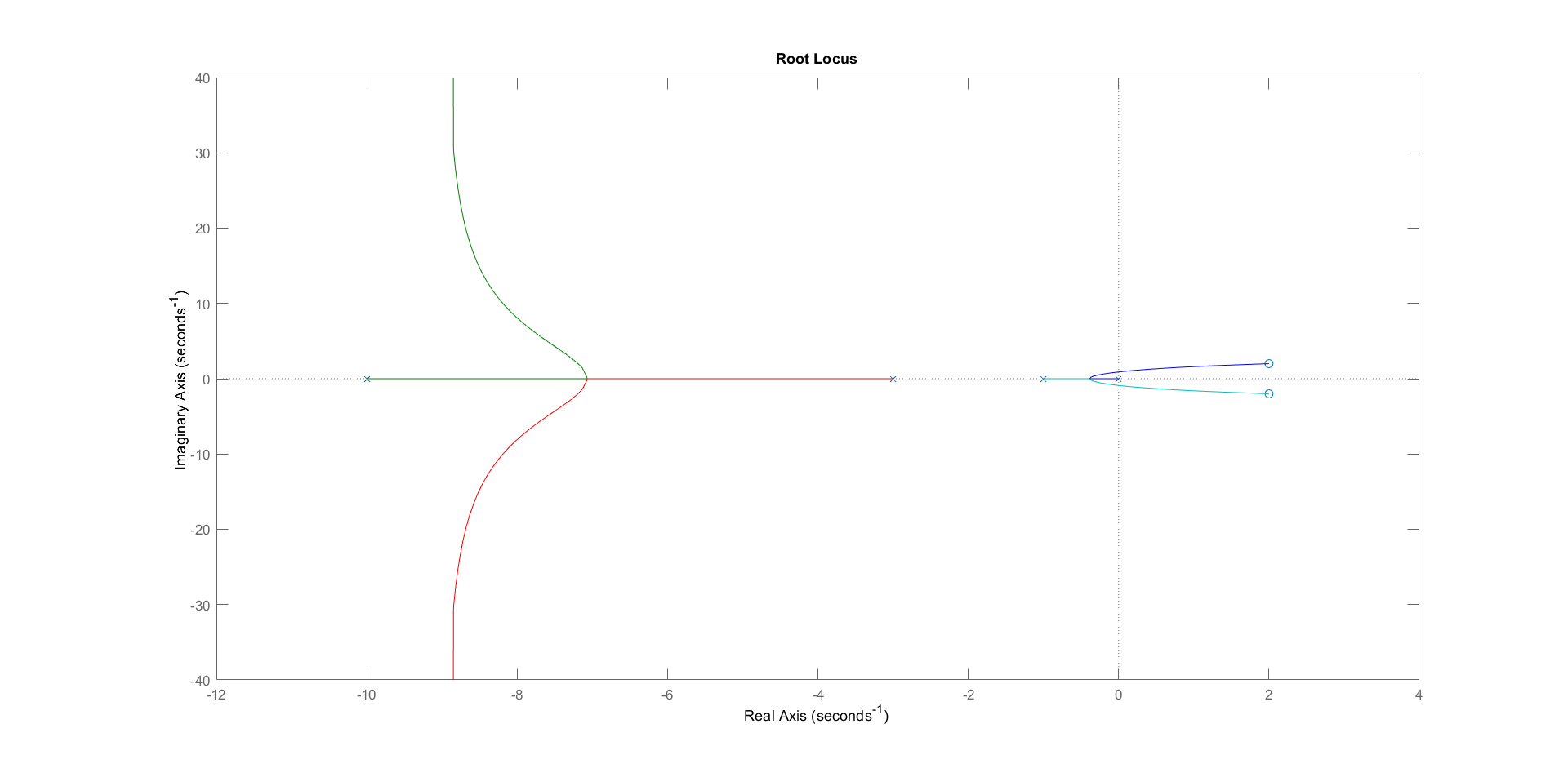
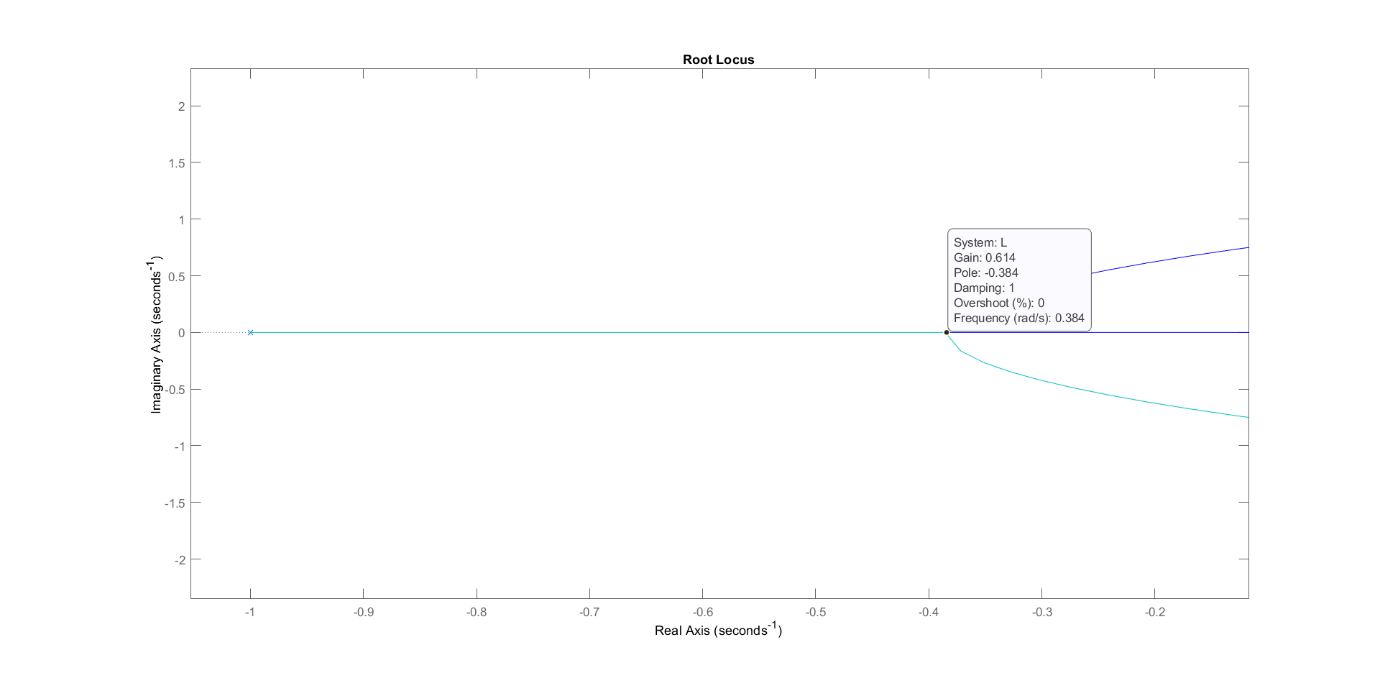
# Question 1)

