Winter Quarter 2019 – UCSB Physics 129L Homework 5 Due Monday, February 25, 5 pm (Extended Deadline)

Read carefully the instructions on the website on how to prepare your homework for turning it in to the TAs.
If your last name starts with A through N, send the homework to Jenny. Otherwise send it to Francesco.
The emails of the TAs are on the website.
Put the instructor in cc to the email.

Make sure to have your rpi updated to pick up the latest/greatest example programs from class.

Exercise 1

Let's calculate the period of oscillation of a pendulum without resorting to the small angle approximation. Let L be the length of the pendulum and M the mass, and take the zero of potential energy at the equilibrium position. If the angle between the pendulum and the vertical is θ the potential energy is $U(\theta) = MgL(1-\cos\theta)$. If the maximum angle of oscillation is α, conservation of energy gives:

$$\frac{1}{2}Mv^2 + MgL(1-\cos\theta) = MgL(1-\cos\alpha)$$

Since v is tangential, $v = L\frac{d\theta}{dt}$, so we get

$$\frac{d\theta}{dt} = \sqrt{\frac{2g}{L}(\cos\theta - \cos\alpha)}$$

Using the identity $\cos A = 1 - 2\sin^2\frac{A}{2}$ this can be written as

$$\frac{d\theta}{dt} = \sqrt{\frac{4g}{L}(\sin^2\frac{\alpha}{2} - \sin^2\frac{\theta)}{2}}$$

The time taken to go from $\theta = 0$ to $\theta = \alpha$ is one fourth of the period T. Therefore

$$\frac{T}{4} = \sqrt{\frac{L}{4g}} \int_0^{\alpha} \frac{d\theta}{\sqrt{\sin^2\frac{\alpha}{2} - \sin^2\frac{\theta}{2}}}$$

At this point we could integrate numerically. It is a bit messy since the integrand goes to ∞ at $\theta \to \alpha$. Instead we can substitute $\sin\beta = \frac{\sin\theta/2}{\sin\alpha/2}$ which after some algebra gives

$$T = \frac{2}{\pi} T_0 \int_0^{\frac{\pi}{2}} \frac{d\beta}{\sqrt{1 - k^2 \sin^2\beta}}$$

with $k = \sin\frac{\alpha}{2}$ and $T_0 = 2\pi\sqrt{L/g}$ is the period of the pendulum in the small angle approximation. This integral is an elliptic integral of the first kind. Integrate numerically this equation, using whatever method you like, to plot T/T_0 as a function of α. Plot α in degrees betwee 1^o and 90^o.
To check yourself, you might want to compare with the Python function that returns this integral for you, see
https://docs.scipy.org/doc/scipy/reference/generated/scipy.special.ellipk.html

Exercise 2

This is an exercise in creating bitmaps to show on the screen or to save in a file. For an example of an 8 bit color map, see
`/home/pi/physrpi/campagnari/python/colorPicture.py`.
We will now make a pretty picture. Consider the function $f(z) = z^2 + c$ where z is a complex variable and c is a given complex constant. We can build a series of complex numbers as $z_{n+1} = f(z_n) = z_n^2 + c$. Depending on the starting point z_0, the series may grow without bouds or remain bounded. Consider a pixel map of the complex plane ($z = x + iy$) with 640 horizontal pixels and 400 vertical pixel to represent the range $-1.6 \to 1.6$ in x and $-1 \to 1$ in y. Take $c = -0.79 + 0.156i$. Loop over all pixels, and for each pixel build up a series z_n starting from a z_0 taken at the center of the pixel. Fill the pixel map with the first value of n for which $|z_n| > 2$, but stop at $n = 255$ (8 bit color). Then show the map on the screen. This is the visualization of a "Julia set".
Hint: `z = complex(x,y)` defines the complex number $z = x + iy$.

