NUMERICAL ANALYSIS OF ODES/PDES USING MATLAB

Mir-Tahmasebi, Mattin
mm5213

1.1 - RL Circuit

The following is the script HeunRL.m, which implements the Heun method for a function, Vin, passed to it:

```
function Vout = HeunRL(Vin, R, L, h, i0, tf )
    t=0;
    N=round((tf-t)/h);  % Number of steps is total time over step size
    i = i0;
    Vout(1,N) = zeros;
    Vout(1) = Vin(0);

    for j=1:N
        ip = i + h*(Vin(t) - R*i)/L;   % Predictor: predicted next value of i = current value + (step * slope at current point)
        iGradAvg = 0.5*(Vin(t) - (R*i) + Vin(t+h) - (R*ip)) / L; % Average slope between the points]
        i = i + h*iGradAvg;          % Corrector: actual next value of i = current value + (step * average slope)
        Vout(j+1) = Vin(t+h) - (R*i); % Output voltage, as per given equation
        t = t + h;             % Next t value
    end
end
```

Figure 1: HeunRL.m

The function implements the improved Euler method as so:

i. Find the number of steps we will need to take by dividing the total length of x-axis we will be working on (i.e. time) by the step size
ii. A better estimate for the gradient between the current and next values is the average of their gradients
iii. Use the average gradient to find a more correct next x value
iv. The voltage across the inductor is the input voltage minus the voltage across the resistor
v. Move on to the next x value

I tested this function with various different input signals. These are the results:

Input: Heaviside – h = 0.001, tf = 0.01

With a 5V Heaviside input to the RL circuit, the output voltage varies as shown in the graph:

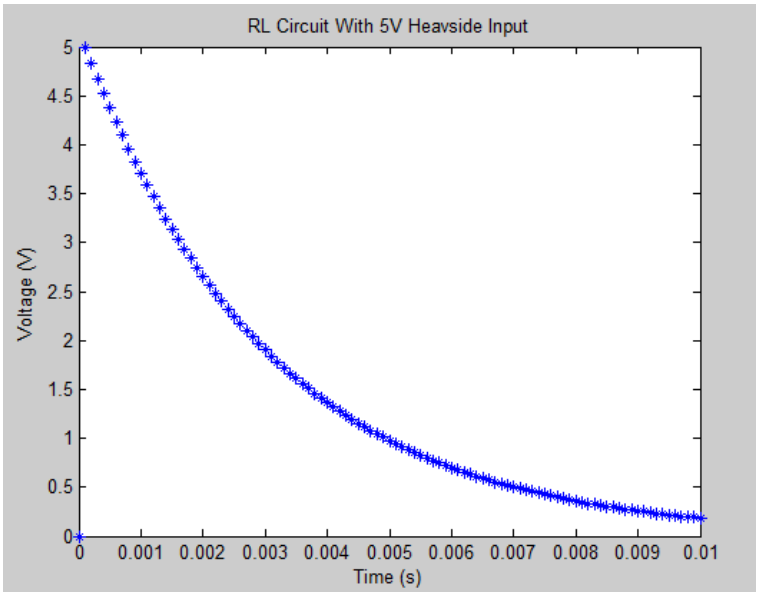

Figure 2: RL Circuit with 5V Heaviside Input

Our differential equation is:

$$L\frac{d}{dt}i(t) + Ri(t) = V_{in}(t)$$

Taking the Laplace transform gives:

$$LsI(s) + LI(0) + RI(s) = V_{in}(s)$$

Since i(0) = 0, this simplifies to:

$$LsI(s) + RI(s) = V_{in}(s)$$

$$I(s)(Ls + R) = V_{in}(s)$$

$$I(s) = \frac{V_{in}(s)}{(Ls + R)}$$

Also, we know that the voltage across an inductor is the product of the current and the reactance, and the transform of the reactance is *sL*:

$$V_L(s) = I(s)X(s) = I(s)sL$$

Substituting I(s) using the previous equation we derived gives:

$$V_L(s) = \frac{sLV_{in}(s)}{sL + R}$$

Our Vin is a Heaviside function, which has a transform of 1/s, so substituting this is gives us:

$$V_L(s) = \frac{1}{s} \cdot \frac{sL}{sL + R} = \frac{L}{sL + R}$$

Taking the inverse transform gives:

$$V_L(t) = e^{-t\frac{R}{L}}$$

Which explains the exponentially decaying shape of the curve we get.

