Ojas Bardiya
UID: 505145284
CEE/MAE 20
08/16/20

1. Mass-Spring Damper Set up

1.1. Introduction

We investigate the governing second differential equation of the mass-springer damper set up using the Forward Euler and Runge-Kutta method(s).
We use 8 different trials and plot the position (x) of the mass (m) over time (t) for both the homogeneous (Free vibration) and inhomogeneous (Forced Vibration) case.
We also analyze the frequency response and plot the normalized frequency vs absolute gain and also normalized frequency vs phase shift.
Finally, we generate 3 ten second animations of 30 fps (frames/second) with a time-step of dt = 1/300 using the Runge-Kutta 4 method each for the three possible cases –

- Under-damped ($\xi < 1$)
- Critically-Damped ($\xi = 1$)
- Over-Damped ($\xi > 1$)

We use the inbuilt value of π in MATLAB in all calculations.

1.2. Model and Methods

The data used is as follows –

Trial No.	mass (kg)	k(N/m)	c(N.s/m)
1	3	200	2
2	4	50	45
3	5	125	50
4	8	25	35
5	10	100	10
6	6	100	4
7	20	80	80
8	12	75	65

Note: Trials 6-8 are user-defined and are such that 6, 7, and 8 refer to the Underdamped, Critically Damped, and Over-Damped cases respectively.

We use the Forward Euler, Runge-Kutta 2 and Runge-Kutta 4 numeric schemes for analyzing this system.

Runge-Kutta 1 (Forward Euler)

$c_1 = \Delta t * f\,(t_k, y_k)$

$y_{k+1} = y_k + c_1$

Runge-Kutta 2

$c_1 = \Delta t * f\ (t_k, y_k)$

$c_2 = \Delta t * f\ (tk + ½*\Delta t, yk + ½* c_1)$

$y_{k+1} = y_k + c_2$

Runge-Kutta 4

$c_1 = \Delta t * f\ (t_k, y_k)$

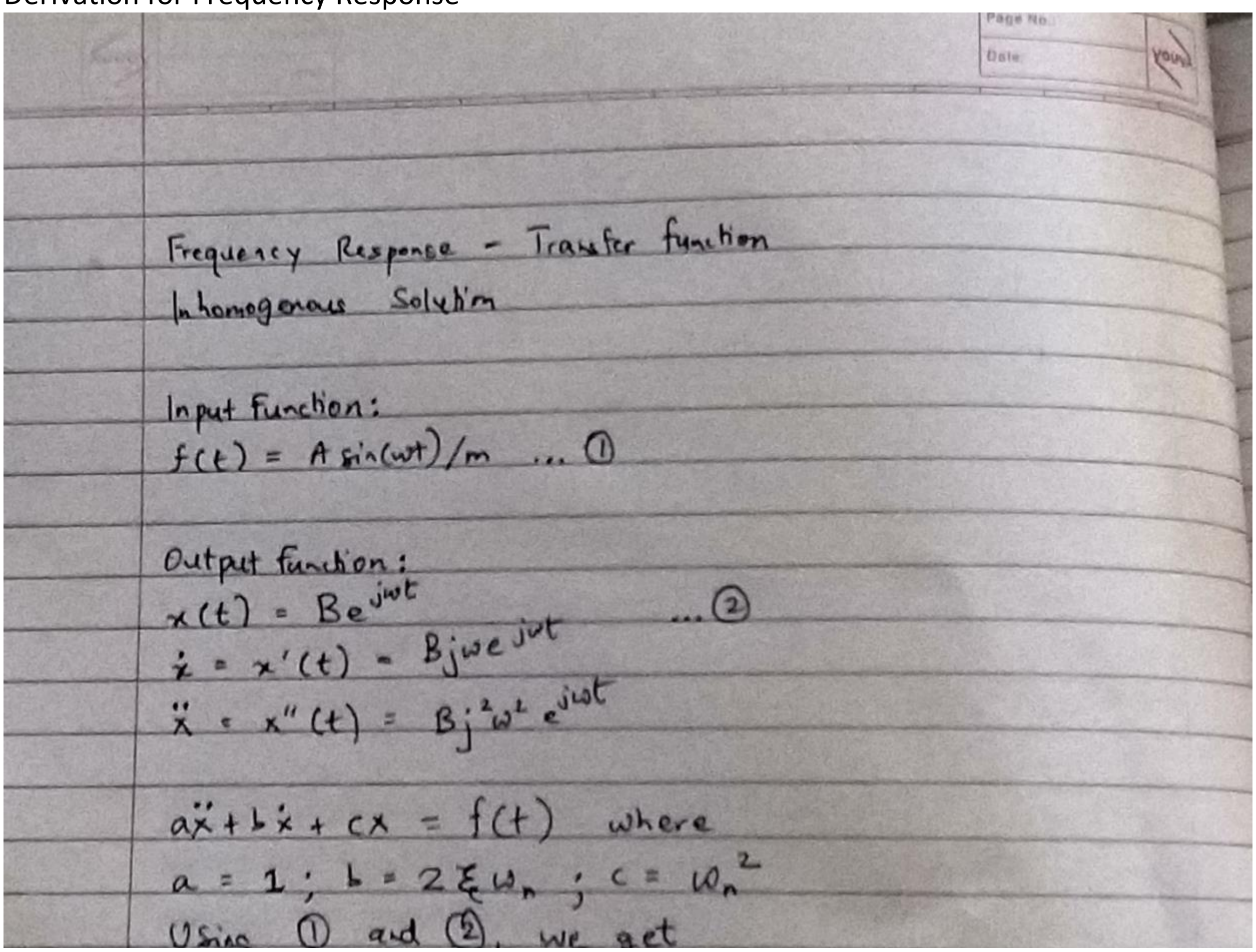
$c_2 = \Delta t * f\ (tk + ½*\Delta t, yk + ½* c_1)$