Exercise 3

This is the same as Exercise 1 in Homework 4, except that the file is now a txt file: `/home/pi/physrpi/campagnari/python/mass.txt`
(Look up the numpy facility for reading a text file into a numpy array).
This data set is meant to summarize the result of an experiment where the entries in the text file are the invariant masses (in GeV/c^2) of electron positron pairs. An experimenter stares at this plot, knowing that there exists a particle A of mass 155 GeV/c^2 which can decay as $A \to e^+e^-$. She then notices the accumulation of entries around 150-160, and wonders whether she is looking at a statistical fluctuation or at the signature of the presence decays of A particle in her dataset. We will come back to this problem later in the quarter. (Incidentally, no such particle exists, as far as we know).

Exercise 4

You are reading data out of some hardware device with many channels that measure time and pulseheight. The data comes out to you in as a bunch of 16 bit integers. It is packed such that bit 0 to 3 encode the channel number, bits 4 to 7 encode the time, and bits 8 to 15 encode the pulse height. Write a program that takes the 16 bit integer and decodes the three pieces of information. To test it, prompt the user for a number, print out channel number, time, and pulse height.
Hint: lookup "python bitwise operations" in case you forgot what we briefly covered in class a few weeks ago.

Exercise 5

Use the bisection method to solve for $x \cos x = \frac{1}{2}$ near $x = 0.7$. Calculate your answer within 0.0001. The bisection method is a numerical method for finding a root of an equation $f(x) = 0$. If values a and b are found such that $f(a)$ and $f(b)$ have opposite signs and f is continuous on the interval $[a, b]$, then (by the Intermediate Value Theorem) the equation has a root in $[a, b]$. The method is to bisect the interval and replace it by either one half or the other, thereby closing in on the root.

Exercise 6

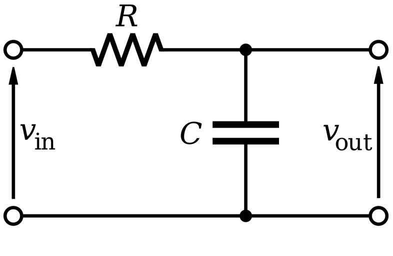
Consider the differential equation $dy/dx = f(x, y)$ with $y(x_0) = y_0$. You can iteratively build an approximate solution as $y_{n+1} = y_n + h \cdot f(x_n, y_n)$ and $x_{n+1} = x_n + h$, where h is a suitably small number. This is the Euler method. A better method is the Runge-Kutta method:

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ x_{n+1} &= x_n + h \\ k_1 &= h \cdot f(x_n, y_n) \\ k_2 &= h \cdot f(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \\ k_3 &= h \cdot f(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \\ k_4 &= h \cdot f(x_n + h, y_n + k_3) \end{aligned}$$

Now consider the RC circuit in the figure. This is a low-pass filter, you should have already seen this in lower division Physics (if you forgot, look it up!). The output voltage satisfies the differential equation

$$\frac{dV_{\mathrm{out}}}{dt} = \frac{1}{\tau}(V_{\mathrm{in}} - V_{\mathrm{out}})$$

where $\tau = RC$ is the time constant.



Take V_{in} to be a square wave:
$V_{in}(t) = +1\ V$ if $0 < t < 1$, $2 < t < 3$, $4 < t < 5$, ...
$V_{in}(t) = -1\ V$ if $1 < t < 2$, $3 < t < 4$, $5 < t < 6$, ... with $t > 0$ and t in μsec.
Use the Runge-Kutta method to calculate $V_{out}(t)$ with the boundary condition $V_{out}(t = 0) = 0$ for $0 < t < 100$ μsec. Setup your program so that you

will accept a value ot τ as an input from the keyboard. To display the results, remember that there will be a transient which dies out with a time constant of τ, followed by a steady state solution. Therefore provide two curves:

1. $V_{out}(t)$ for $0 < t < 10$ μsec
2. $V_{out}(t)$ for $90 < t < 100$ μsec

The first curve will show the transient going away, and the second curve will show the steady state solution (as long as $\tau <<$ 10 μsec).