Input: Exponential Decay – h = 0.001, tf = 0.02

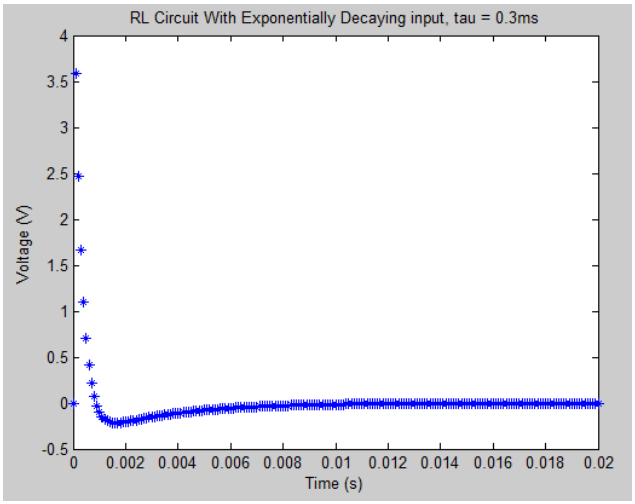

Figure 3: τ = 0.3ms

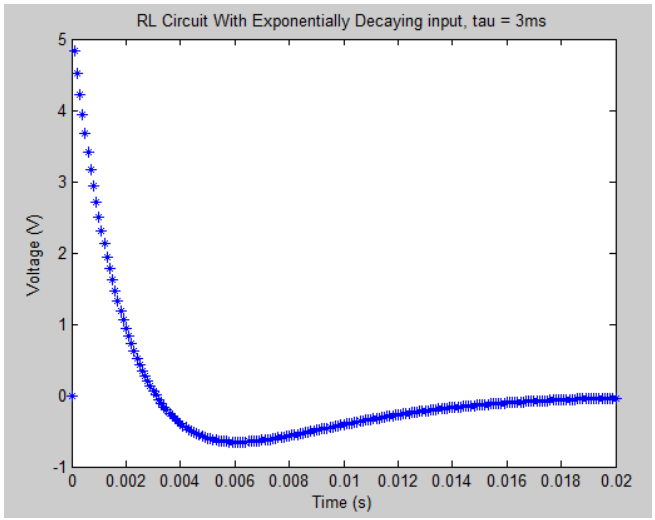

Figure 4: τ = 3ms

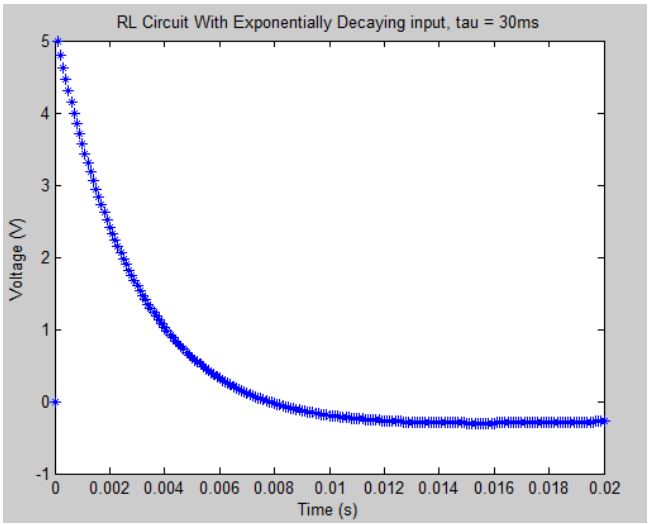

Figure 5: τ = 30ms

We can see that as τ gets smaller, the response tends towards a horizontal line at y = 0. This is because a small τ value causes the exponent to be very negative, so the exponential term becomes very small and significantly outweighs whatever constant factor it is being multiplied by (in this case 5). The zero input response of this circuit gives zero output, and that is what the graph portrays.

If τ is larger, the absolute value of the exponent becomes small, and the exponential term tends to 1, meaning it is insignificant in comparison to the constant factor, so our input essentially becomes equivalent to the Heaviside input shown earlier, which is why the graphs are similar.

The reason the graph dips below the 0V mark for a short time with exponential input is that the input voltage is decreasing, and a drop in voltage causes a negative change in current. Since the output voltage is proportional to the change in current, the output voltage becomes negative. However, as the exponentially decaying input starts to approach 0, the drop in voltage over time decreases and so the negative change in current becomes less significant, which makes our output graph level out.

Input: 5V Sine Wave

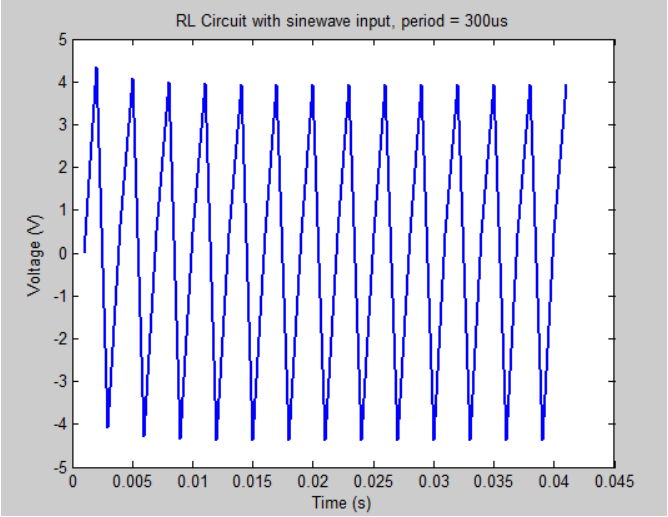

Figure 6: T = 300µs, h = 0.001, tf = 0.04

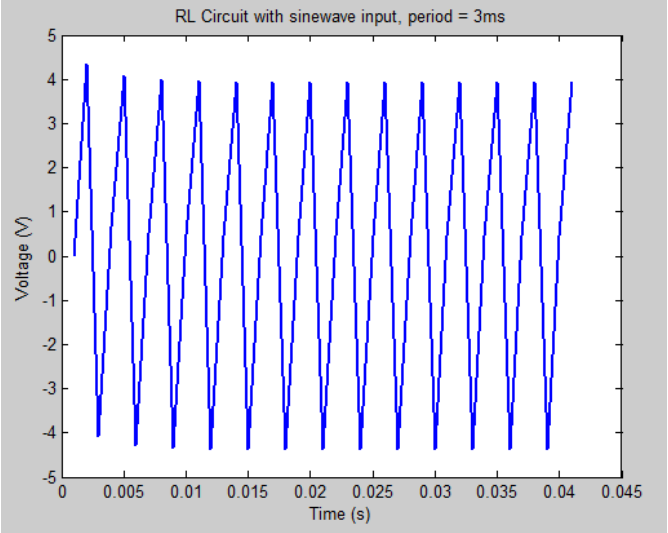

Figure 7: T = 3ms, h = 0.001, tf = 0.04

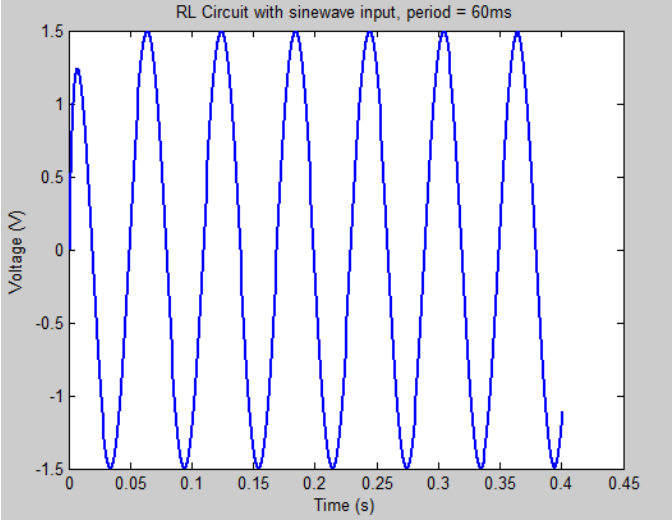

Figure 8: T = 60ms, h = 0.001, tf = 0.4

A sinusoidal input to our RL circuit produces a sinusoidal output, as the output is proportional to the rate of change of current, and oscillating change in voltage means that the current will also oscillate.

A low frequency input will have a lower amplitude output because the changes in voltage are less steep, so the response from the inductor will be smaller.

