PHZ4151C: Exercise 9

Computational Physics Lab
Due Friday April 12

For each problem you will write one or more python programs. These programs should follow the Python 2.7.x coding and formatting conventions outlined for our course. You must hand in copies of the programs and outputs as prescribed in each problem.

In addition, you must submit all Python programs as an archive tgz file via email to phz4151c@hadron.physics.fsu.edu. Place copies of only your Python programs in a directory called <last_name>-exercise8/where <last_name> is your last name and use similar commands as provided in earlier exercises for submission.

1. The Lotka–Volterra equations: The Lotka–Volterra equations are a mathematical model of predator–prey interactions between biological species. Let two variables `x` and `y` be proportional to the size of the populations of two species, traditionally called "rabbits" (the prey) and "foxes" (the predators). You could think of x and y as being the population in thousands, say, so that `x = 2` means there are 2000 rabbits. Strictly the only allowed values of `x` and `y` would then be multiples of 0.001, since you can only have whole numbers of rabbits or foxes. But 0.001 is a pretty close spacing of values, so it's a decent approximation to treat `x` and `y` as continuous real numbers so long as neither gets very close to zero.

In the Lotka–Volterra model the rabbits reproduce at a rate proportional to their population, but are eaten by the foxes at a rate proportional to both their own population and the population of foxes:

$$\frac{dx}{dt} = \alpha x - \beta x y$$

where α and β are constants. At the same time the foxes reproduce at a rate proportional the rate at which they eat rabbits—because they need food to grow and reproduce—but also die of old age at a rate proportional to their own population:

$$\frac{dy}{dt} = \gamma x y - \delta y$$

where γ and δ are also constants.

a) Write a program to solve these equations using the fourth-order Runge–Kutta method for the case $\alpha = 1$, $\beta = \gamma = 0.5$, and $\delta = 2$, starting from the initial condition `x = y = 2`. Have the program make a graph showing both x and y as a function of time on the same axes from `t = 0` to `t = 30`. (Hint: Notice that the differential equations in this case do not depend explicitly on time t—in vector notation, the right-hand side of each equation is a function f (r) with no t dependence. You may nonetheless find it convenient to define a Python function f(r,t) including the time variable, so that your program takes the same form as programs given chapter 8. You don't have to do it that way, but it can avoid some confusion. Several of the following exercises have a similar lack of explicit time-dependence.)

b) Describe in words what is going on in the system, in terms of rabbits and foxes.

For full credit turn in a printout of your finished program and graph from part a) and the discussion from part b).

2. The Lorenz equations: One of the most celebrated sets of differential equations in physics is the Lorenz equations:

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

where σ, *r*, and *b* are constants. (The names σ, *r*, and *b* are odd, but traditional—they are always used in these equations for historical reasons.)
These equations were first studied by Edward Lorenz in 1963, who derived them from a simplified model of weather patterns. The reason for their fame is that they were one of the first incontrovertible examples of *deterministic chaos*, the occurrence of apparently random motion even though there is no randomness built into the equations.

a) Write a program to solve the Lorenz equations for the case $\sigma = 10$, $r = 28$, and $b = 8/3$ in the range from $t = 0$ to $t = 50$ with initial conditions $(x, y, z) = (0, 1, 0)$. Have your program make a plot of y as a function of time. Note the unpredictable nature of the motion. (Hint: If you base your program on previous ones, be careful. This problem has parameters *r* and *b* with the same names as variables in previous programs—make sure to give your variables new names, or use different names for the parameters, to avoid introducing errors into your code.)
b) Modify your program to also produce a plot of *z* against *x*. You should see a picture of the famous "strange attractor" of the Lorenz equations, a lop-sided butterfly-shaped plot that never repeats itself.

For full credit turn in a printout of your finished program and the two graphs which the program generates.