$c_3 = \Delta t * f\ (tk + ½*\Delta t, yk + ½* c_2)$

$c_4 = \Delta t * f\ (tk + ½*\Delta t, yk + c_3)$

$y_{k+1} = y_k + 1/6*c_1 + 1/3*c_2 + 1/3* c_3 + 1/6* c_4$

Derivation for Frequency Response –



Frequency Response - Transfer function

Inhomogenous Solution

Input Function:

$f(t) = A \sin(\omega t)/m \quad \ldots (1)$

Output function:

$x(t) = Be^{j\omega t} \quad \ldots (2)$

$\dot{x} = x'(t) = Bj\omega e^{j\omega t}$

$\ddot{x} = x''(t) = Bj^2\omega^2 e^{j\omega t}$

$a\ddot{x} + b\dot{x} + cx = f(t)$ where

$a = 1 \; ; \; b = 2\xi\omega_n \; ; \; c = \omega_n^2$

Using (1) and (2), we get

$$\frac{B}{A}\left(a\ddot{x} + L\dot{x} + cx\right) = \frac{\sin(\omega t)}{m}$$

$$\Rightarrow \frac{B e^{j\omega t}}{A}\left(-\omega^2 + 2\xi\omega_n(j\omega) + \omega_n^2\right) = \frac{\sin(\omega t)}{m}$$

$$\Rightarrow \frac{B}{A} = \frac{\sin(\omega t)}{e^{j\omega t}} \cdot \frac{1}{\left(\omega_n^2 + 2\xi\omega_n(j\omega) - \omega^2\right)m}$$

Since $\dfrac{\sin(\omega t)}{e^{j\omega t}} = k$, we get

$$\frac{B}{A} = \frac{k}{m} \cdot \frac{1}{(\omega_n^2 - \omega) + j(2\xi\omega_n\omega)}$$

Since $\omega_n = \sqrt{\dfrac{k}{m}}$, we conclude

$$\frac{B}{A} = G(j\omega) = \frac{\omega_n^2}{(\omega_n^2 - \omega) + j(2\xi\omega_n\omega)}$$

1.3. Pseudocode

<u>For VibrationPosition</u>

Set up initial values
Determine which numeric method to use
- *If Forward Euler, update the initial conditions according to the relevant equations*
- *If Runge-Kutta 2, update the initial conditions according to the relevant equations*
- *If Runge-Kutta 4, update the initial conditions according to the relevant equations*

End

<u>For main</u>

Set up values for all 8 trials
Set up time parameters
Calculate damping ratio and natural frequency