The exact solution can be calculated using the following:

$$\frac{d}{dt}i(t) + \frac{R}{L}i(t) = \frac{\bar{V}_{in}}{L}sin(\frac{2\pi}{T}t) \qquad a = \frac{R}{L}, \omega = \frac{2\pi}{T}$$

$$\frac{d}{dt}i(t) + ai(t) = \frac{\bar{V}_{in}}{L}sin(\omega t)$$

Now we solve this first order ODE:

$$P(t) = a,\ Q(t) = \frac{\bar{V}_{in}}{L}sin(\omega t)$$

$$\mu(t) = e^{\int a\ dt} = e^{at}$$

$$i(t)e^{at} = \frac{\bar{V}_{in}}{L}\int sin(\omega t)e^{at}\ dt = \frac{\bar{V}_{in}}{L}\left(\frac{e^{at}(asin(\omega t) - \omega cos(\omega t))}{a^2 + \omega^2} + c\right)$$

Since i(0) = 0:

$$0 = \frac{-\omega}{a^2 + \omega^2} + c$$

$$\therefore\ c = \frac{\omega}{a^2 + \omega^2}$$

$$\therefore\ i(t) = \frac{\bar{V}_{in}}{L}e^{-at} \cdot \frac{1}{a^2 + \omega^2}\left(e^{at}(asin(\omega t) - \omega cos(\omega t)) + \omega\right)$$

$$= \frac{\bar{V}_{in}}{L} \cdot \frac{1}{a^2 + \omega^2}\left(asin(\omega t) - \omega cos(\omega t) + \omega e^{-at}\right)$$

Substituting this into our equation for Vout finally gives us:

$$V_{out} = \bar{V}_{in}sin(\omega t) - \frac{a\bar{V}_{in}}{a^2 + \omega^2}(asin(\omega t) - \omega cos(\omega t) + \omega e^{-at})$$

Plotting our Heun approximation alongside the exact solution with a step size of *0.002* yielded Figure 9:

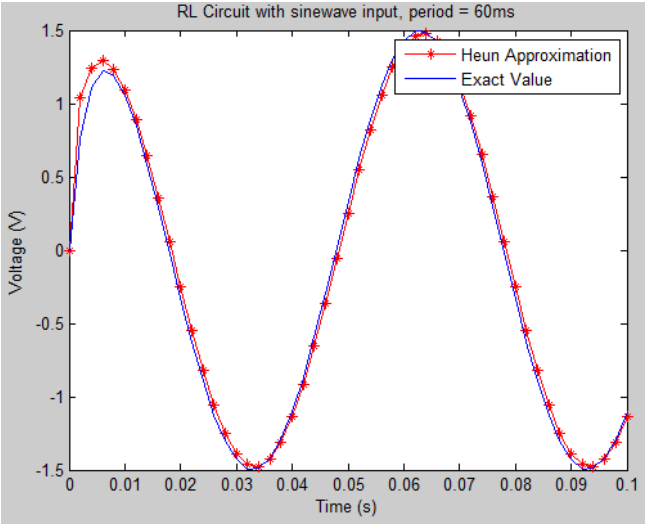

Figure 9: Step size 0.002

As you can see in Figure 10, there is a lot of error between the approximation and the exact solution. Decreasing the step size to *0.0002* yields a much better approximation:

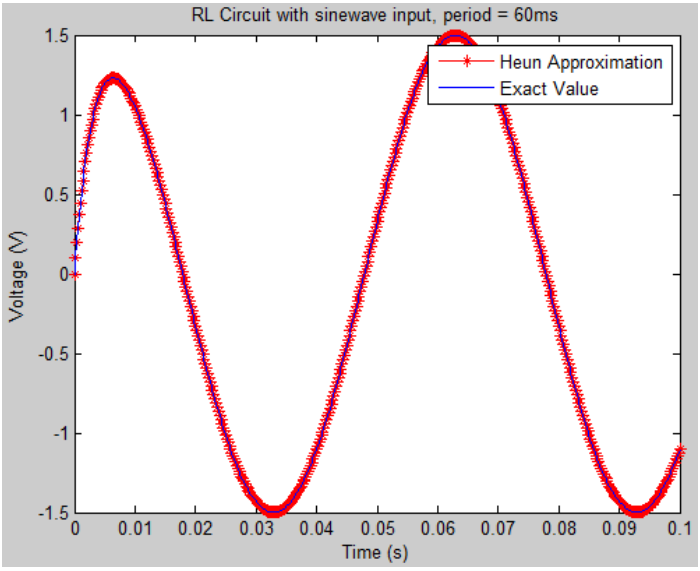

Figure 10: Step size 0.0002

To compare our Heun approximation with the exact solution, we use the two concepts of local truncation error and global truncation error. Local truncation error is calculated by taking the difference between two contiguous points on the approximated plot, and also on the exact plot, then finding the difference between those two values. We repeat this process for many values of h, and plot each of these on a graph.

Global truncation is found by calculating the sum of the differences between each contiguous point on the approximated plot, and finding the average of those. Then calculate the same value for the exact plot, and find the different between the two. Again, repeat this

for many values of h, and plot on a graph. The reason we do this is that different h values will produce data sets of differing lengths, so we can't compare them directly.

We plot the graphs with log scales so that we can find the order of error by calculating the gradient.

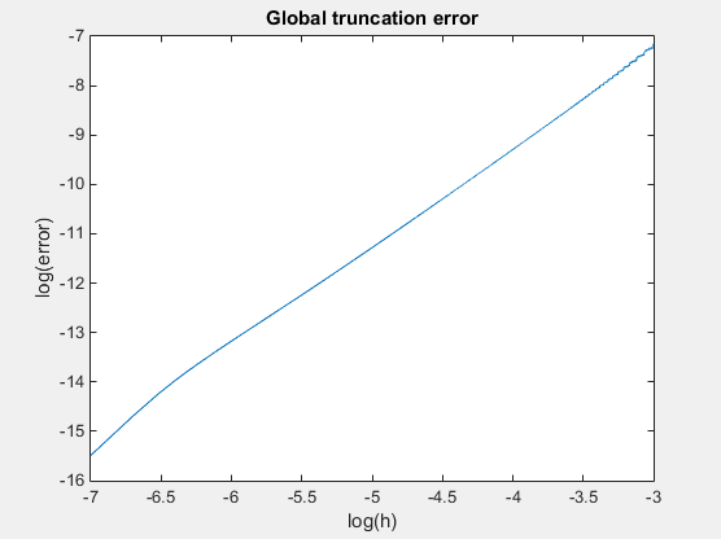

Figure 11: Global Truncation Error, $O(h^2)$

Calculating the gradient of the global truncation error gives us a gradient of about 2, so the global error is $O(h^2)$

