Physics 331 **Advanced Classical Mechanics** **Fall 2010**

Problem A

We will study the phenomenon of period doubling of the driven damped pendulum. For parameters, take $\omega = 2\pi$, $\omega_0 = 3\pi$, and $\beta = 3\pi/4$, and the initial conditions $\phi(0) = 0$ and $\dot{\phi}(0) = 0$.

(a) Start with $\gamma = 1.06$, and solve the equation with `NDSolve` over the time range $t = 0$ to 50. Store your solution in a variable, that is, `soln = NDSolve[ ...]`

Now you can plot the function via `Plot[ϕ[t] /. soln, {t, 0, 50}]`

Change the plot range to a narrower time window at sufficiently late times and determine whether the period is equal to 1, 2, 4, … (like in Fig. 12.8). Print out this plot.

(b) Now do the problem again, but with $\gamma = 1.078$. You may want to plot a horizontal line near the top or bottom of the oscillator's range to highlight differences. You can do this by adding a constant "function" to plot, i.e. `Plot[{ϕ[t] /. soln, -1.452}, {t, ...}]`.

(c) Now do the same for $\gamma = 1.081$.

(d) Now do the same for $\gamma = 1.835$.

Problem B

We will make a bifurcation diagram for the driven damped pendulum, I had originally intended to make a template, but I think it'll be more fun if you program your own. Here's the steps:

1. Define `eqn =` the differential equation, with $\omega = 2\pi$, $\omega_0 = 3\pi$, and $\beta = 3\pi/4$. Be sure to leave γ as an uspecified variable.

2. We want to grow a list of (γ, ϕ) pairs, so we need to start with an empty list. Type

   ```
   list = {}
   ```

3. Now we want to set up a for-loop to go from $\gamma = 1.06$ to $\gamma = 1.087$ in steps of size 0.0001. As an example, the for-loop syntax to take x from 1 to 10 in steps of 0.1 would be:

   `For[x=1, x<=10, x+=0.1,` *expressions*`]`

   Here *expressions* is as many mathematica expressions as you want, separated by semicolons (and carriage returns usually, for readability).

4. In the *expressions* part of the for-loop you will have three expressions. The first is the numerical solution command. Initially, just take the time range to be 0 to 20 until you have everything working. You need to set step limit to infinity, via

   ```
   soln = NDSolve[ ... {t, 0, 20}, MaxSteps → Infinity].
   ```

5. The second expression does the strobe light: it extracts a set of ϕ values at the driving period, using the `Table` function. Let me just give you the syntax:

   ```
   newlist = Table[{ γ, ϕ[t] /. soln[[1]] }, {t, 10, 20, 1}];
   ```

   This would extract the values from $t = 10$ to 20 in increments of 1.

6. The third expression adds the new (γ, ϕ) values stored in `newlist` to the ever-growing `list`, via the command

   ```
   list = Union[list, newlist]
   ```

7. Be sure to close up your for-loop with the closing square bracket and see if this runs. If it completes sucessfully, you can plot your data with

   ```
   ListPlot[list, PlotStyle→PointSize[Small]]
   ```

8. Your plot probably looks bad, but if it has a bunch of points with the horizontal range between $\gamma = 1.06$ and 1.087, you're probably in good shape. If not, be sure to empty the `list` variable again before re-trying.

9. Once it is looking decent, empty the list, increase the `NDSolve` solution out to time $t = 400$, and in the `Table` command, extract the values from $t = 300$ to $t = 400$.

   On a linux machine, this longer run takes around a minute. It might be a bit slower on Windows, but hopefully in that ballpark.

In principle, you should now have a beautiful bifurcation diagram. Please print out your whole notebook (perhaps after tidying it up) and not just the plot.