Can Small Changes Really Change Everything?

Well, as a passionate student in the field of mathematics and physics, the first and most basic model which I learnt in school was a pendulum. A mass which swings back and forth. Its trajectory, its velocity, and location can be calculated to the edge of precision. As a student solving equations in class, I found it to be elegant. As a person with a physics background I loved the deterministic model. Ask yourself a question, but what happens when we make a small change?

If we were to replace the normal pendulum with a double pendulum, a second mass attached to the first. Pause, I want you to think, what might happen? Will the motion of the simple pendulum be conserved? In what ways would things change? Well, this small addition transforms the system entirely. The motion becomes wildly unpredictable, looping and twisting in ways that seem almost random. Two nearly identical starting positions quickly diverge into completely different trajectories. This is our first encounter with Chaos Theory my dear friend!

I know you might be wondering why the motion of the system is so unpredictable and chaotic? Before we continue forward we need to define what we mean by chaos. We have the following criteria for a system to be considered chaotic:

1. Sensitivity to Initial Conditions: Small differences in starting points grow exponentially over time. This can be understood by this thing known as the butterfly effect. It's like a butterfly located in India causing a hurricane in the UK.
2. Topological Mixing: The system eventually “mixes” all regions of space. It's essentially that any region of the system will eventually overlap with any other region after some time.
3. Dense Periodic Orbits: Periodic behavior exists everywhere within the system. For any point in space, there are periodic points arbitrarily close to it.

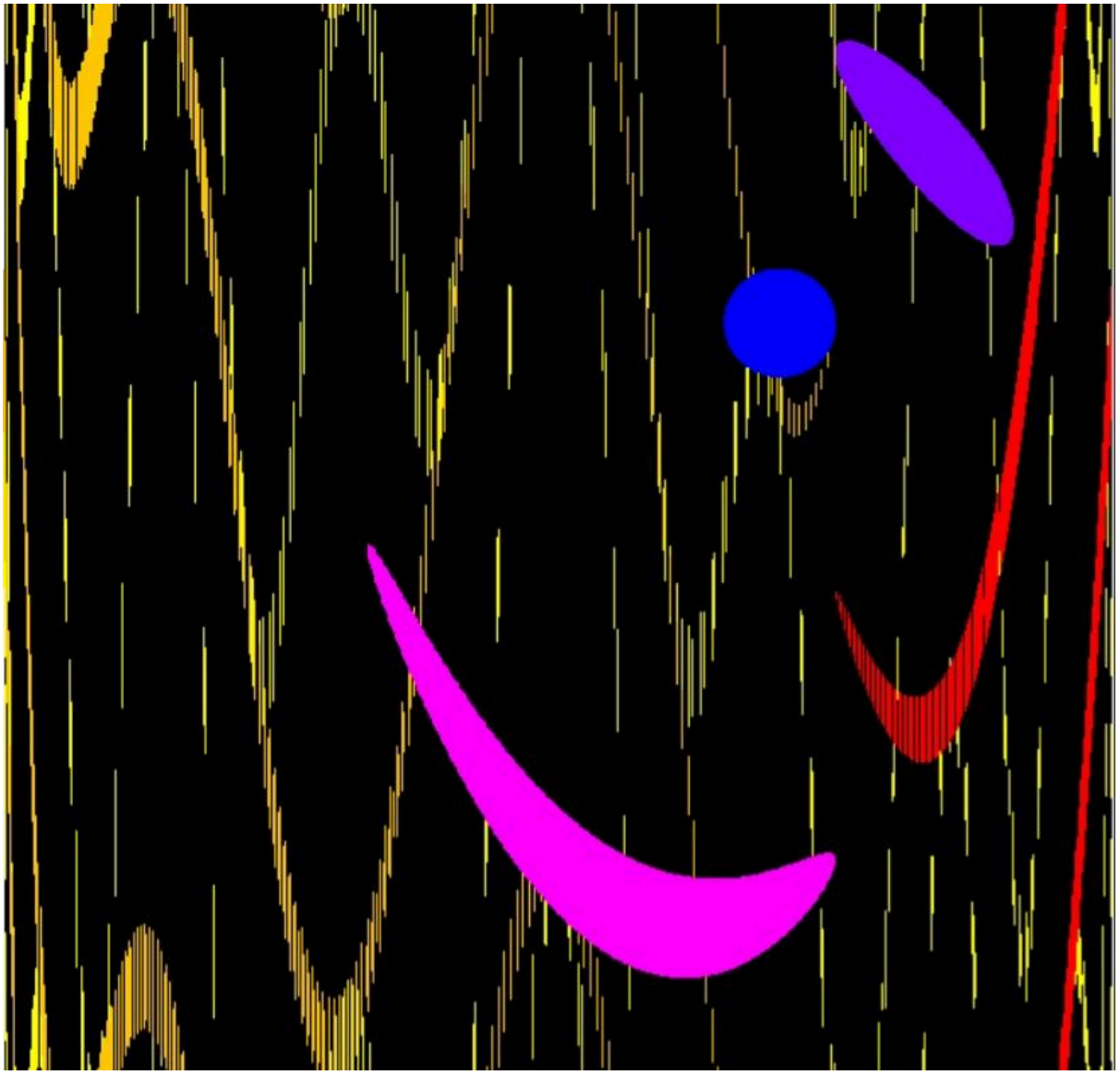
Maths behind the motion of a double pendulum