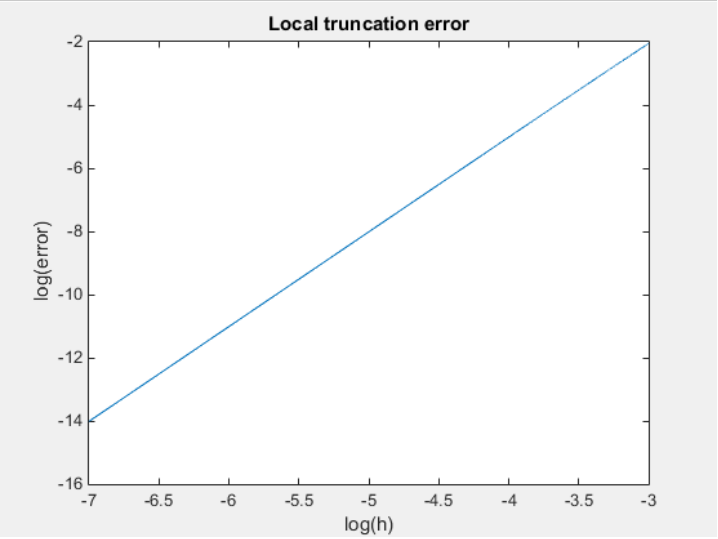

Figure 12: Local Truncation Error, $O(h^3)$

Calculating the gradient of the local truncation error gives us a gradient of about 3, so the local error is $O(h^3)$

These values match the algebraic solutions for the truncation errors, so our Heun method is correct.

Input: 5V Square Wave

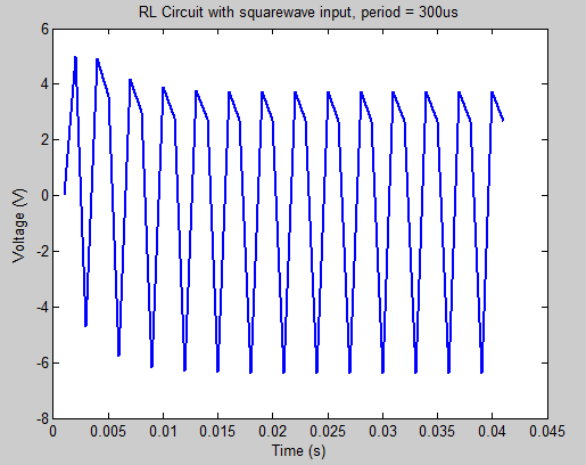

Figure 13: T = 300us, h = 0.001, tf = 0.04

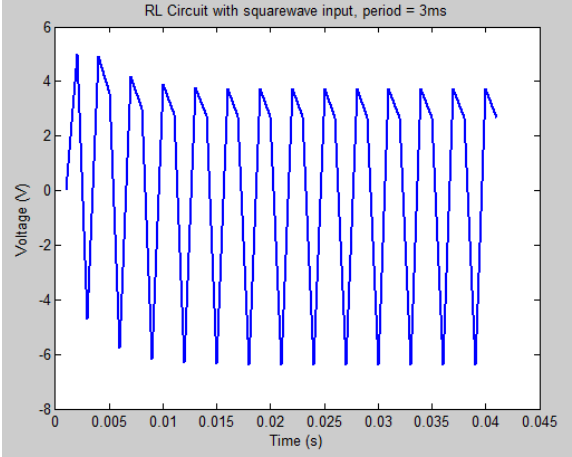

Figure 14: T = 3ms, h = 0.001, tf = 0.04

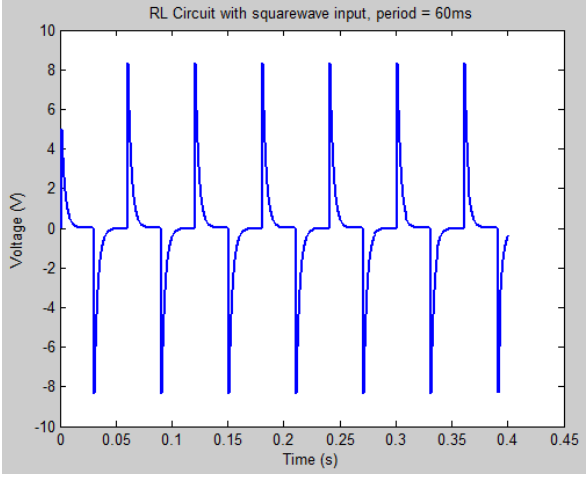

Figure 15: T = 60ms, h = 0.001, tf = 0.4

Looking at the T=60ms response, we see that the circuit treats the input as several Heaviside inputs, with opposing signs. When the frequency is higher, the system does not have time to reach a steady state by the time the next edge occurs, so the output oscillates a lot more.

1.2 - RLC Circuit

Script ruka_.m that defines the 4th-order Runge-Kutta

```
function [x, y] =ruka_(func1,func2,ti,xi,yi,h)

    k_x1=feval(func1,ti,xi,yi);
    k_y1=feval(func2,ti,xi,yi);

    k_x2=feval(func1,ti+(1/2)*h,xi+(1/2)*h*k_x1,yi+(1/2)*h*k_y1);
    k_y2=feval(func2,ti+(1/2)*h,xi+(1/2)*h*k_x1,yi+(1/2)*h*k_y1);

    k_x3=feval(func1,ti+(1/2)*h,xi+(1/2)*h*k_x2,yi+(1/2)*h*k_y2);
    k_y3=feval(func2,ti+(1/2)*h,xi+(1/2)*h*k_x2,yi+(1/2)*h*k_y2);

    k_x4=feval(func1,ti+h,xi+h*k_x3,yi+h*k_y3);
    k_y4=feval(func2,ti+h,xi+h*k_x3,yi+h*k_y3);

    phi_x=(1/6)*k_x1+(1/3)*k_x2+(1/3)*k_x3+(1/6)*k_x4;
    phi_y=(1/6)*k_y1+(1/3)*k_y2+(1/3)*k_y3+(1/6)*k_y4;

    x=xi+h*phi_x;
    y =yi+h*phi_y;
end
```

Figure 16: ruka_.m

This implements the RK equations given to us in the notes.

Script LCR.m that implements the 4th-order Runge-Kutta and the equation of the system:

```
function Vout = LCR(Vin, q, i, t, L, C, R, h, tf);

    N = round((tf-t)/h);
    Vout(1) = R*i;

    for j = 2:N+1
        igrad = @(t , xi, yi) 1/L*(-R*i - (1/C)*q + Vin(j*h));
        id = @(t, xi, yi) i;
        [q , i] = ruka_ (id, igrad, t, q, i, h);
        t = t + h;
        Vout(j) = R*i;
    end
end
```

Figure 17: LCR.m

In the following graphs, we will plot current instead of voltage, at this more accurately represents how the amplitude changes inversely proportionally to the damping factor.

