CHAOS IN PHYSICS

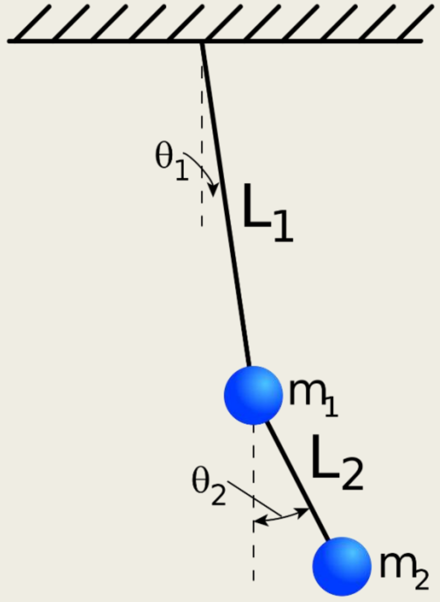
Ali Chaudhary
67694450

The Goal

- The main question is: How can we use a double pendulum and a Lorenz attractor to demonstrate chaos in physics?
- A system that is chaotic is one that will act drastically different and unpredictable by changing the initial values of the system.

Inputs to vary: The Pendulum

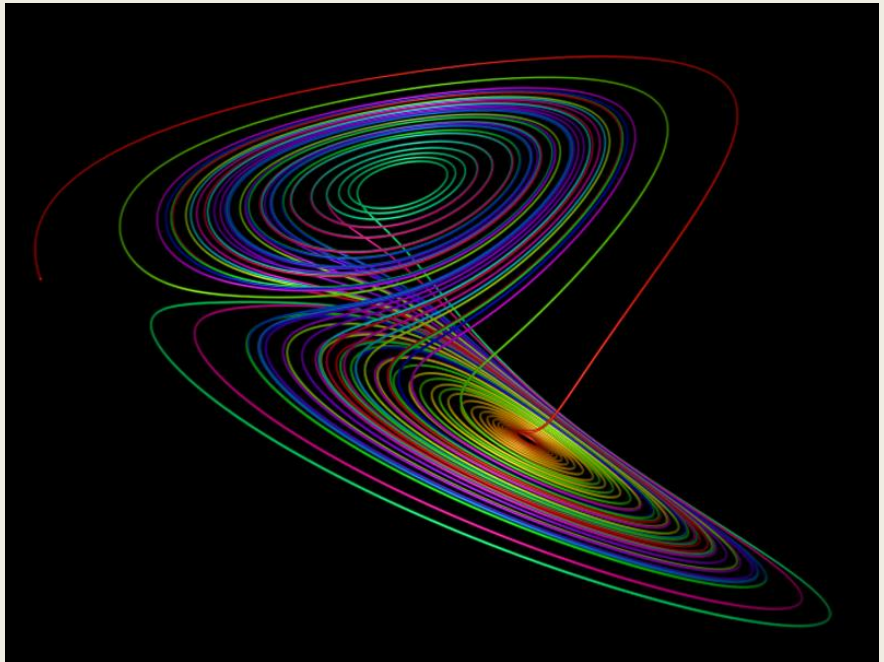
- Bellow is the double pendulum. A single pendulum is simply a bob on a massless string or a rod that oscillates back and forth.
- A double pendulum is simply two of these.
- The inputs you can vary are the angle, the length, and the mass of both of the pendulums.
- In theory differing inputs should produce wildly different outputs.



Inputs to vary: The Lorenz Attractor

- The Lorenz Attractor is a solution to the Lorenz system.
- This also has chaotic solutions.
- By slightly varying sigma, rho or Beta you can get completely different results.

