Phys 416 – Chapter 3b Problems

[Due 1 PM Tuesday, March 3, 2020]

Note: Please mark the folder you turn the programs in as chapter3b

This unit will look at some non-linear dynamics models.

The Lorenz Model

In the early 1960's Ed Lorenz formulated a simple model of global weather in order to study its nonlinear behavior. The simple 3-variable Lorenz model is written as

$$\frac{dx}{dt} = \sigma(y - x)$$
$$\frac{dy}{dt} = rx - y - xz$$
$$\frac{dz}{dt} = xy - bz$$

In brief, x measures the state of convection overturning, y and z measure the horizontal and vertical temperature gradients respectively. The parameters σ and b depend on the fluid properties and the geometry of the system; we will use σ=10 and b=8/3. The parameter r is proportional to the applied temperature gradient. The program `Lorenz` solves the coupled set of ODE's using the adaptive Runge Kutta method. The function `lorzrk` specifies the right-hand side of equations 1.

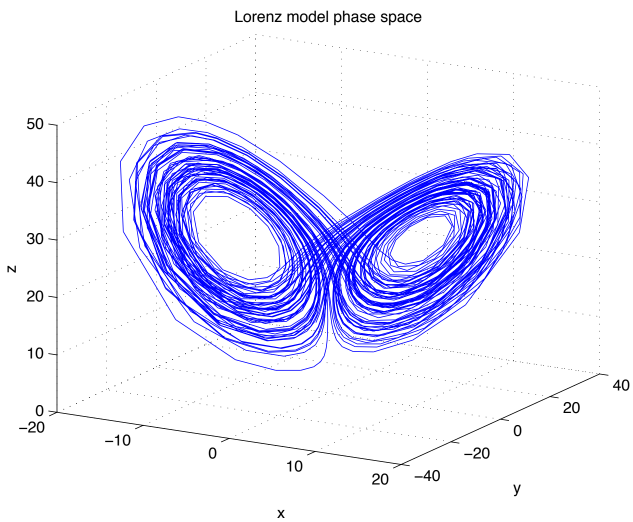


Figure 3.13

2. [Chapter 4, exercise 2] Derive the steady states for the Lorenz model and show that they are $x^* = y^* = z^* = 0$ and $x^* = y^* = \pm\sqrt{b(r-1)}$, $z^* = r - 1 \; if \; \sigma \neq 0$.
What are the steady states when $\sigma = 0$? Plot these points along with the 3 dimensional plots in the Lorenz program.

[Turn in program and sample plots.]

25. [Chapter 3, exercise 25]. One characteristic of chaotic dynamics is sensitivity to initial conditions. Using `rk4`, write a non-adaptive version of the `lorenz` program that simultaneously computes the trajectories for 2 different initial conditions. Use the initial conditions that are very close together (e.g. [1 1 20] and [1 1 20.001] and $r = 28$). Plot the distance between these trajectories as a function of time, using both normal and logarithmic scales. What can you say about how the distance varies with time? [Computer]
Hint: Use the original `lorenz` program to determine the minimum timestep `tau` and use that for your program.

[Turn in program and sample plots.]

22. [Chapter 3, exercise 22]. The following set of nonlinear ODE's is known as the Lotka-Volterra model:

$$\frac{dr}{dt} = a\left(1 - \frac{r}{b}\right)r - crf$$

$$\frac{df}{dt} = -af + crf$$

where *a, b, and c* are positive constants.

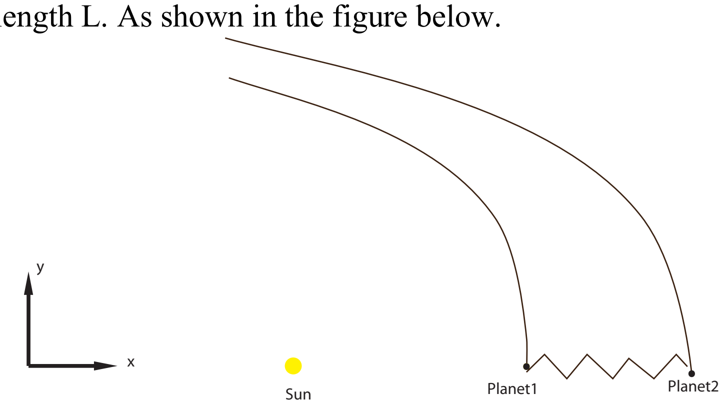
(a) These equations model a simple ecological system of predators and prey. For example, the variables r and f represent the number of rabbits (r) and foxes (f) in a forest. Describe the physical meaning of the parameters. Compute the steady state solutions for the rabbit and fox population. *[Pencil]*

(b) Write a program using the adaptive Runge Kutta to compute the trajectory (r(t), and f(t)), and plot f(t) versus r(t) for a variety of initial conditions, using a =10, b=10^6 and c =0.1. Take $r(0)$ >0, $f(0)$>0, since the number of animals should be positive. What happens when you start the program around the computed steady state numbers? *[Computer]*

[Turn in program and sample plots.]