Double pendulum can be described by a system of coupled nonlinear differential equations, I would demonstrate this using a very very basic and approximate python code which uses no library to keep it simple and grounded.

There are two masses m1 and m2 and lengths of L1 and L2 . These two are described using angles of theta1 and theta2 with angular velocity w1 and w2. Gravity begins g which provides the restoring force. And Δ = theta1 - theta2The motion is essential a coupled mom linear equation where each pendulum affects the angle of the other. The accelerations are computed directly using trigonometric relations and the interaction terms between the two

masses. The denominator: 2m1 +m2 - m2cos2Δ comes from combining the equations and represents how the system resists motion depending on relative configuration.

Acceleration includes gravity which is pulling m1 down, the influence of m2 via sin(theta1-2theta2) along with sinΔ multiplied by the velocity term pf $w^2$2L2 + $w^2$1L1 cosΔ. The second acceleration depends mainly on the interaction of the sinΔ term. The system uses small times steps dt where angular velocity affects the acceleration and angles by velocity. The sine and cosine functions are approximated using a basic version of theTaylor series, which replaces standard library functions.



The following future highlights the Topological Mixing, a weak requirement for a system to be chaotic.

Now what's the motivation behind all this? Behind us learning about chaos theory.

The application of chaos theory

One of the most elegant applications of chaos theory is in the field of population dynamics! Well similar to the double pendulum, population is an interesting system. Pause and think why and how population growth could be chaotic?

Let's start with the most simple idea, unlimited grown, let's say every generation reproduces freely so
$dN/dt = rN$
Where
N is population
r is growth rate
K is carrying capacity

Let's add a limitation, if we have a small population the growth is fated and vice versa. Hence growth is directly proportional to N x (available resources. {AR}) As AR shrinks as population grows hence
$AR = 1- N/K$

The new growth equation is:
$dN/dt = rN(1-N/K)$
Well you might say, it's still smooth. There is no chaos! The latter arises only when we switch from continuous time to discrete generation. Why? Well, the real population often reproducts in steps, don't they?

The logistic map, aka the chaos formula is
$x(n+1) = rx(n) \bullet (1-x(n))$

x(n) is the normalised population, which is between 0 to 1
And r is the growth parameter

Conclusion

It's the small things in life which have the biggest impact. Whenever you are exploring a new concept in maths or physics remember we live in a non linear problem. Continue to learn, it can lead you to find yourself in places and allow you to have opportunities which you couldn't ever have dreamt of!

