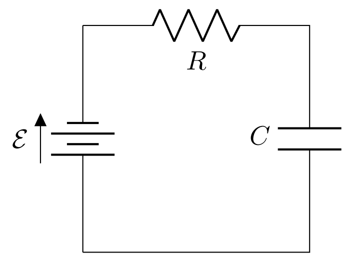
Physics 4B Lab by Peter Ho

Lab #6: Euler's Method with RC Circuits



From our previous investigation with combining a resistor R and capacitor C into a single circuit, we developed a description on the rise of charges to potential drops across these components. Imagining then, when a potential is applied across the circuit from a connected battery, current then flows through the resistor to the capacitor as it begins to accumulate charges. Kirchoff's law states that we can then express the potential drops across both components as

$$\mathcal{E} - iR - \frac{q}{C} = 0.$$

By noting that current i is expressed through the differential form of charge per unit of time as $i = dQ/dt$. This then yields a differential equation of the first order, or what is known as an **ordinary differential equation (ODE)**. Writing out further the expression above gives us

$$\frac{dQ}{dt} + \frac{q}{RC} - \frac{\mathcal{E}}{R} = 0,$$

and more importantly rewriting the notation as $dQ/dt = q'$, the expression evolves. That is, the ODE we obtain will have the expression

$$q' + \frac{1}{RC}q - \frac{\mathcal{E}}{R} = 0.$$

The solution to this differential equation is one that we can compute through implicit differentiation whose solution is

$$q(t) = C\mathcal{E}(1 - e^{-t/RC}).$$

This solution then shows an exponentially increasing function that flattens out for large values of t. In other words, we can deduce that $q(t = 0) = 0$ and $q(t = \infty) = C\mathcal{E}$, proportional to the total potential of the battery and capacitance of a charging capacitor. The goal and purpose for this experiment is through applying the process of a numerical method called **Euler's method** of approximation. This method gives us an approximation, and a plot that is generated with relative comparison.

Euler's Method of Approximation

Now there are several other known methods of numerical approximation to solve ODE's. Some perform at a faster calculation, or other perform with near certain accuracy in approximation to actual values. For this experiment, we will perform an approximation that is approachable in the list of other numerical approximations.

To begin, we first have the expression for an ODE with a given initial condition $y(x_0) = y_0$,

$$y'(x) = f(x, y) \quad \text{with} \quad y(x_0) = y_0 \tag{1}$$

Integrating any differential equation requires that we perform the approximation on an interval from x_0 to x_f, or $[x_0, x_f]$. Breaking this interval into separate points that are equally spaced, we have for n many spaced points, $x_0, x_1, ..., x_n$,

$$\Delta x = \frac{x_f - x_0}{n - 1}$$

Still working with n many spaced out points, we have $x_j = x_0 + j \cdot \Delta x$, which denotes specific values of x_n based on the position of the jth point. To be more clear, lets write out a few expressions to examine the layout,

$$\begin{aligned} y(x_0 = x_0 + 0 \cdot \Delta x) &= y_0 \\ y(x_1 = x_0 + 1 \cdot \Delta x) &= y_1 \\ &\vdots \\ y(x_n = x_0 + n \cdot \Delta x) &= y_n \end{aligned}$$

Following this, is the use of **the tangent line approximation** method we have learned in calculus. The expression for a tangent line approximation goes as

$$y(x) - y(x_0) = y'(x_0)(x - x_0),$$

which is what we would expect when solving for the *slope-intercept* form of a linear equation. The next consideration to make is from equation (1) above, which allows us to write that $y'(x_0) = f(x_0, y_0)$, recalling that $y(x_0) = y_0$. Hence, noting that $\Delta x = x - x_0$, we can now write the approximation for the first value as

$$y(x) \approx f(x_0, y_0) \cdot \Delta x + y_0.$$

Furthermore, this extends in an *recursive* and *iterative* fashion,

$$\begin{aligned} y(x_1) &\approx y'(x_0) \cdot (x_1 - x_0) + y_0 = f(x_0, y_0) \cdot \Delta x + y_0 \quad (\text{note that} \quad \Delta x = x_1 - x_0) \\ y(x_2) &\approx y'(x_1) \cdot (x_2 - x_1) + y_1 = f(x_1, y_1) \cdot \Delta x + y_1 \\ &\vdots \\ y(x_n) &\approx y'(x_{n-1}) \cdot (x_n - x_{n-1}) + y_{n-1} = f(x_{n-1}, y_{n-1}) \cdot \Delta x + y_{n-1} \end{aligned}$$

Lastly, for each value of x_n and $y(x_n)$, the values generate a plot that will approximate a solution to any given first order ODE. Our goal then is to produce such a plot given the differential equation for charge in a RC Circuit.

