√2 is approximately 3.8284, which is in the chaotic regime (3.57 < a < 4). In this regime, the logistic map exhibits complex and unpredictable behavior, despite being a simple deterministic system. The trajectory's behavior is highly sensitive to the initial condition x0, and even a tiny change in x0 can lead to drastically different trajectories. This sensitivity to initial conditions is one of the hallmarks of chaos.

.

.

Why are we not choosing a>4 for logistic map? .

The logistic map is defined as xn+1=axn(1−xn) where x is a number between 0 and 1, and a is the bifurcation parameter.

When a > 4, the logistic map no longer behaves as expected because the values of x can exceed the interval [0, 1]. This is contrary to the original biological motivation for the logistic map, which was to model population growth with a carrying capacity (the maximum population size that the environment can sustain indefinitely). In this model, x represents the population size as a proportion of the carrying capacity, so it should always be between 0 and 1.

Moreover, for a > 4, the logistic map becomes unstable and the values of x can become very large or negative, which doesn't make sense in the context of population dynamics. Therefore, we usually restrict a to the interval [0, 4] when studying the logistic map.

[ ]: