PH 120 Project # 2: Pendulum and chaos

Due: Friday, January 16, 2004

In PH109, you studied a *simple pendulum*, which is an effectively massless rod of length ℓ that is fixed at one end with a small mass m at the other end. There are two forces on the mass: the tension in the rod and gravity. But the tension in the rod is radial, so it does not exert a torque around the pivot point where the rod is fixed. The torque due to gravity is $mg\ell\sin\theta$, where g is the acceleration of gravity and θ is the angle of the pendulum (taking the angle to be zero at the bottom). Thus Newton's Law $\tau = I\alpha$ becomes

$$-mg\ell\sin\theta(t) = I\frac{d^2\theta(t)}{dt^2} = m\ell^2\frac{d^2\theta(t)}{dt^2}$$

where the moment of inertia I is just $m\ell^2$ for a point mass. Defining $\omega = \sqrt{\frac{g}{\ell}}$, this simplifies to

$$\frac{d^2\theta(t)}{dt^2} = -\omega^2\sin\theta(t)\,.$$

To solve this equation, you *assumed* that $\theta(t)$ was always small, so that $\sin\theta \approx \theta$. Then the equation becomes

$$\frac{d^2\theta(t)}{dt^2} = -\omega^2\theta(t)$$

and we have simple harmonic motion with angular frequency ω. The general solution is $\theta(t) = \theta_0\cos\omega t + \frac{\omega_0}{\omega}\sin\omega t$, where θ_0 is the angle at time $t = 0$ and ω_0 is the angular velocity $\frac{d\theta}{dt}$ at $t = 0$. (If you would like more details than I've provided in this lightning review, see page 372-3 of Wolfson & Pasachoff.)

Of course, a real pendulum is not restricted to oscillate only through small angles. Your job on this problems set will be to analyze the pendulum *without* making this assumption.

1. The mathematical problem here is a *differential equation*. To remind you, a differential equation is an equation for a *function* — in our case $\theta(t)$. Solving a differential equation means taking a relationship between a function and its derivatives and finding function (or a set of functions) that satisfy this relationship. Just as it can solve ordinary equations, Mathematica can also solve differential equations. Before attempting the pendulum problem, let's start with some simpler examples.

   (a) A simple example of a differential equation is

   $$f'(t) = -kf(t)$$

which describes the decay of radioactive isotopes like ^{14}C — the rate at which the isotope disappears is proportional to how much you have at a given time.

Even if you know how to solve this equation, use Mathematica to find the solution. Pick a value of k and an initial condition $f[0]$ and generate a plot of a particular solution as a function of t. Estimate graphically how long it takes for half the material to disappear. Check your answer analytically.

Hint: Just as it did with `Solve`, when Mathematica solves a differential equation, it generates a *rule* describing the solution (or solutions). So you will need to apply this rule to substitute the result back into `f[t]`. Solutions to differential equations involve unknown constants (for example, any solution to this equation can be multiplied by a constant to yield another solution), which Mathematica represents as `C[1]`, `C[2]`, etc. In this case, the constant represents how much radioactive material we started with (to see this, evaluate the solution at $t = 0$).

(b) A slightly more complicated example is the system of differential equations

$$r'[t] = 1 - w[t] \qquad w'[t] = r[t] - 1$$

describing populations of rabbits and wolves: Rabbits multiply, so their population increases (the first term in the first equation), but wolves eat them, reducing their population (the second term in the first equation). When there are a lot of rabbits around the wolf population grows (the first term in the second equation), but otherwise they die (the second term in the second equation). Solve these differential equations with Mathematica. Then pick some initial conditions and make a plot of both populations as functions of t on the same graph.

(This is not actually a very realistic model, since values can go negative, but it's a simple starting point. To make it even plausible, the right-hand side of the first equation should be multiplied by $r(t)$ and the right-hand side of the second equation should be multiplied by $w(t)$, but we want to start with the simplest system we can.)

Hint: You can use `FullSimplify[]` to make the result Mathematica returns somewhat nicer.

(c) Continuing with the rabbits and wolves example, use `ParametricPlot` to generate a graph in which the number of rabbits is on the x-axis, and the number of wolves is on the y-axis, with time as the parameter of the path in the xy-plane. Try several initial conditions and interpret what the graphs are telling you about population dynamics in this model. Find the period of any oscillatory behavior you find. Find the steady-state solution, where the number of rabbits and wolves does not vary with time.

2. Next, we would like to analyze the textbook result for small angles and develop some graphical tools to help us understand the solution.

(a) Use Mathematica's `DSolve` function to solve the differential equation in the small angle approximation. Note that in the result, the initial conditions above are replaced by the unknown constants `C[1]` and `C[2]`.

(b) Fix a particular value of ω (choose any reasonable value you want and keep it fixed throughout this problem set), and plot $\theta(t)$ as a function of time for a full cycle in the following cases:

   i. Motion that starts at rest with $\theta(0) = \theta_0$. (Pick some reasonable θ_0.)

   ii. Motion that starts at $\theta = 0$ with an initial angular velocity ω_0. (Pick some reasonable ω_0.)

(c) We have found a set of solutions, which are functions of t involving two unknown constants. Reduce this to one constant by assuming that the pendulum starts at $\theta = 0$, and then form a function that takes in a value for the initial angular velocity at ω_0 and returns the corresponding function of t. In other words `motionFromZero[1.5]` should return the solution as a function of t in which the pendulum starts with $\theta(0) = 0$ and $\theta'(0) = 1.5$.

(d) A very useful tool for analyzing differential equations is a *phase space plot*. This is a plot with position (in our case θ) on the x-axis and velocity (in our case $d\theta/dt$) on the y-axis. The motion of the system is described by a path through the xy-plane parameterized by time. Define a function that takes an initial angular velocity and creates a `ParametricPlot` in the phase space of the pendulum's motion of the solution starting from $\theta(0) = 0$ with that initial angular velocity.

(e) Use `Show` and `Table` to superimpose several of these plots on the same graph. The result is a picture of the possible motions of the pendulum (in the small angle approximation).

   Hint: If you want to avoid displaying all the individual plots but still see the final superposition of the results, set `DisplayFunction -> Identity` for each individual plot and then override this with `DisplayFunction -> $DisplayFunction` in the `Show` command.

(f) At any given time, the pendulum has kinetic energy $\frac{1}{2}m\ell^2\left(\frac{d\theta(t)}{dt}\right)^2$ and potential energy mgh, where $h = \ell(1 - \cos\theta(t))$ is the height of the mass as compared to its position when $\theta = 0$. In the small angle approximation, $\cos\theta \approx 1 - \frac{\theta^2}{2}$ (as a check on this expression, note that it is consistent with $\cos 0 = 1$ and $\frac{d}{d\theta}\cos\theta = -\sin\theta$ in the small angle approximation). Thus the potential energy becomes $\frac{1}{2}mg\theta(t)^2$ in the small angle approximation, and the total energy becomes $\frac{m\ell^2}{2}\left(\left(\frac{d\theta(t)}{dt}\right)^2 + \omega^2\theta(t)^2\right)$. Choosing a particular numerical value for $I = m\ell^2$, take a particular trajectory and plot the energy as a function of time. Does it behave as you expect?

(g) For the same value for the moment of inertia, consider a variety of different initial conditions. Generate an ordinary plot showing the area enclosed by a trajectory as a function of the energy on that trajectory. Use this plot to guide you to an analytic formula relating these two quantities.

Hint: the area of an ellipse is πab, where a and b are the lengths of the semimajor and semiminor axes respectively. (This makes sense since for a circle, where $a = b = r$, it reduces to the familiar πr^2.)

3. Next we would like to do the case without approximation.

   (a) Use `DSolve` to find the solution to the full pendulum problem. It should be able to solve the differential equation (ignore warnings about inverse functions) but it will come back with a result that is rather obscure (unless you know a lot more about elliptic integrals than I do).

   (b) What you should find two possibilities involving the `JacobiAmplitude` function, both with two unknown constants. Applying `FullSimplify` to your result, you should find that one of the constants just shifts the time t. We are free to set this to zero — this just corresponds to choosing what time we call $t = 0$.

   (c) Pick a value of the other initial condition and plot $\theta(t)$ and its derivative as functions of t for both solutions. Make sure what you are seeing makes sense physically.

   (d) As you did before, generate phase space plots of both solutions for a variety of initial conditions. Use the same type of approach as in the previous problem to generate many curves at once. Identify the region where the approximate picture is correct. You should see a second type of solution as well — interpret physically what it is doing.

4. We can make the problem more realistic by adding friction (damping). For example, this could be in the form of air resistance. The friction generates a torque proportional to the velocity, and in the opposite direction: it always slows the pendulum down, and the faster the pendulum is going, the stronger this effect will be. Thus the torque is of the form $\tau = -\mu\theta'(t)$, and our equation becomes

$$-mg\ell \sin\theta(t) - \mu\frac{d\theta(t)}{dt} = I\frac{d^2\theta(t)}{dt^2} = m\ell^2\frac{d^2\theta(t)}{dt^2}$$

Rescaling as before, The equation for $\theta(t)$ becomes

$$\frac{d^2\theta(t)}{dt^2} = -\omega^2\sin\theta(t) - \gamma\frac{d\theta(t)}{dt}$$

where $\gamma = \dfrac{\mu}{m\ell^2}$. With this addition, the differential equation no longer has a solution in terms of any of the special functions known by Mathematica (or anyone else, for that matter), so we will have to proceed numerically. Pick a reasonable numerical value for γ and perform the same sort of phase space analysis of the previous problem numerically, using `NDSolve` (note that some of the options are slightly different, since Mathematica needs extra data, such as the range of t, to do the numerical calculation; nonetheless, the basic structure of the command is the same as `DSolve`). Generate curves for several different combinations of initial positions and velocities (you might need to do each one on a separate plot to be able to see what is going on). Explain physically what is happening in your pictures, and make sure you choose a wide enough variety of initial conditions to see all the different kinds of behavior you would expect physically (or based on what you found on the previous problem).

5. Finally, we can add one more possibility to this analysis: Suppose we *drive* the pendulum with a periodic external torque of amplitude A and angular frequency ω_d, $\tau = A\cos\omega_d t$. Then the reduced equation becomes

$$\frac{d^2\theta}{dt^2} = -\omega^2 \sin\theta - \gamma\frac{d\theta}{dt} + f_0 \cos\omega_d t$$

where $f_0 = m\ell^2 A$.

Note: In this problem you will get much nicer results in many places if you use a lot of points, and let the pendulum run a long time. Start by getting your code working with just a few points, but for the final version try to generate some good pictures (suitable for submission to the Science is Art show!) by letting it run for a while (overnight should be plenty). If you need computer time for this purpose, please let me know.

   (a) We now have a lot of different possibilities to explore. To narrow things down, set $\gamma = 1/2$, $\omega = 1$, and $\omega_d = 2/3$. Rather than varying the initial conditions, in this problem, we will start by always taking $\theta(0) = \pi/2$ and $\theta'(0) = 0$. Instead, we would like to produce several separate phase space plots, each with different values of the amplitude of the driving force. We'd also like to make another modification: we will want to let this system run a long time (to see chaotic behavior). In principle, if the pendulum keeps spinning around in the same direction, θ will get bigger and bigger. Physically, of course, it's coming back to the same place: $\theta = 5\pi/2$ is the same as $\theta = \pi/2$. To see the patterns, we'd like our phase space plot to only cover the physical 2π range, which these two values mapped onto the same point. So you should use `Mod` to restrict your plot in this way. When you do so, however, you get a problem: When the angle jumps by 2π, `ParametricPlot` will draw an ugly line across your plot connecting the value near $-\pi$ to the value near $+\pi$. So instead of using `ParametricPlot` in this problem, you should generate a list of points to plot and use `ListPlot`. Write a function `phasePlot[f0]` that takes in a value for f and returns a phase space plot done this way. Investigate in particular the values $f_0 = 0.9$, $f_0 = 1.07$, and $f_0 = 1.15$, and describe physically what is happening in each of these cases. Pick some other values in between these and observe how one type of behavior changes into another. Also try some other nearby values and see what kinds of behavior you find. Be sure to let the graphs trace out a long enough time! You'll probably want to adjust the number of points that `NDSolve` uses in solving the differential equation.

(b) A *Poincaré section* is another useful tool for analyzing dynamical systems. It's rather like viewing the pendulum with a strobe light synchronized to the external force. Specifically, it's a scatter plot: Every time the external force goes through a complete cycle, we add a point in phase space representing the pendulum's current position and angular velocity. As in the previous problem, the plot should map equivalent values of θ to the same point. Also, since we are interested in describing the long-term behavior of the system, (in particular, whether it is periodic), let the pendulum run for a little while before you start recording points, so you don't get confused by transient features due to the initial conditions.

Define a function `poincarePlot[f0]` that generates the Poincaré section for a given value of f_0. Apply it to the cases you studied in the last problem. Explain how the plots you see correspond to the motion you described in the last problem. For the chaotic case, watch what happens as you add more and more points — you should see fractal behavior!

(c) If we vary the initial conditions, we can study the *basins of attraction*. (This is the only part of this problem in which you need to change the initial conditions.) This procedure is like what you did on the first project with `FindRoot`. For a particular value of f_0, what we would like to have is a `DensityPlot` in which each point is dark or light depending on whether its velocity after a fixed time is positive or negative. What you should do is to define a function `basinPlot[f0]` that generates such a plot for a given value of f_0. Generate a series of such plots and explain what is happening as f_0 approaches the chaotic point.

(d) The last of these tools is the *bifurcation diagram*. It just like a Poincaré section, except that we now have f_0 on the x-axis. In other words, for each value of f_0, you should let the pendulum run for a while, and then gather a collection of values of $\theta'(t)$ values by sampling each time the driving force returns to a particular point in its cycle. You place each on of these points on the graph, at position $(x = f_0, y = \theta'(t_n))$, and repeat this process for a range of f_0. Generate such a plot over the range of f_0 we have been considering, paying particular attention to the behavior as you approach the chaotic point. (You might find the `Flatten` command useful in manipulating your lists for `ListPlot`. Explain how your graph describes the onset of chaos.

6. (You don't have to hand in anything now for this part.) Based on your responses to the questionnaire, I'm setting up the third project in this course as a C++ project. Although I will present the salient features of C++ syntax that you will need, it will help to start now just getting familiar with the mechanics.

   If you prefer, you will also have the option to use Java instead — it also has all the features of C++ that I will use. You will just have to translate what I present into Java, but if you are comfortable with Java this should be straightforward. If you are very uncomfortable with both, you may also use Mathematica (but of course you will have to implement the algorithms we will study yourself, not by calling built-in functions).

   Assuming you are planning to use C++ or Java, make sure you can get started programming. For example, see if you can write a program that prints the numbers from 1 to 10, or something like that. I just want to make sure you're comfortable using a text editor, writing very basic syntax, calling the compiler, and running your program. I am happy to help and provide references. In the labs, we will use the gcc compiler, which comes with Linux and OS X, and can be downloaded for free for Windows at http://www.cygwin.com. (The default cygwin package provides you a command shell and a number of packages, including gcc.) Any other standard compilers (Microsoft, Borland, etc.) you might have access to are also fine. If you have any problem getting access to computers to meet your needs, please don't hesitate to ask me.