Code

Unable to share them via a pdf. Hence decided to put it here

Double Pendulum Visual Simulation

```
import time

# Constants
g = 9.81
L1 = 1.0
L2 = 1.0
m1 = 1.0
m2 = 1.0

theta1 = 1.0
theta2 = 1.0
omega1 = 0.0
omega2 = 0.0

dt = 0.05

# Math Functions
def sin(x):
    return x - (x**3)/6 + (x**5)/120 - (x**7)/5040

def cos(x):
    return 1 - (x**2)/2 + (x**4)/24 - (x**6)/720

#Accelerations
def accelerations(theta1, theta2, omega1, omega2):
    delta = theta1 - theta2

    den = (2*m1 + m2 - m2 * cos(2*delta))

    a1 = (
        -g * (2*m1 + m2) * sin(theta1)
        - m2 * g * sin(theta1 - 2*theta2)
        - 2 * sin(delta) * m2 *
        (omega2**2 * L2 + omega1**2 * L1 * cos(delta))
    ) / (L1 * den)

    a2 = (
        2 * sin(delta) *
        (
            omega1**2 * L1 * (m1 + m2)
            + g * (m1 + m2) * cos(theta1)
            + omega2**2 * L2 * m2 * cos(delta)
        )
```

```
    ) / (L2 * den)

    return a1, a2

width = 80
height = 24

for step in range(300):
    a1, a2 = accelerations(theta1, theta2, omega1, omega2)

    omega1 += a1 * dt
    omega2 += a2 * dt

    theta1 += omega1 * dt
    theta2 += omega2 * dt

    x1 = int(30 * sin(theta1)) + width // 2
    y1 = int(10 * cos(theta1)) + 5

    x2 = x1 + int(30 * sin(theta2))
    y2 = y1 + int(10 * cos(theta2))

    screen = [[" " for _ in range(width)] for _ in range(height)]

    screen[2][width // 2] = "O"

    if 0 <= y1 < height and 0 <= x1 < width:
        screen[y1][x1] = "o"

    if 0 <= y2 < height and 0 <= x2 < width:
        screen[y2][x2] = "x"

    print("\n" * 5)
    for row in screen:
        print("".join(row))

    time.sleep(0.03)
```

Logistic Map Chaos Simulator

```
def logistic_map(r, x):
    return r * x * (1 - x)
```

```
def run_simulation():
    print("\n=== New Simulation ===")

    r = float(input("Enter value of r (0 to 4): "))
    x = float(input("Enter initial value x0 (0 to 1): "))
    iterations = int(input("Enter number of iterations: "))

    print("\nStep |    x value    | Visualization")
    print("-----------------------------------------------")

    for i in range(iterations):
        x = logistic_map(r, x)

        bar_length = int(x * 40)
        bar = "" * bar_length

        print(f"{i+1:4} | {x:12.6f} | {bar}")


def sensitivity_test():
    print("\n=== Sensitivity Test (Butterfly Effect) ===")

    x1 = float(input("Enter first initial value: "))
    x2 = float(input("Enter slightly different value: "))
    r = float(input("Enter r value: "))
    iterations = int(input("Iterations: "))

    print("\nStep |     x1        |     x2        | Difference")
    print("-----------------------------------------------------------")

    for i in range(iterations):
        x1 = logistic_map(r, x1)
        x2 = logistic_map(r, x2)
        diff = abs(x1 - x2)

        print(f"{i+1:4} | {x1:12.6f} | {x2:12.6f} | {diff:10.6f}")

print(" Logistic Map Chaos Simulator ")

while True:
    print("\nChoose an option:")
    print("1. Run Simulation")
    print("2. Sensitivity Test")
    print("3. Exit")

    choice = input("Enter choice (1/2/3): ")

    if choice == "1":
```

```
    run_simulation()

elif choice == "2":
    sensitivity_test()

elif choice == "3":
    print("\nGoodbye! Keep exploring chaos ")
    break

else:
    print("Invalid choice. Try again.")

cont = input("\nDo you want to continue? (y/n): ").lower()
if cont != 'y':
    print("\nSession ended. Stay curious ")
    break
```