Two planets attached to a spring.

A. For this problem, we will revisit briefly orbital mechanics with a twist, or rather a spring. Imagine we have 2 planets approaching the sun that are connected to each other by a spring of constant k and rest length L. As shown in the figure below.



Assume initially the planet1 is 1 au from the sun, and planet2 is 2 au, when they cross the y=0 plane, they are both moving upwards and both are moving on unbound parabolic orbits. Solve the equations of motion by modifying one of your programs, assume the only gravitational force is from the sun, but there is an addition force from the spring. For example, planet2 will feel the following force

$$\vec{F_2} = -k(|\vec{x_2} - \vec{x_1}| - L)\frac{(\vec{x_2} - \vec{x_1})}{|\vec{x_2} - \vec{x_1}|}$$

(A similar equation can be written down for planet 1 with the indices interchanged.) Assume each planet has mass =1(negligible gravitational force) and the rest length of the spring L = 1 au. For certain values of k, it turns out that this unbound orbit can be turned into a bound, but chaotic orbit by transferring some of the energy into the spring. Find a value for k where this happens. Plot the resulting trajectory, and show that your program conserves energy. Use the rka method. [Computer]

[Turn in program and sample plots.]

The Double Pendulum

19. [Chapter 3, Exercise 19] The double pendulum, as shown in the figure 3.11 of the text below has a Lagrangian of the form:

$$L = \frac{m_1 + m_2}{2} L_1^2 \dot{\theta}_1^2 + \frac{m_2}{2} L_2^2 \dot{\theta}_2^2 + m_2 g L_2 cos\theta_2 + m_2 L_1 L_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) + (m_1 + m_2) g L_1 cos\theta_1$$

Where $\dot{\theta} \equiv \frac{d\theta}{dt}$.

(a) Use the Lagrangian to compute the equations of motion. *[Pencil]*
(b) Write a program that uses the adaptive Runge-Kutta method to simulate the motion of the double pendulum. Take g=9.81 m/s^2, $m_1=10\ m_2$, and $L_1=L_2$=0.1; compute examples of the motion for various initial conditions. Make an animation of the motion of the pendulum.
(c) Plot the Poincare surface of section plot by recording θ_1 and $\dot{\theta_1}$ every time θ_2 goes through 0. Make 2 plots, one for initial conditions $\theta_1 = \theta_2 = 10^\circ$ and the other $\theta_1 = \theta_2 = 45^\circ$ and compare the results. You will need to run the code for around 50,000 steps and make sure your error tolerance is small enough ($\sim 10^{-6}$).
Hints:

- See notes called "Lagranges Equations.pdf" for hints on how to compute the equations of motion from the Lagrangian.
- You can setup the equations in (a) and solve them within MATLAB/python.
- You can make an animation of the motion by using the `drawnow` command in MATLAB (`draw()` in python, followed by a `pause()` statement).
- Be sure to check to see if your program is working by checking for the conservation of energy.
- Turn in program and sample plots.

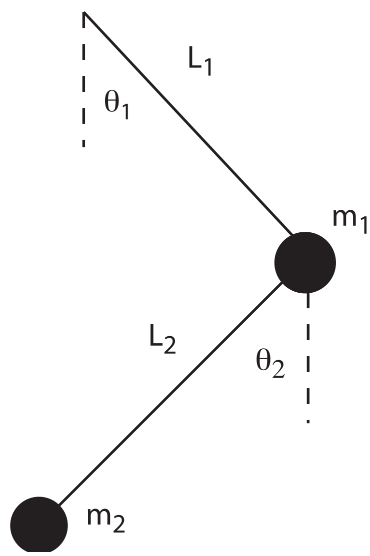


Figure 3.11

Optional Extra Exercise – 1 point extra credit

ECA. For a series of *r* from 20 to about 200, record the y-location of the trajectory of the Lorenz model when it crosses the *x*=0 plane. Make a scatter plot of these *y*-locations as a function of *r*; what you should get is a figure known as a bifurcation diagram.

- Run the code for about 10000 iterations (use less if your computer is slow) and only start recording after about 5000 iterations; this eliminates artifacts due to the initial conditions.
- While you are testing your program, use a smaller number of iterations, as this program may take a while to run.
- Use a starting point as [1,1,20].
- Set r to be from 20 to 200 in steps of 1, (i.e., *r*=20:1:200, you could use a finer resolution depending on how fast your computer is.) [*Computer*]
- Turn in sample plots and program.