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



What the Program Outputs: Pendulum

```
 ---------------------------------------------
|                                             |
|      WE WILL BE COMPARING THE MOTION OF A   |
|      SINGLE AND DOUBLE PENDULUM IN THIS     |
|      PROGRAM TO DEMONSTRATE CHAOS IN PYSICS |
|                                             |
 ---------------------------------------------


----------------------------------------
                    |\
                    |A\
                    |  \
                        \L
                         \
                          \
                           \
                           (M)
Above is what a single pendulum looks like.
'A' represents angle of the pendulum in degrees.
Please enter a  starting value of A (between 10 and 179):40

'L' represents the length of a rigid, string/rod in cm.
Please enter the value of L:150

'M' represents the mass of the pendulum ball/string in grams.
Please enter the value of M:8

Our angle in radians is: 0.698132
 Our length in cm is: 150.000000
 Our mass in g is: 8.000000


THE DATA HAS BEEN SAVED TO 'spendulum.dat'!
```

```
 -----------------------------------------
|                                         |
|    NOW WE WILL MODEL A DOUBLE PENDULUM  |
|     AND COMPARE THE RESULTS TO ONE      |
|           SINGULAR PENDULUM             |
|                                         |
 -----------------------------------------


-----------------------------------
                  |\
                  | \
                  |A1\
                  |   \L1
                       \
                        \
                         \
                         (M1)
                          /|
                         / |
                     L2/A2|
                       /   |
                      /
                     /
                    /
                  (M2)
Above is what a double pendulum looks like.

'A2' represents the angle of the second pendulum .
Please enter the initial value of A2:-90

'L2' represents the length of the second  rigid, string/rod in cm.
Please enter the value of L2:250

'M2' represents the mass of the second pendulum ball/string in grams.
Please enter the value of M2:5

For the second pendulum, our angle in radians is: -90.000000
 Our length in cm is: 250.000000
 Our mass in g is: 5.000000


THE DATA HAS BEEN SAVED TO 'dpendulum.dat'!
```

- Once it takes these inputs it numerically solves the equations below

$$\ddot{\theta}_1 + Ml\ddot{\theta}_2 \cos\Delta\theta + Ml\dot{\theta}_2^2 \sin\Delta\theta + \omega^2 \sin\theta_1 = 0,$$

$$\ddot{\theta}_1 \cos\Delta\theta + l\ddot{\theta}_2 - \dot{\theta}_1^2 \sin\Delta\theta + \omega^2 \sin\theta_2 = 0,$$

$$\Delta\theta \equiv \theta_1 - \theta_2$$

$$M \equiv m_2/(m_1 + m_2)$$

$$l \equiv l_2/l_1$$

$$\omega^2 = g/l_1$$

What the Program Outputs: Lorenz Attractor

- Sample input is shown
- The data is then to output to a file based on the equations showed again below.

```
 ----------------------------------------------
|                                              |
|      WE WILL USING A LORENZ ATTRACTOR TO     |
|      DEMONSTRATE CHAOS IN PHYSICS            |
|                                              |
 ----------------------------------------------



Please enter a value for sigma greater than zero:10

Please enter a value for rho greater than zero:28

Please enter a value for beta greater than zero:2.66666666666666666
[atchaudh@hpc-login-13-1 project]$
```

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

How it works: Pendulum

- The double pendulum's equations can be derived through Newtonian mechanics, and used in the Runge-Kutta method through casting.
- The casted equations are as shown below
- It should be noted that these equations can not be solved analytically, only numerically interestingly enough.

```
double double1v(double t1, double t2, double av1, double av2,
               double l1, double l2, double m1, double m2)
{
  //t1 is angle one t2 is angle two
  //av is angular velocity
  //l is length
  //m is mass

  double d = t2-t1;//doing this makes calculating the eqn easier
  double answer = m2*l1*av1*av1*sin(d)*cos(d)+m2*GRAV*sin(t2)*cos(d);
  answer += m2*l2*av2*av2*sin(d)-(m1+m2)*GRAV*sin(t1);
  answer /= ((m1+m2)*l1-m2*l1*cos(d)*cos(d));
  return answer;
}

//Second pendulum first derivative
double double2a(double angularvelocity2)
{
  return angularvelocity2;
}

//Second pendulum second derivative
double double2v(double t1, double t2, double av1, double av2,
               double l1, double l2, double m1, double m2)
{
  //t1 is angle one t2 is angle two
  //av is angular velocity
  //l is length
  //m is mass

  double d = t2-t1;//doing this makes calculating the eqn easier
  double answer = -m2*l2*av2*av2*sin(d)*cos(d)+(m1+m2)*(GRAV*sin(t1)*cos(d)-l1*av1*av1*sin(d)-GRAV*sin(t2));
  answer /= (m1+m2)*l2-m2*l2*cos(d)*cos(d);
  return answer;

}
```

