The Lorenz Attractor Interactive Investigatory Tool
MATH 480A Final Project

Alex Arslan

June 3rd, 2011

1 Introduction

Professional mathematicians, students, and even the curious layperson often wish to study the trajectory of the beautiful and mysterious Lorenz attractor. Or perhaps they just want to look at pretty pictures. Nonetheless, there is a shortage of truly interactive tools for plotting the Lorenz attractor. Often one must code it from scratch, editing the raw code to produce different output. But no longer! Harnessing the mathematical and graphical capabilities of Sage, the Lorenz Attractor Interactive Investigatory Tool provides a powerful yet approachable and intuitive interface for studying the behavior of the Lorenz attractor.

2 Purpose

The purpose of the Lorenz Attractor Interactive Investigatory Tool is to provide Sage users with a convenient and aesthetically pleasing way to study the Lorenz attractor at various time points, for various values of the parameters and initial conditions. The resulting plot can be used in academic papers or in homework assignments, wherever a quality plot of the Lorenz attractor is needed.

3 Background

In 1963, Edward N. Lorenz, a mathematician and meteorologist at MIT, published a paper called "Deterministic Nonperiodic Flow" in which he analyzed finite systems of deterministic ordinary differential equations intended to represent forced dissipative hydrodynamical systems. He focused primarily on the nonperiod solutions of such systems, since nonperiodic trajectories are representations of deterministic nonperiodic flow. [1]

Consider a system describing the flow in a layer of liquid with uniform depth H where the difference in temperature between the upper and lower surfaces is held steady at ΔT. If all motion is parallel to the xz-plane with no variation in the y direction, the equations governing the system can be written as

$$\frac{\partial}{\partial t}\nabla^2\psi = -\frac{\partial(\psi, \nabla^2\psi)}{\partial(x,z)} + \nu\nabla^4\psi + g\alpha\frac{\partial\theta}{\partial x} \tag{1}$$

$$\frac{\partial}{\partial t}\theta = -\frac{\partial(\psi,\theta)}{\partial(x,z)} + \frac{\Delta T}{H}\frac{\partial\psi}{\partial x} + \kappa\nabla^2\theta. \tag{2}$$

Here, ψ is a stream function for the two-dimensional motion, defined such that the velocity components $\vec{u} = \langle u, w\rangle$ of the fluid motion are $u = \frac{\partial\psi}{\partial z}$ and $w = \frac{\partial\psi}{\partial x}$. [3] Further, θ is the difference in temperature from that occuring in a state of no convection, g is acceleration due to gravity, α is the coefficient of thermal expansion, ν is the kinematic viscosity, and κ is the thermal conductivity. [2]

Let $R_a = g\alpha H^3\Delta T\nu^{-1}\kappa^{-1}$, called the Rayleigh number, and let $R_c = \pi^4 a^{-2}(1+a^2)^3$, a critical value for

the Rayleigh number. Rayleigh found that periodic solutions of the form

$$\psi = \psi_0 \sin\Big(\frac{\pi a x}{H}\Big) \sin\Big(\frac{\pi z}{H}\Big)$$
$$\theta = \theta_0 \cos\Big(\frac{\pi a x}{H}\Big) \sin\Big(\frac{\pi z}{H}\Big)$$

would grow when $R_a > R_c$. [1] [3] In Lorenz's study, he made an approximation immediately in the beginning by truncating a series to 3 terms. Accordingly, he ended up with the equations

$$a(1+a^2)^{-1}\kappa^{-1}\psi = X\sqrt{2}\sin\Big(\frac{\pi a x}{H}\Big)\sin\Big(\frac{\pi z}{H}\Big) \tag{3}$$

$$\pi R_c^{-1} R_a \Delta T^{-1}\theta = Y\sqrt{2}\cos\Big(\frac{\pi a x}{H}\Big)\sin\Big(\frac{\pi z}{H}\Big) - Z\sin\Big(\frac{2\pi z}{H}\Big), \tag{4}$$

where $X = X(t)$, $Y = Y(t)$, and $Z = Z(t)$ are functions of time. [1]

When equations (3) and (4) are subtituted into equations (1) and (2), the following system of nonlinear ordinary differential equations arises:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}' = \begin{bmatrix} \sigma(y-x) \\ x(\rho - z) - y \\ xy - \beta z \end{bmatrix}$$

where $\sigma = \frac{\nu}{\kappa}$ is the Prandtl number, $\rho = \frac{R_a}{R_c}$, and $\beta = \frac{4}{1+a^2}$. [1] The solution to this system has come to be known as the Lorenz oscillator, while the long-term behavior of the solution has come to be known as the Lorenz attractor.

For certain values of the parameters, the system exhibits chaotic behavior. That is, the solution is nonperiodic. For this reason, it is of interest to mathematicians who study chaotic systems and maps. It is also of interest to physicists and meteorologists for its applications to fluid dynamics, as was the original intent of Lorenz.

4 Algorithm

First, a function `lorenz_att` is created. This function solves the system defining the Lorenz oscillator for a given set of parameters and initial values along with a range of time using SciPy's `odeint` function from the `integrate` module. The result is then plotted using MatPlotLib's `pyplot` module, as well as the `Axes3D` function from the `mplot3d` module of the MatPlotLib toolkit library. This produces a high-quality static image of the Lorenz attractor in 3-space.