3. The Nonlinear Pendulum: Building on the results from Example 8.6 in chapter 8, calculate the motion given the nonlinear equations of motion for a pendulum:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}\sin(\theta)$$

a) Write a program to solve the two first-order equations obtained from the above second-order equation, using the fourth-order Runge–Kutta method for a pendulum with a L=10 cm arm. Have the program initialize the starting angle from a value obtained on the command line(i.e. use the *sys* module). Use your program to calculate the evolution of the angle θ for small starting angles over several periods of the pendulum. Use $\theta = 10^\circ$ from the vertical as your small starting angle. Make a graph of θ as a function of time and an additional graph of $d\theta/dt$ as a function of θ(i.e. the velocity of the oscillator against its position). The later plot is called a phase space plot.

b) Use your program to calculate the angle evolution θ(t) over the range of several pendulum periods with the angle initially released from a standstill at $\theta = 179^\circ$ from the vertical. Make a graph of θ as a function of time and a graph of $d\theta/dt$ as a function of θ.

For full credit turn in a printout of your finished program together with your plots from part a) and b).

4. The Character of a Short Spring: Let's examine realistic effects of a short spring system limited in its stretching length. For example, a spring made of several windings at most would stretch to the unwound length of the spring(assuming the wire itself does not stretch). One could expect some non-linearity between the force applied to the spring and the stretch of the spring. The spring constant, instead of being constant as in a simple harmonic oscillator, could have a quadratic dependence. That is, k would be proportional to x^2 giving the nonlinear equation:

$$\frac{d^2x}{dt^2} + Ax^3 = 0$$

A realistic system will also have a frictional term proportional to the velocity. To over come losses due to friction, and to make the system more interesting at large time values, let's have the system periodically driven. The resulting equation of motion is given below:

$$\frac{d^2x}{dt^2} + b\frac{dx}{dt} + Ax^3 = B\cos(\omega_d t)$$

This equation describes what is known as Duffing's oscillator, and it is know to exhibit a wide variety of behaviors. A, B, b, and ω_d are adjustable parameters. For this study, let's set A=1 and ω_d =1; though they need not be.

a) Write a program to solve the two first-order equations obtained from the above second-order equation, using the fourth-order Runge–Kutta method. Develop your program to graph the position x versus time for 25 periods of the driving frequency ω_d and to graph the phase space (i.e. x versus dx/dt). Use as starting values for t =0, x = 3 and dx/dt=0. Test your program with parameters (B, b) = (7, 6). You should see the motion transition to a steady state harmonic motion.

b) Study the system for various choices of the following parameters (B, b) = (7, 6), (7,0.6), (10, 0.05), and (7, 0.01). For each set of parameters plot the position vs time and the phase space. After the initial transient motion, describe the motion. Is it a steady motion? Do you see any bifurcation(or splitting) of the motion? Are there any solutions which are clearly not a steady motion?

c) A measurement of the divergence is attained with the use a device known as the Poincare' section. To generate a Poincare' section, one asks what the phase space looks like at the same phase angle ($\omega_d t + \varphi$) of the driving frequency, which for our short spring system, is the time corresponding to a multiple of 2π(remember ω_d =1) plus an arbitrary phase. In other words, Poincare' sections are phase space plots with all points erased save those that correspond to a time value of $n2\pi$ of the forcing period.

Write a new program (copy and modify the existing program) to create a Poincare' section for the parameters (B,b) = (7, 0.01), at $n2\pi$. Since the Poincare' section uses only one data point per driving cycle, to receive good resolution many steps are needed(Usually one to two orders of magnitude over that required in the earlier plots). (Hint: one may want to choose a time step equivalent to one degree of the driven phase angle so that one cycle equals 360 steps .)

Chaos, chaotic behavior, is described as arising from phase space operations of stretching and folding. Because energy conservation confines the motion to a finite volume in phase space, the chaotic system cannot diverge exponentially forever. The phase space motion must at some time pass near or on prior states; this is know as stretching (exponential divergence in phase space) and folding (confinement in phase space) and it is responsible for the picturesque fractal behavior that is observed in the Poincare' section.

For full credit turn in a printout of your programs from part a) and part c) along with your plots from part b) and c) and the discussion from part b).