Input: 5V Heaviside:

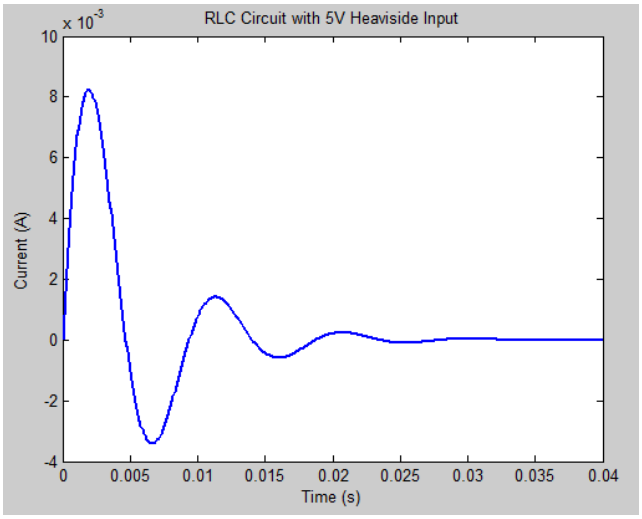

Figure 18: 5V Heaviside Input

Figure 18 shows the output with all the given input parameters. The signal fades with a damping factor given by:

$$\zeta = \frac{R}{2}\sqrt{\frac{C}{L}}$$

, which in the case of the given values is 0.3019. Changing R to be equal to

$$R = \frac{2}{\sqrt{\frac{C}{L}}}$$

causes zeta to become 1, and produces Figure 19 which demonstrates critical damping:

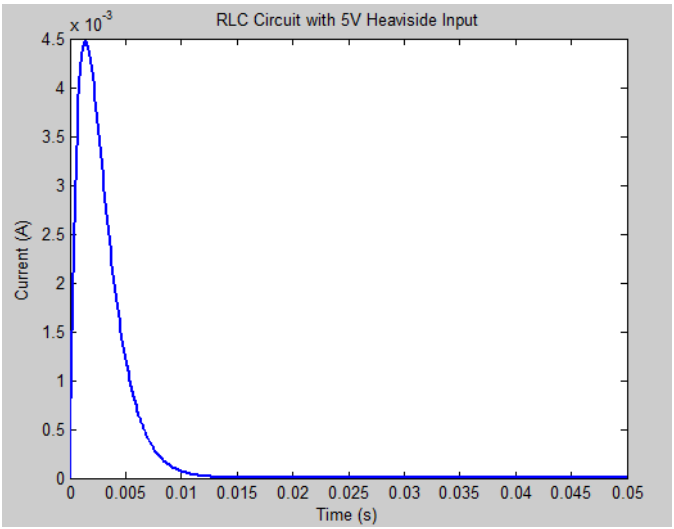

Figure 19: Critical Damping

Input: 5V Impulse:

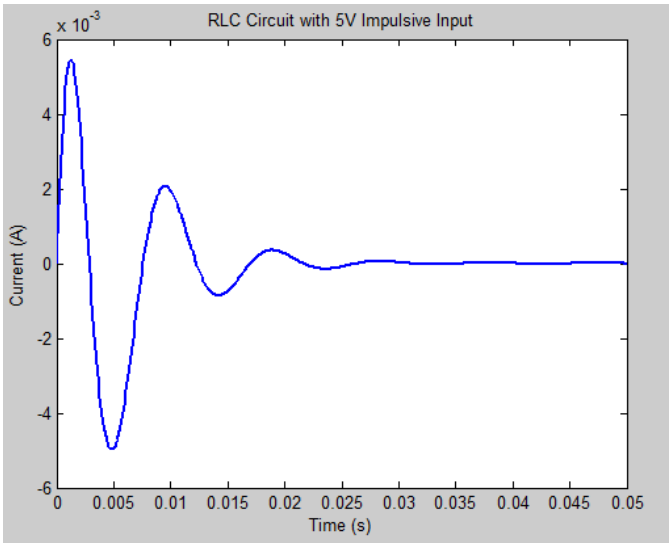

Figure 20: 5V Impulsive Input

The response for an impulsive input is very similar to that of a Heaviside input. The Heaviside response is caused by the sudden change in voltage, which we also get from an impulse, so for these purposes they are effectively equivalent.

Input: 5V Square Wave:

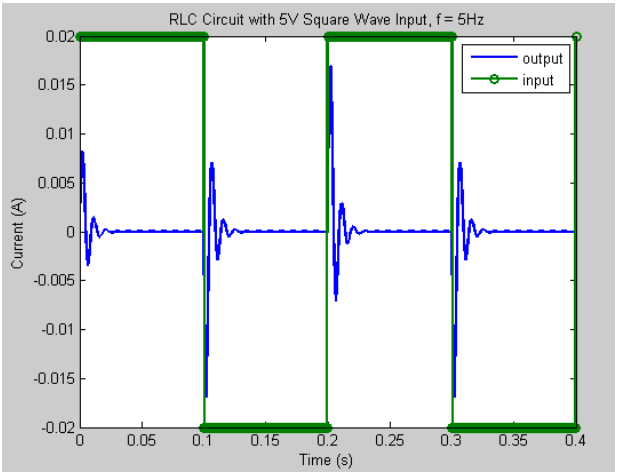

Figure 21: f = 5Hz

The square wave is equivalent to a repetition of the Heaviside input with alternating positive and negative voltage changes, so the response reflects that: for every positive edge there is a sharp positive spike in output which decays, and vice versa for a negative edge. (The amplitude of the input has been reduced from its actual value by a factor of R in Figure 21 for demonstration purposes, so the input and current output can be shown on the same scale but keep the same proportion as with the voltage output.)

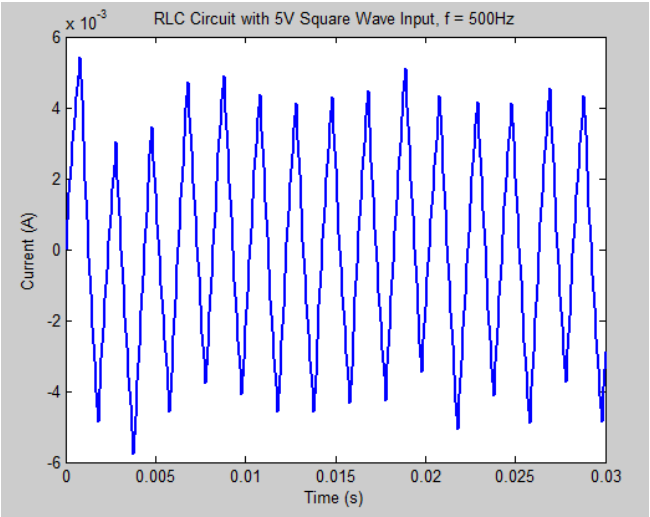

Figure 22: f = 500Hz

In Figure 22, the frequency is too high, and the output signal does not have a chance to reach a steady state before the next edge comes in, so the output oscillates as shown.

The resonant frequency can be found using the formula:

$$\omega_0 = \frac{1}{2\pi\sqrt{LC}}$$

In the case of our given parameters, this evaluates to around 109Hz. Using this frequency for our square wave gives us the following response:

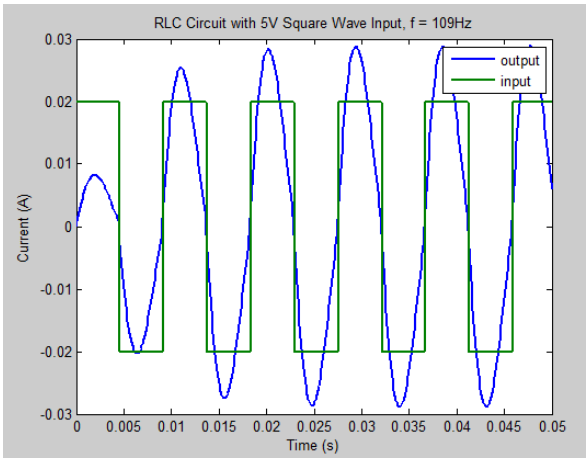

Figure 23: f = 109Hz

Since we are using the resonant frequency, resonance occurs in the system and the output is sinusoidal and amplified past the input amplitude. (Again, the amplitude of the input has been reduced in Figure 23)

Input: 5V Sine Wave:

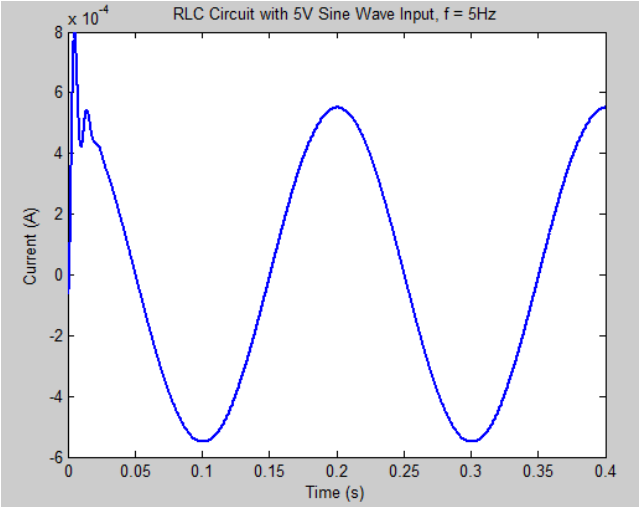

Figure 24: f= 5Hz

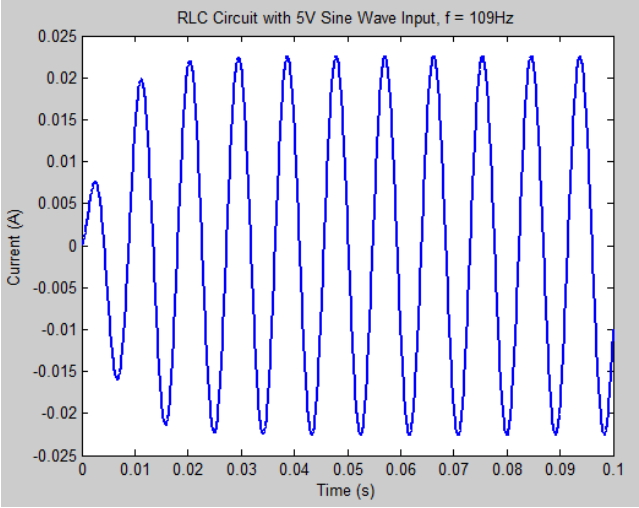

Figure 25: f = 109Hz

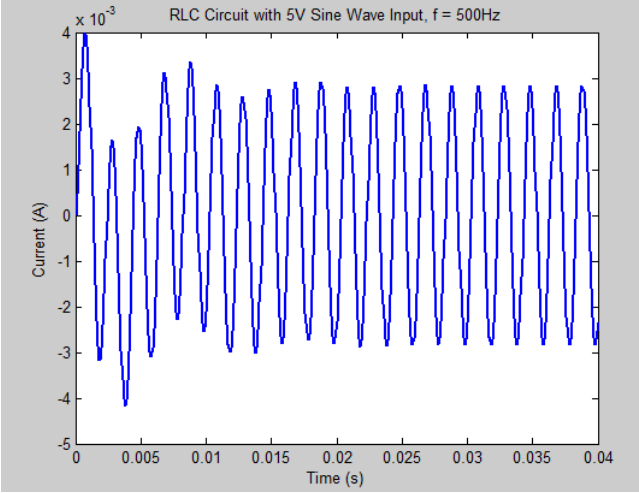

Figure 26: f = 500Hz

The output is proportional to the derivative of the charge, i.e. the second derivative of the current, and the second derivative of a sinusoid is still a sinusoid. So, since the input is a sinusoid, the output will be also.

Remember that zeta is around 0.3, meaning that the output is underdamped. This causes oscillations that you can see in the beginning of the 5Hz and 500Hz plots. The reason it is not as visible in the 109Hz plot is that 109Hz is the resonant frequency and the oscillations caused by underdamping are of the same frequency as the resonant frequency.

1.3 - Van der Pol Oscillator

A Van der Pol oscillator is governed by the differential equation:

$$\frac{d^2x}{dt^2} - \epsilon(1 - x^2)\frac{dx}{dt} + x = 0$$

The second term dictates the damping, so epsilon can be considered a damping factor. When modelling a Van der Pol oscillator, we plot two graphs – firstly a plot of our function, x, against time, t, and also a 'phase portrait' (also known as a 'limit cycle'), which is a plot of x against its derivative.