Next, a function `lorenz_attractor` is defined, which takes no arguments. It uses Sage's `@interact` function decorator applied to a function called `doit`. This function defines a static text control which displays "THE LORENZ ATTRACTOR" roughly centered over the interactive output. It then defines another static text control which just says "Parameters" for use as a label for the subsequently defined user input boxes for each parameter. The parameters are labeled "sigma," "rho," and "beta," and have respective default values 10, 28, and 8/3, which corresponds to the widely known chaotic image of the Lorenz attractor. A static text control "Initial conditions" is specified to serve as a label for the then defined user input boxes for the initial values x_0, y_0, and z_0, which are labeled as such. The initial values default respectively to 1, 0, and 1.05, again corresponding to the well-known image of the Lorenz attractor. Lastly, a slider for the observed time point is created, which goes from 1/10 (time 0 is not defined) to 100 by steps of 0.01. The default is 100.

The function `doit` passes all of these interactive controls to the `lorenz_att` function, which returns the Lorenz Attractor Interactive Investigatory Tool.

5 Code

```
def lorenz_att(coeffs,initial,time):
```

```
r"""
The solver and plotting tool for the Lorenz oscillator.

INPUT:

- A list of length 3 defining the values of the coefficients sigma, rho, and beta
- A list of length 3 defining the initial values of x, y, and z (x0, y0, z0)
- A list of length 3 defining the start, end, and step size for the time points for
  which to compute the solution

OUTPUT:

- An n x 3 matrix of solutions, where n is the end time point.
- The first column corresponds to the x value of the solution.
- The second column corresponds to the y value of the solution.
- The third column corresponds to the z value of the solution.

EXAMPLES:

Basic calling without output (too long)::

sage: lorenz_att([10,28,8/3],[1,0,1.05],[0,100,0.01])

ALGORITHM:

- Uses SciPy and MatPlotLib

.. NOTE::

    This function is not intended to be called on its own, as there is
    no error handling for incorrect or invalid input. This is made
    impossible by using the later-defined lorenz_attractor function.
    Adding error checking to this function would only decrease the
    overall performance of the tool.

.. WARNING::

    Be very careful when using this function on its own. As previously
    mentioned, there is no error checking, so incorrect input will
    result in confusing errors.

AUTHORS:

- Alex Arslan (2011-5-20)
"""

# Import necessary packages
import matplotlib as mpl
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import scipy
from scipy.integrate import odeint
```

```
    # Initialize plot
    fig = plt.figure()
    ax = Axes3D(fig)

    # Define the system of nonlinear ODEs
    def lor(incon,tt):
        x = incon[0]
        y = incon[1]
        z = incon[2]
        sigma = coeffs[0]
        rho = coeffs[1]
        beta = coeffs[2]
        x_prime = sigma*(y - x)
        y_prime = x*(rho - z) - y
        z_prime = x*y - beta*z
        return [x_prime, y_prime, z_prime]

    # Define initial conditions and time
    init = initial
    t = scipy.arange(time[0],time[1],time[2])

    # Solve the system
    solution = odeint(lor,init,t)
    X = [i[0] for i in solution]
    Y = [i[1] for i in solution]
    Z = [i[2] for i in solution]

    # Plot it!
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_zlabel('z')
    ax.plot(X,Y,Z)
    plt.savefig('lorenz.png')

def lorenz_attractor():
    r"""
    The interactive Lorenz Attractor investigatory tool.

    INPUT:

    - None

    OUTPUT:

    - The interface for plotting the Lorenz attractor

    EXAMPLES:

    Basic calling of the function::

    sage: lorenz_attractor()

    ALGORITHM:
```

```
- Uses Sage's @interact decorator
- Calls the lorenz_att function

.. NOTE::

    The purpose of this function is to explore the trajectory of the particle described
    by the solution to a particular system of nonlinear ordinary differential equations
    for various values of the parameters. Also editable are the initial conditions on
    x, y, and z, as well as the end time in which the particle has been in motion. This
    interactive tool is intended only for graphical analysis of the Lorenz attractor.
    No governing mathematics are displayed.

.. WARNING::

    No plot is displayed when x0 = y0 = z0 = 0, as the origin is not a valid point for
    the solution to the system.

AUTHORS:

- Alex Arslan (2011-05-20)
"""
@interact
def doit(title=text_control('                          THE LORENZ ATTRACTOR'),
paramtitle=text_control('Parameters'),
sigma=input_box(default=10,label='sigma ='),
rho=input_box(default=28,label='rho ='),
beta=input_box(default=8/3,label='beta ='),
initialtitle=text_control('Initial conditions'),
x0=input_box(default=round(1,2),label='x0 ='),
y0=input_box(default=round(0,2),label='y0 ='),
z0=input_box(default=round(1.05,2),label='z0 ='),
t1=slider(vmin=1/10,vmax=100,step_size=round(0.01,2),default=100,label='End time')):
    return lorenz_att([sigma,rho,beta],[x0,y0,z0],[0,t1,0.01])
```

References

[1] Edward N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, 20, January 1963.

[2] Barry Saltzman. Finite Amplitude Free Convection as an Initial Value Problem–I. *Journal of the Atmospheric Sciences*, 19, May 1962.

[3] Eric W. Weisstein. Lorenz Attractor. http://mathworld.wolfram.com/LorenzAttractor.html. From Mathworld–A Wolfram Web Resource.