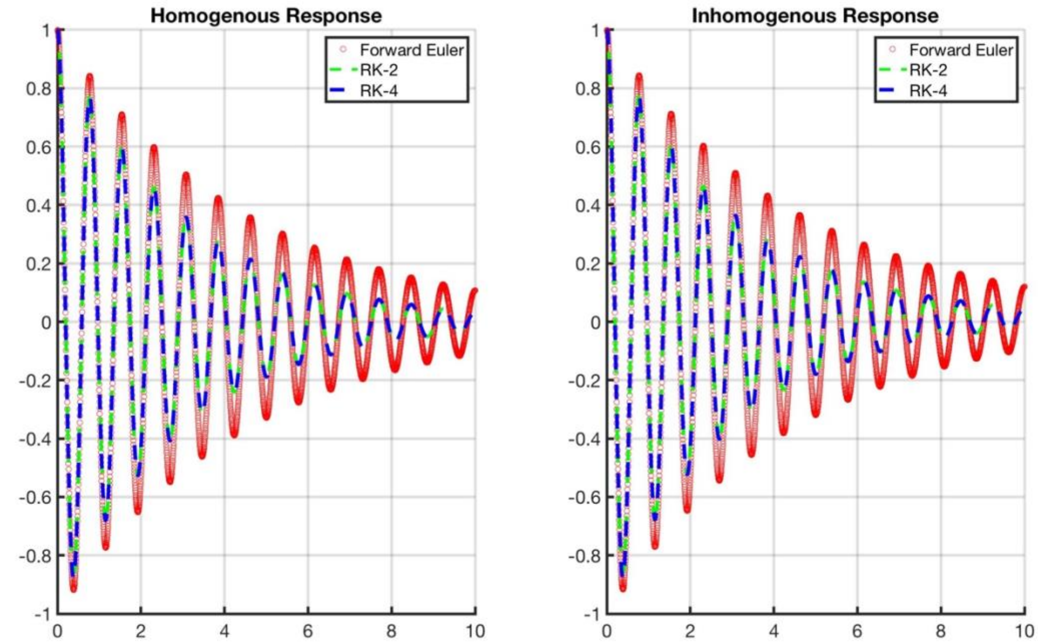
Set up initial position and force constant
Initialize arrays for all 3 numeric methods in the case of Free Vibration
Loop through each trial
  Find the position and velocity for the next iteration using the VibrationPosition function
  Update the values
Initialize arrays for all 3 numeric methods in the case of Force Vibration
Loop through each trial
  Find the position and velocity for the next iteration using the VibrationPosition function
  Update the values
Plot the relevant graphs
For frequency response, determine the normalized frequency
Initialize arrays for absolute Gain and phase shift
Loop through each trial
  Determine values for absolute Gain and phase shift
  Plot the relevant graphs
Animate for Underdamped, Critically damped and Overdamped cases as per the defined specifications

Note:

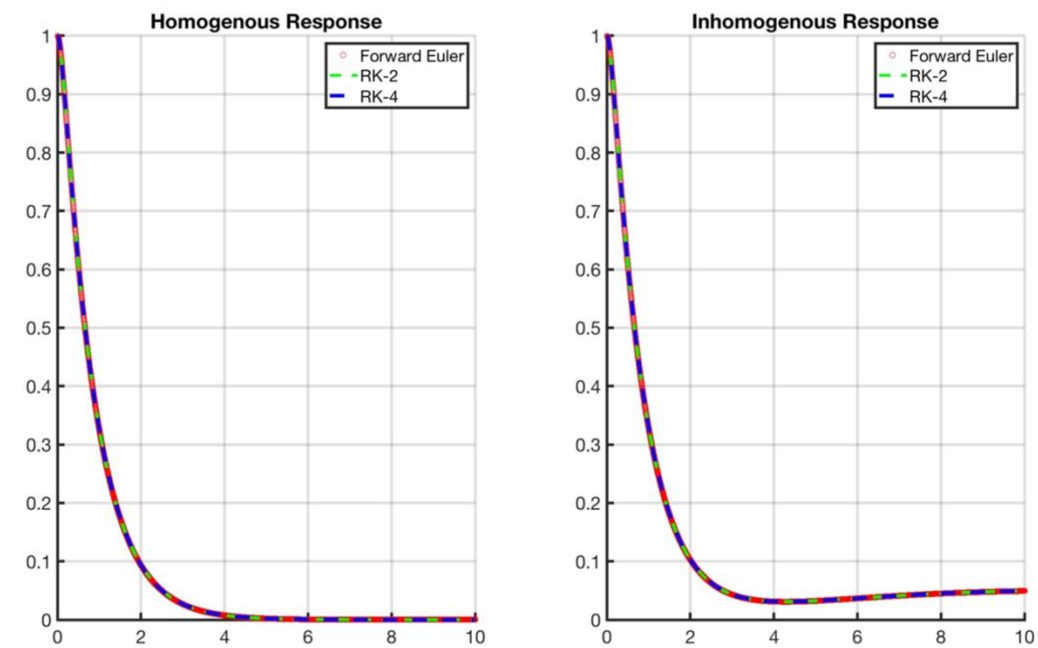
I have commented out the animation code for the Critically Damped and OverDamped cases, but have not done so for UnderDamped case in the MATLAB code I submitted.