$$
\begin{aligned}
\dot{\theta}_1 &= \omega_1 \\
\dot{\omega}_1 &= \frac{m_2 \ell_1 \omega_1^2 \sin\Delta\cos\Delta + m_2 g \sin\theta_2 \cos\Delta + m_2 \ell_2 \omega_2^2 \sin\Delta - (m_1+m_2) g \sin\theta_1}{(m_1+m_2)\ell_1 - m_2 \ell_1 \cos^2\Delta} \\
\dot{\theta}_2 &= \omega_2 \\
\dot{\omega}_2 &= \frac{-m_2 \ell_2 \omega_2^2 \sin\Delta\cos\Delta + (m_1+m_2)(g\sin\theta_1\cos\Delta - \ell_1\omega_1^2\sin\Delta - g\sin\theta_2)}{(m_1+m_2)\ell_2 - m_2\ell_2\cos^2\Delta}
\end{aligned}
$$

How it works: Pendulum cont.

- By using pointers in a void function I was able to change all 4 equations simultaneously
- These were then printed to a File for gnuplot
- I was able to animate the data in gnuplot by using a counter, and then the replot function in an “if statement”

```
//The implementation of the Rung Kutta 4 method for the double pendulum
void doublerk4(double ti, double ai1, double vi1, double ai2, double vi2, double tf,
               double* af1, double* vf1, double* af2, double* vf2,
                double l1, double l2, double m1, double m2)
{
  double k1a, k2a, k3a, k4a;
  double k1v, k2v, k3v, k4v;

  double h = tf - ti;
  double t = ti;
  t+= 0;

  //Do one pendulum first then the other
  k1a = double1a( vi1);
  k1v = double1v( ai1, ai2, vi1, vi2, l1, l2, m1, m2);

  k2a = double1a( vi1+k1v*h/2.0);
  k2v = double1v( ai1+k1a*h/2.0, ai2, vi1+k1v*h/2.0, vi2, l1, l2, m1, m2);

  k3a = double1a( vi1+k2v*h/2.0);
  k3v = double1v( ai1+k2a*h/2.0, ai2, vi1+k2v*h/2.0, vi2, l1, l2, m1, m2 );

  k4a = double1a(vi1+k3v*h);
  k4v = double1v(ai1+k3a*h, ai2, vi1+k3v*h, vi2, l1, l2, m1, m2);


  *af1 = ai1 + (k1a + 2.0*(k2a+k3a) + k4a)* h/6.0;
  *vf1 = vi1 + (k1v + 2.0*(k2v+k3v) + k4v)* h/6.0;

  //Now for the second one
  k1a = double2a( vi2);
  k1v = double2v( ai1, ai2, vi1, vi2, l1, l2, m1, m2);

  k2a = double2a( vi2+k1v*h/2.0);
  k2v = double2v( ai1, ai2+k1a*h/2.0, vi1, vi2+k1v*h/2.0, l1, l2, m1, m2);

  k3a = double2a( vi2+k2v*h/2.0);
  k3v = double2v( ai1, ai2+k2a*h/2.0, vi1, vi2+k2v*h/2.0, l1, l2, m1, m2 );

  k4a = double2a(vi2+k3v*h);
  k4v = double2v(ai1, ai2+k3a*h, vi1, vi2+k3v*h, l1, l2, m1, m2);


  *af2 = ai2 + (k1a + 2.0*(k2a+k3a) + k4a)* h/6.0;
  *vf2 = vi2 + (k1v + 2.0*(k2v+k3v) + k4v)* h/6.0;
}
```