Input: Zero:
We begin with a zero input, for which the system still starts and settles into a stable oscillation. When epsilon is 0, we get Figure 27:

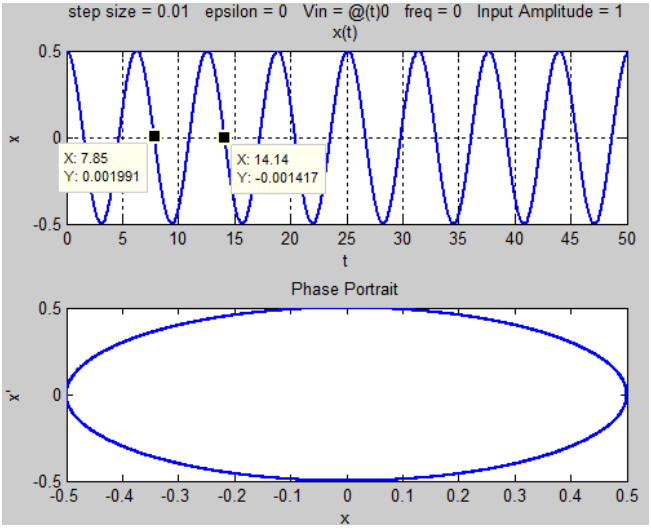

Figure 27: Zero Input Response, e = 0

We see that x(t) does not decay, which supports the fact that our damping factor is 0. Using the data points shown on the graph, we calculate the frequency of the zero-input response to be about 0.16Hz.

Increasing epsilon to 0.5 produces a plot with a damped x(t) and the spiral shaped phase portrait in Figure 28:

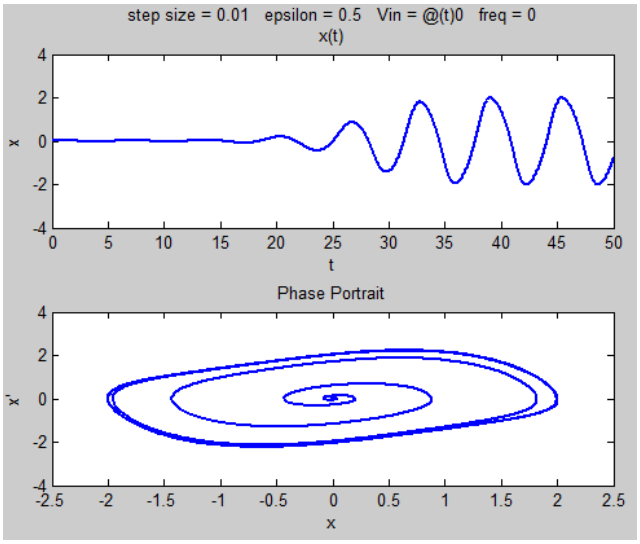

Figure 28: Zero Impulse Response, e = 0.5

The phase portrait is a plot of x against its derivative. For every point in x(t), its slope is calculated, which is a vector that is plotted on our portrait. For a signal of constant amplitude, the phase portrait is a circle, as the magnitude of the vectors stay constant, as shown in Figure 29, where the radius of the left hand circle represents the vector. When the signal suffers attenuation, the amplitude of x(t) varies, and so does the radius of the circle you plot, which produces the spiral-like portraits from signals with a non-zero epsilon.

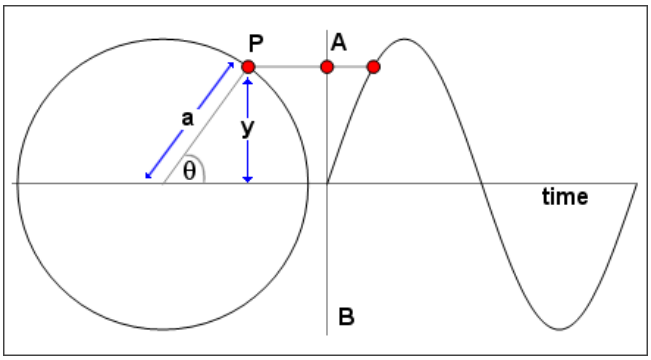

Figure 29: Phase portrait of sine wave

As epsilon grows larger, x(t) becomes distorted, but the amplitude is more consistent so the phase portrait is less spiralled in Figure 30:

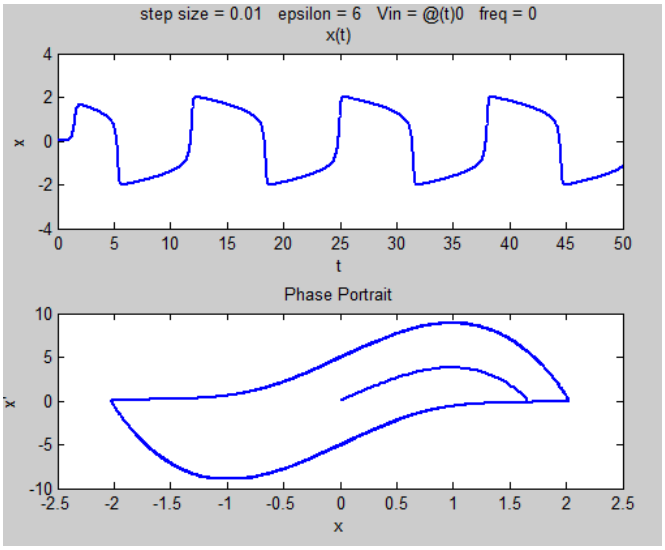

Figure 30: Increased epsilon to 6

The line starting from the centre of the phase portrait appears because the signal starts at x = 0, so the amplitude was lower for a short while.

Input: Sine wave:

Now we change the input to be a sine wave with 0.1Hz frequency and an amplitude of 1:

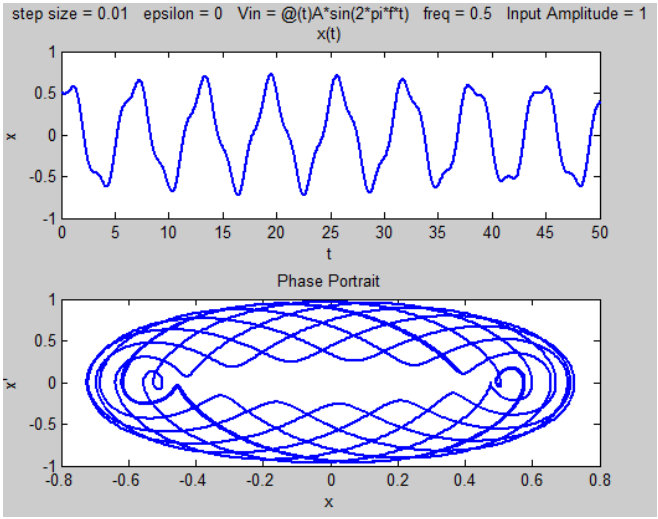

Figure 31: Sine wave input

The x(t) in Figure 31 displays chaotic behaviour, which you can see as it is unpredictable every period. The unpredictability is also visible from the wild pattern of the phase portrait.