1.4. Calculations and Results

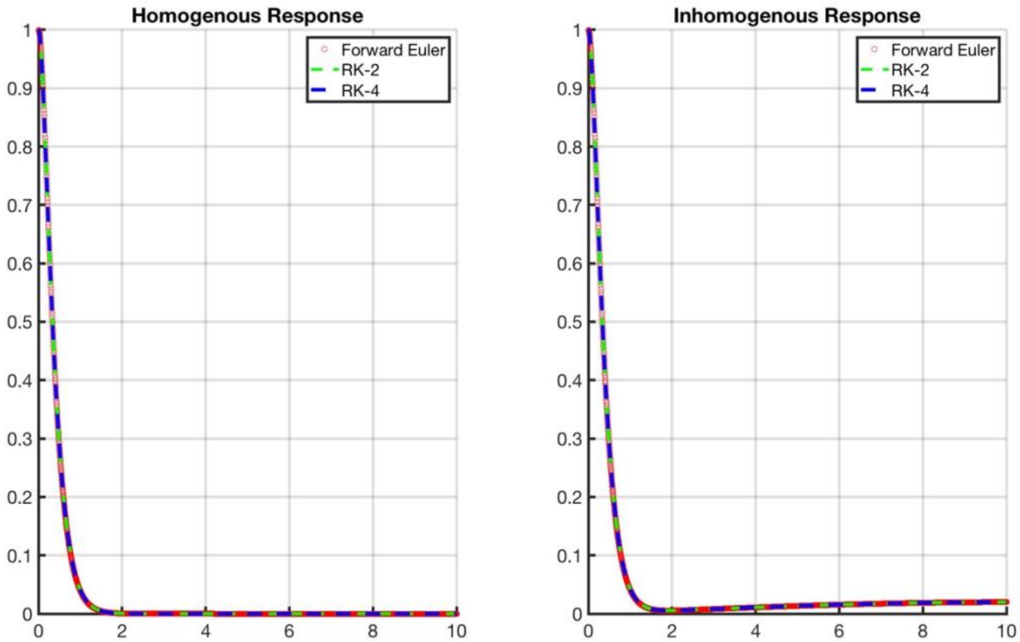
Plots –
For trial 1 –



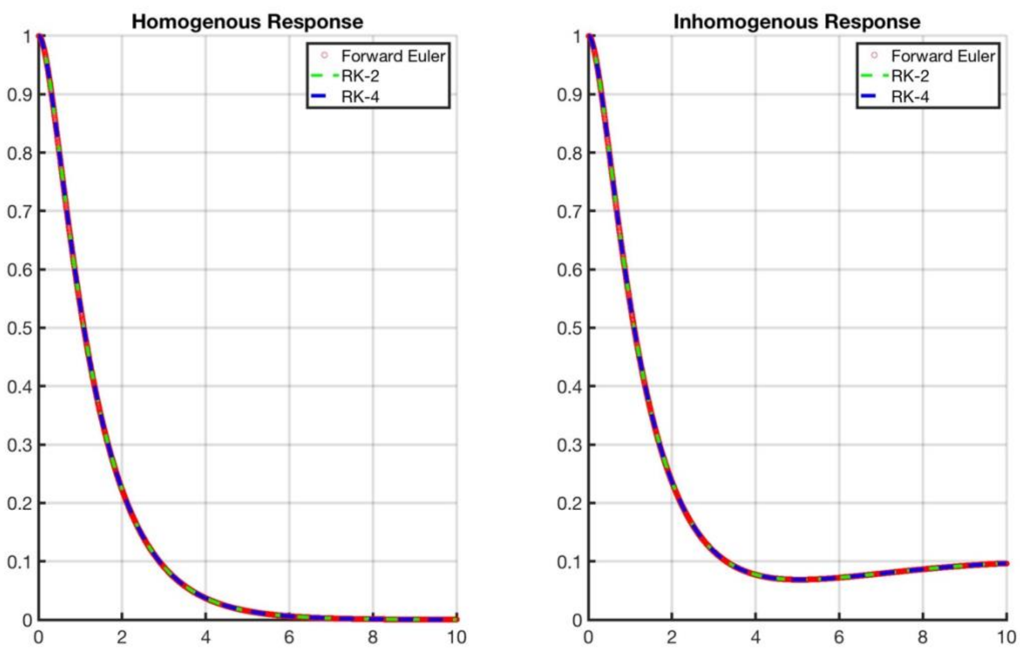
For trial 2 –



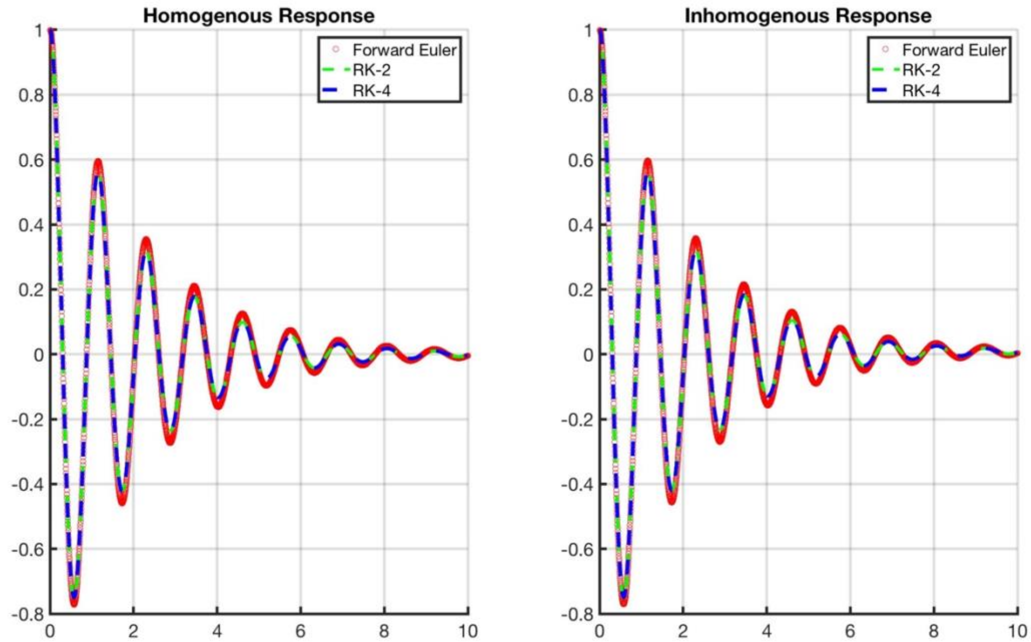
For trial 3 –



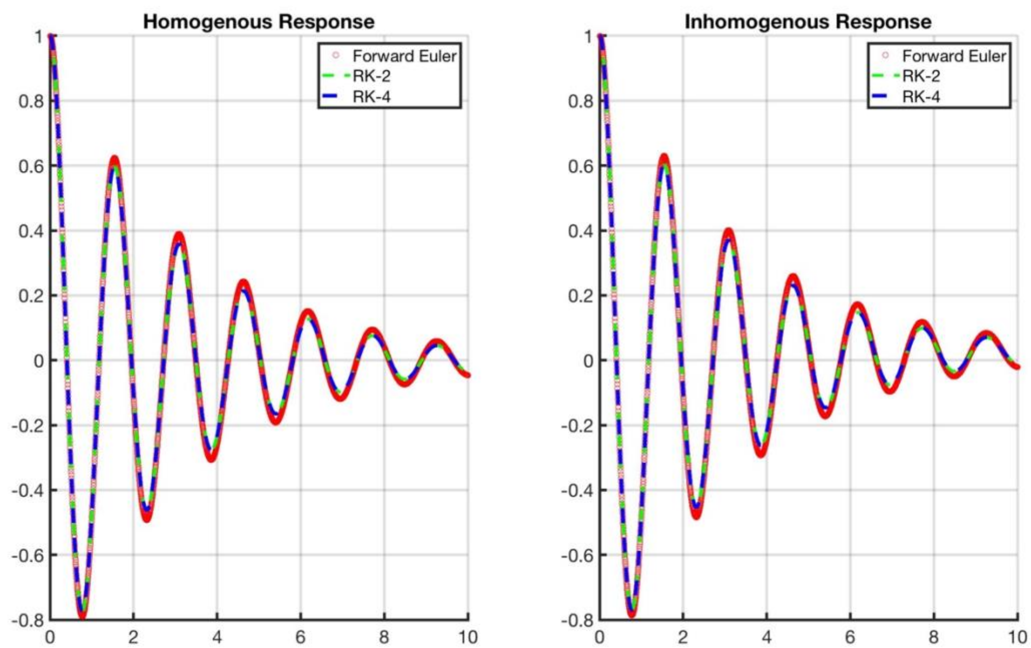
For trial 4 –



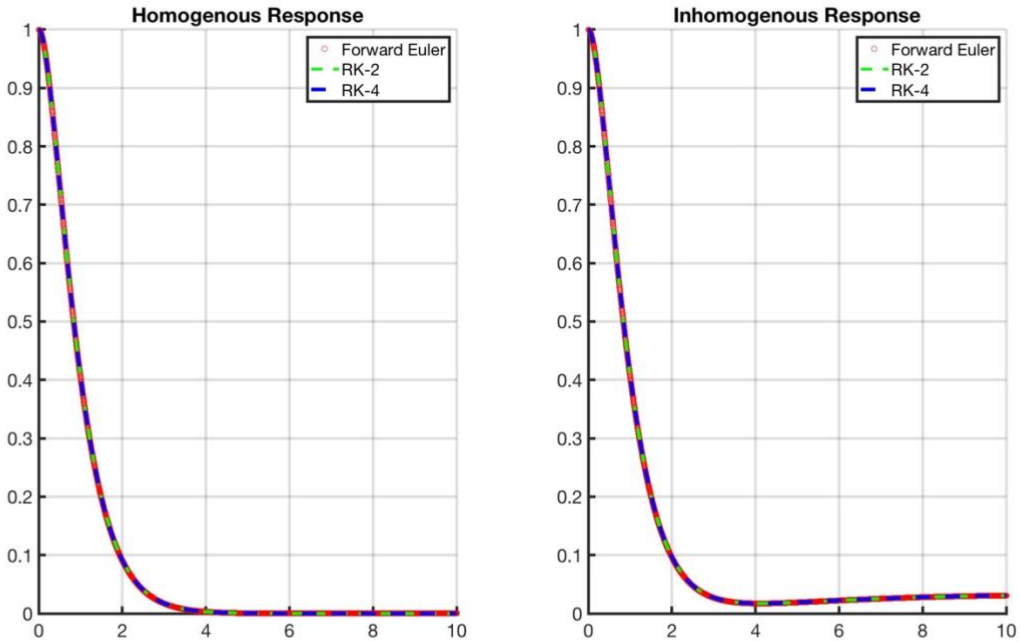
For trial 5 –



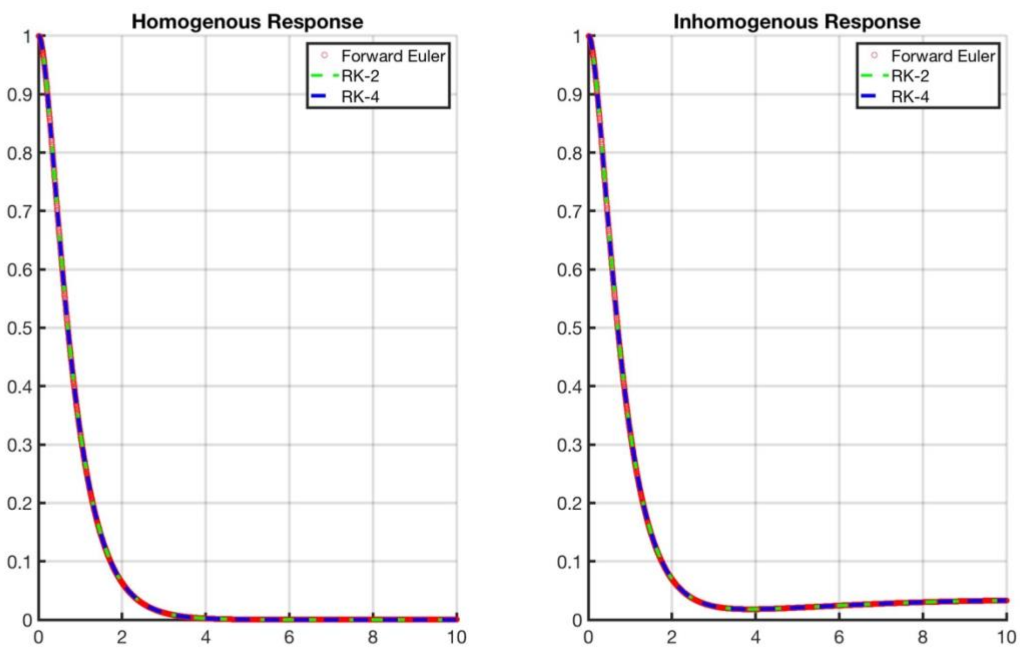