Instructions

1. Defining now a generalization of the recursive equation for Euler's method of approximation, we have the expression

$$\boxed{y_n(x_n) = f(x_{n-1}, y_{n-1}) \cdot \Delta x + y_{n-1}}$$

Our goal now is to build a Python script that will run these recursive expressions following by generating a plot. We will have to utilize Python's **for** loop to help us generate a valid solution. With that, lets plan out how we would approach this problem in a psuedo code fashion.

Step 1: Initialize values for x_0 and y_0, which will be our **initial conditions**.

Step 2: Define a function for Δx. Have it include an input for n number of x_n.

Step 3: Create a list of input values $x_0, x_1, ..., x_n$.

Step 4: Perform loop using Euler's method.

Step 5: Print output values then plot. Plot actual solution to compare.

Step 6: Perform error and any regression analysis, for fun.

Seems simple enough? OK lets begin.

2. The first thing we need to do is import the necessary modules.

```
******************************
import numpy as np
import matplotlib.pyplot as plt
******************************
```

3. Lets note that our ODE of interest is found on the first page,

$$q' + \frac{1}{RC}q - \frac{\mathcal{E}}{R} = 0$$

Our job now is to make an attempt at approximating the solution that is given by the above expression as $q(t) = C\mathcal{E}(1 - e^{-t/RC})$. We can start with some initial conditions and set our values for the battery and capacitor.

```
******************************
E = ?    # insert a value for the potential of a battery
C = ?    # insert a  value for the capacitance
R = ?    # insert a value for resistance
t0 = 0   # note that t=0 is t0
q0 = 0   # At t=0, the capacitor has zero charge
tf = 10  # we will assume final time is 10 seconds
n = 101  # here gives us 100 points from 0 to 10 seconds
# define our delta t function
def delta(t1,t2,n):
    return (t2 - t1) / (n - 1)
******************************
```

4. Now, we have to create an array of x_n values, and as usual we will apply numpy's np.linspace command. Doing so, we achieve

```
******************************
t = np.linspace(t0,tf,n) # set a value for t0,t1,...,tn
******************************
```

Following this, we need to give initial values for all values of charge q. Meaning that we need to start with setting every q-value to zero.

```
******************************
q = np.zeros([n]) # set every q-value to zero in an array format
******************************
```

5. This comes the fun part. Lets now create a loop rule that performs a recursive iteration of Euler's method to calculate the output values for each x_n. Lets start by initializing the intial value y_0.

```
******************************
q[0] = q0
******************************
```

Then lets create our loop.

```
******************************
for i in range (0,n):
    q[i] = delta(t1,t2,n) * (-1/(R*C) * q[i-1] + E/R) + q[i-1]
******************************
```

Notice that we have written the expression as

$$q(t_n) = q(t_{n-1}, q_{n-1}) \cdot \Delta t + q_{n-1}$$

where $q(t_{n-1}, q_{n-1}) = q'(t)$ and $q'(t) = -(1/RC)q + \mathcal{E}/R$. Double check the loop in our script to verify the expression above is placed in.

6. Lets then print out our values alongside printing the plot of what we have obtained. Notice that we have **not** generated a plot for the <u>exact</u> function.

```
******************************
# print out data values
for i in range(n):
    print(t[i],q[i])

# plotting the solution
plt.plot(t,q)
plt.xlabel("Time (s)")
plt.ylabel("Charge (C)")
plt.title("Approximation of Charge in RC Circuit")
plt.show()
******************************
```

7. We're not quite done yet. Lets then define a new function for the exact solution. It must look like in some form to what we have as a solution for charge of a capacitor. That is, it should be as of this form

$$q(t) = C\mathcal{E}(1 - e^{-t/RC})$$

Using the same t values we had above, generate a plot of both the exact solution and the approximation done with Euler's method on the same plot. Designate that one plot has a different color than the other. Be able to make use of Google to search matplotlib's function library. There it will explain how to have the plot show a different color bewtween the exact solution and approximation.

8. Be able to **summarize** the lab. What was the purpose and what did we do in order to complete the experiment? following this, answer the following prompts for your conclusion.

a. Once the lab is summarized for your conclusion, make sure that we have a full lab report template that includes question, hypothesis, and analysis.

b. How do the two plots compare? are they close or are they far apart?

c. What's the difference between solving an ODE versus an integral problem?

d. How would we perform error analysis?

BONUS: Find the average value of the error. What does this say?