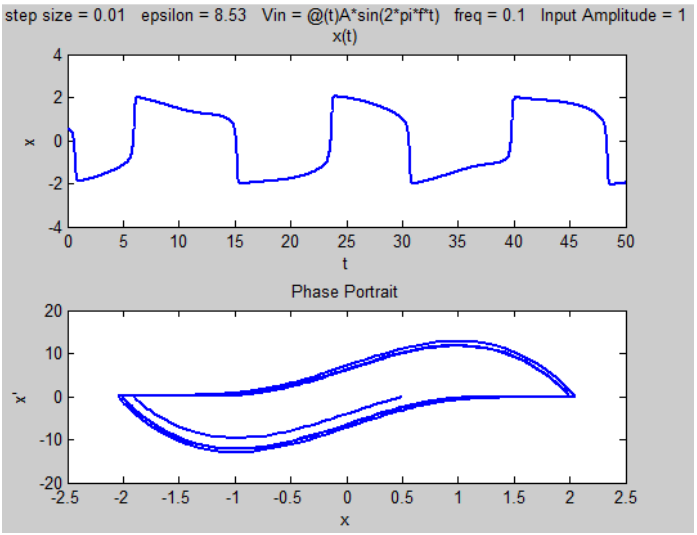

Figure 32: Increased epsilon to 8.53

From Figure 32 above, we find that at high epsilon values, we see relaxation oscillation. This is when the x value takes a long time to decrease to its minimum, but reaches its maximum very quickly.

Using a frequency of 0.5Hz instead of 0.1Hz in the input increases the frequency of the output:

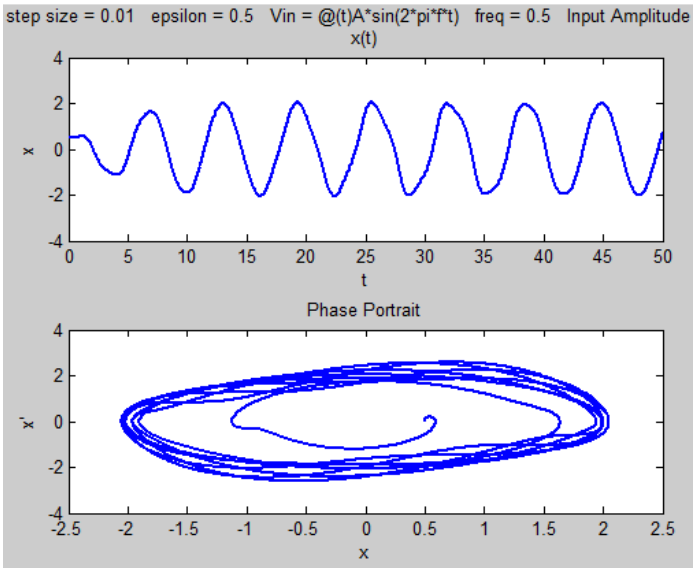

Figure 33: Increased freq to 0.5Hz

This is evident not only from the extra cycles in the x(t) plot in Figure 33 but also from the extra area covered by the phase portrait.

For a frequency of 0.5Hz, increasing the amplitude modulates the signal with another sine wave of frequency 0.16Hz, i.e. the frequency of the zero-input response. That is because the equation of the system with a forced input, F(t), is:

$$x = -x'' + \epsilon(1 - x^2)x' + F(t)$$

, so the input signal is just added to the zero input signal, as shown in Figure 34.

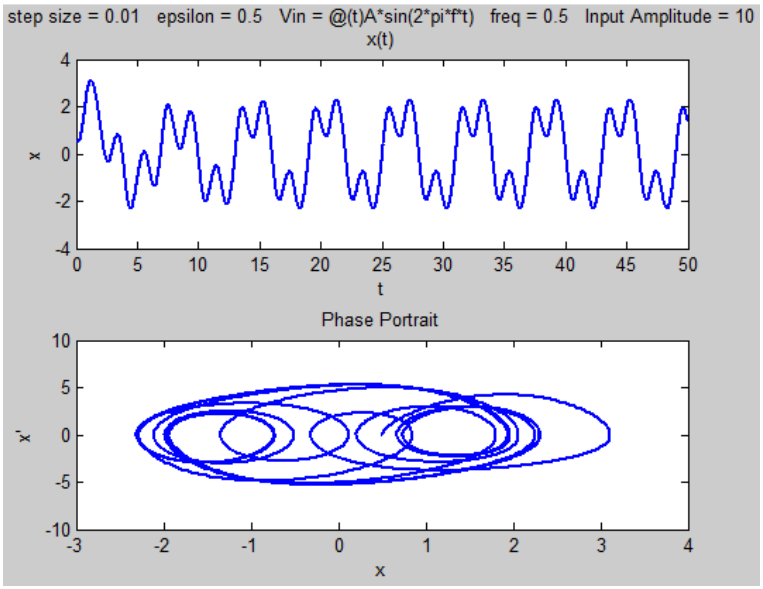

Figure 34: Increased Amplitude to 10

1.4 - Error Analysis

The second order central difference is given by:

$$\delta^2 y(x_i) = \underbrace{y(x_i+h)}_{(1)} \underbrace{-2y(x_i)}_{(2)} \underbrace{+y(x_i-h)}_{(3)}$$

The Taylor series expansions for the first and last terms are:

$$(1) : y(x_i+h) = y(x_i) + hy'(x_i) + \frac{h^2}{2!}y''(x_i) + \ldots$$

$$(3) : y(x_i-h) = y(x_i) - hy'(x_i) + \frac{h^2}{2!}y''(x_i) - \ldots$$

Summing (1) and (3) gives the following expression (the odd derivatives cancel out):

$$(1)+(3) = 2y(x_i) + 2\frac{h^2}{2!}y''(x_i) + 2\frac{h^4}{4!}y^{(4)} + \ldots$$

Now add (2) to complete the sum:

$$\begin{aligned}(1)+(2)+(3) &= 2\frac{h^2}{2!}y''(x_i) + 2\frac{h^4}{4!}y^{(4)}(x_i) + \ldots \\ &= h^2 y''(x_i) + 2\left[\frac{h^4}{4!}y^{(4)}(x_i) + \ldots\right]\end{aligned}$$

Since we just want the *second order* term, divide by h^2 to get:

$$\delta^2 y(x_i) = y''(x_i) + 2\left[\frac{h^2}{4!}y^{(4)}(x_i) + \ldots\right]$$

The second term has a factor of h^2, and this term is the error, so the error is of **$O(h^2)$.**