For trial 6 –



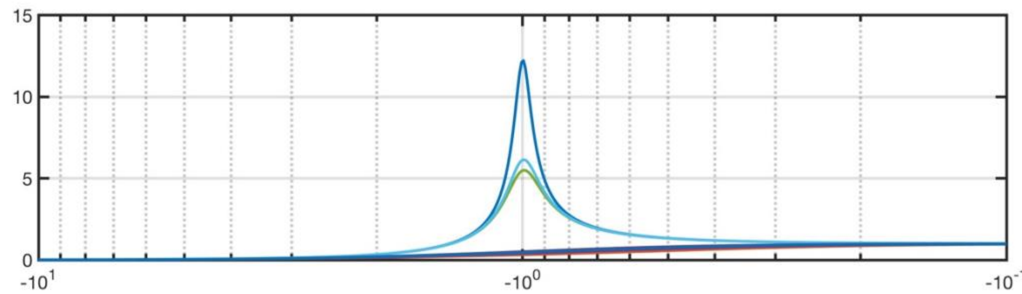
For trial 7 –

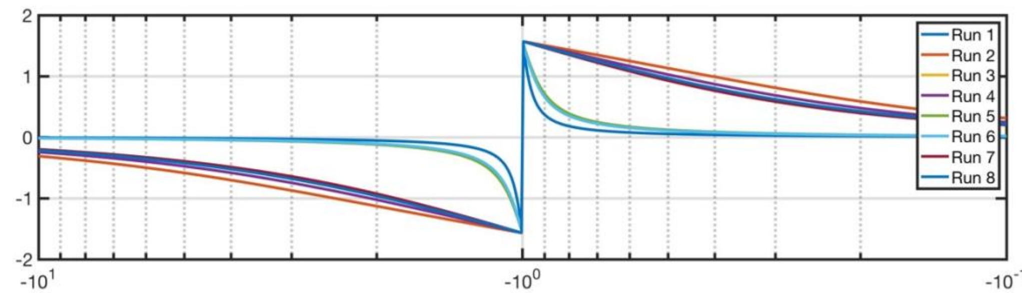


For trial 8 –



For frequency response –





We see that there is no appreciable difference between Runge-Kutta 2 and Runge-Kutta 4 for trials 1-8 but there is some degree of variation with the Forward Euler method for both of them.
For the frequency response, we the first plot shows normalized frequency vs absolute gain and the second shows normalized frequency vs phase shift.
For cases of Underdamping we observe that there is a greater amplitude factor, and resonance is observed more sharply than the Critically damped and Overdamped cases. The phase shift also tends to be lower in the Underdamped case vs Critically Damped and Overdamped until the normalized frequency approaches 0 form either side, where the 3 different cases tend to converge.

Conclusion:
We observe that the Underdamped case leads to a oscillating motion that gradually comes to a stop while the Critically damped and Overdamped cases return to a state of equilibrium faster, the former returning to such a state at the optimal rate.