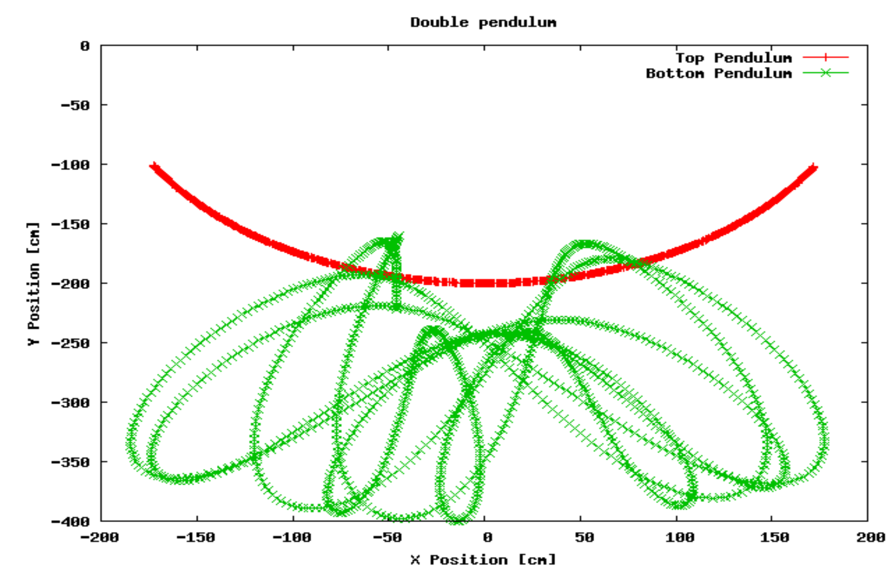
How it works: Lorenz Attractor

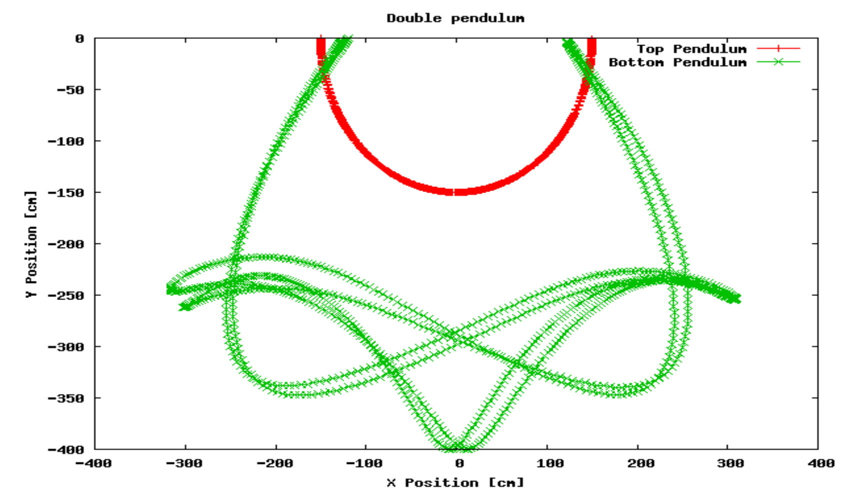
- After the data is inputted all that I had to do was pass it through a slightly modified RK4 method
- This involved a few more moving parts since there were 3 equations involved.

```
//System of 3 differential equations
//dx/dt
double fx(double sigma, double x, double y)
{
  return sigma*(y-x);
}
//dy/dt
double fy(double rho, double x, double y, double z)
{
  return x*(rho-z)-y;
}
double fz(double beta, double x, double y, double z)
{
  return x*y-beta*z;
}

void rk4(double ti, double xi, double yi, double zi,
         double sigma, double rho, double beta,
         double tf, double* xf, double* yf, double* zf)
{
  double k1x, k2x, k3x, k4x;
  double k1y, k2y, k3y, k4y;
  double k1z, k2z, k3z, k4z;

  double h = tf - ti;
  double t = ti;
  t+=0;

  //Do it for each 3 equations
  k1x = fx(sigma, xi, yi);
  k1y = fy(rho, xi, yi, zi);
  k1z = fz(beta, xi, yi, zi);

  k2x = fx(sigma, xi+k1x*h/2.0, yi+k1y*h/2.0);
  k2y = fy(rho, xi+k1x*h/2.0, yi+k1y*h/2.0, zi+k1z*h/2.0);
  k2z = fz(beta,xi+k1x*h/2.0, yi+k1y*h/2.0, zi+k1z*h/2.0);

  k3x = fx(sigma, xi+k2x*h/2.0, yi+k2y*h/2.0);
  k3y = fy(rho, xi+k2x*h/2.0, yi+k2y*h/2.0, zi+k2z*h/2.0);
  k3z = fz(beta,xi+k2x*h/2.0, yi+k2y*h/2.0, zi+k2z*h/2.0);

  k4x = fx(sigma, xi+k3x*h, yi+k3y*h/2.0);
  k4y = fy(rho, xi+k3x*h, yi+k3y*h, zi+k3z*h);
  k4z = fz(beta,xi+k3x*h, yi+k3y*h, zi+k3z*h);

  *xf = xi + (k1x + 2.0*(k2x+k3x) + k4x)* h/6.0;
  *yf = yi + (k1y + 2.0*(k2y+k3y) + k4y)* h/6.0;
  *zf = zi + (k1z + 2.0*(k2z+k3z) + k4z)* h/6.0;

}
```

Results: Pendulum

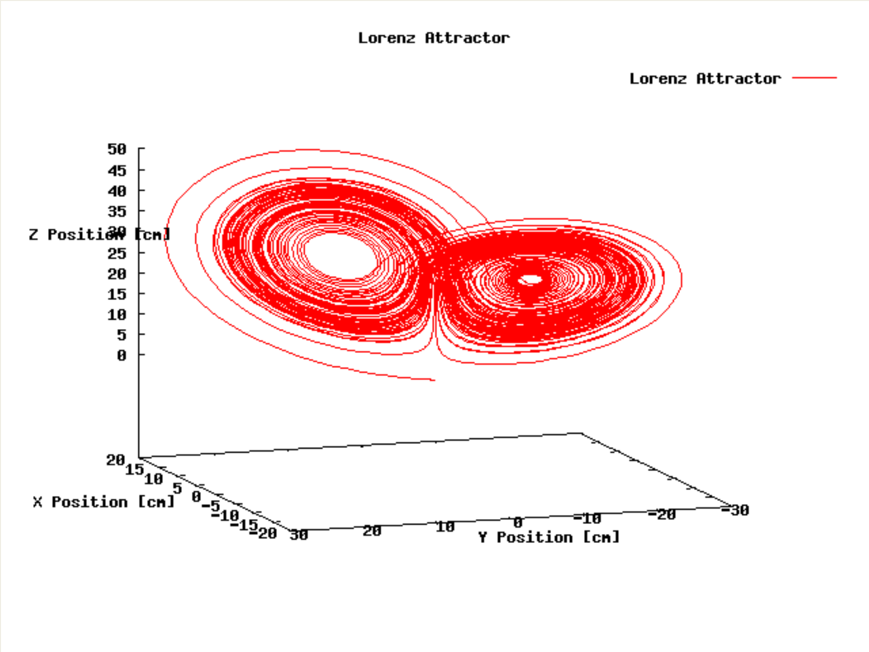
- The data was plotted using gnuplot.
- Just by varying the inputs by a few percent these drastic changes were observed demonstrating chaos

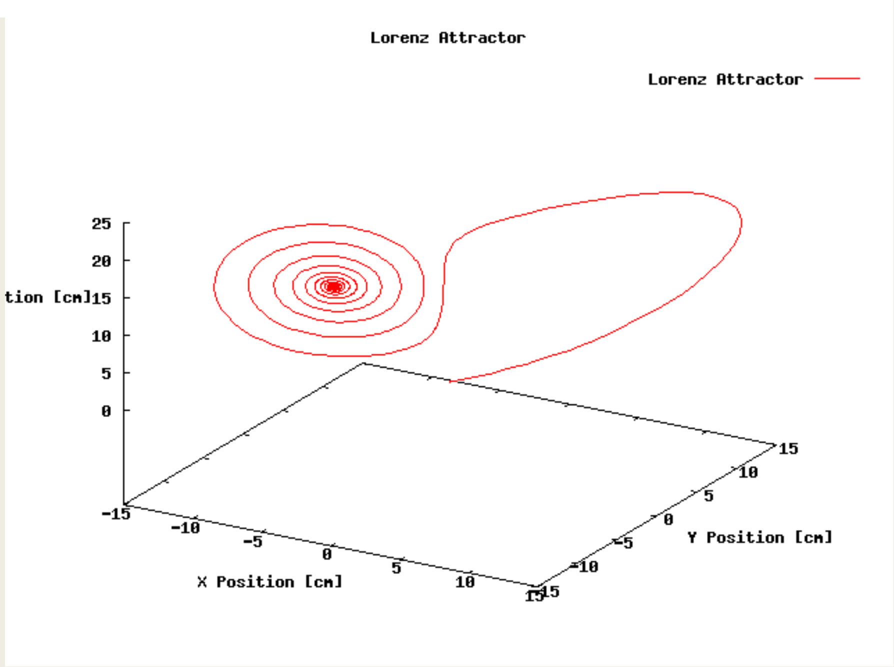




Results: Lorenz Attractor

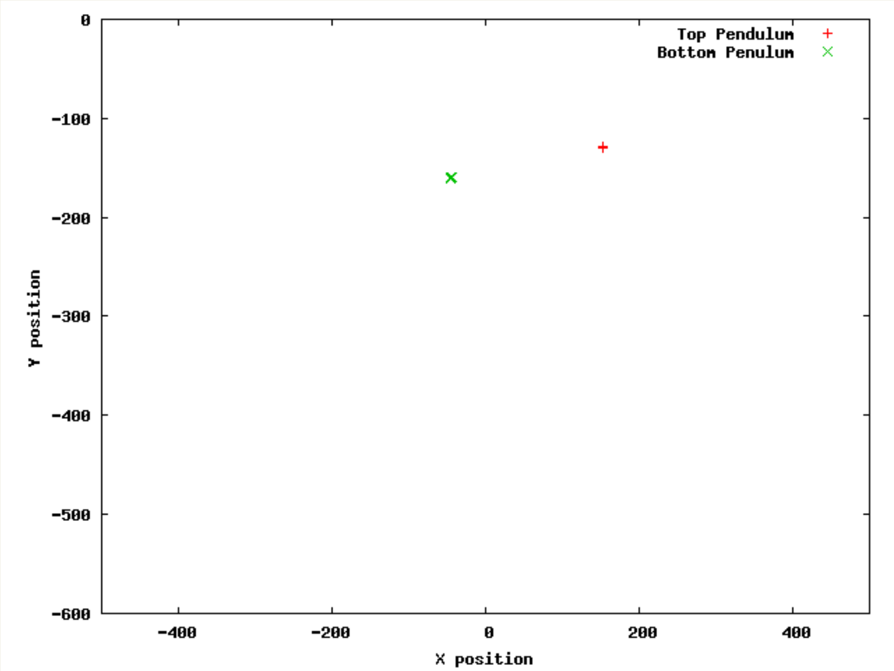
- Just through a slight variance in inputs the following two png images were produced.
- Such severe changes in outcomes clearly demonstrates the chaotic nature of the Lorenz Attractor





Final Conclusions

- Numerical methods are a patently powerful tool, especially when there is no analytical solution as in this case
- Chaos can be adequately demonstrated through numerical methods using C coding



Useful sources:

- Introduction to Chaos by Nagashima and Baba
- Chaos and Fractals by Peitgen, Jurgens, and Saupe
- Wikipedia