## Question 1) b) RL plot

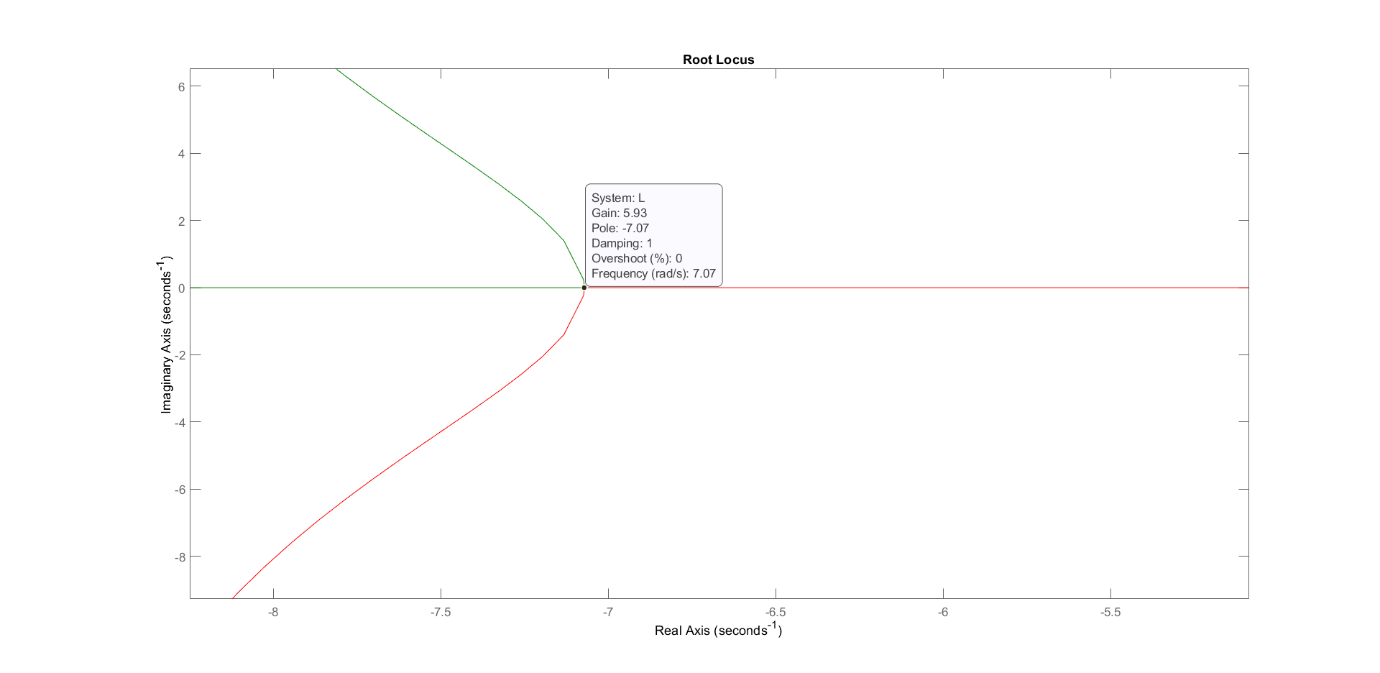


*Figure 1.1: Root Locus Plot generated using rlocusplot in MATLAB*

We can also see that the asymptotes meet at -9 (centroid) and at angles of 90 and 270 degrees respectively

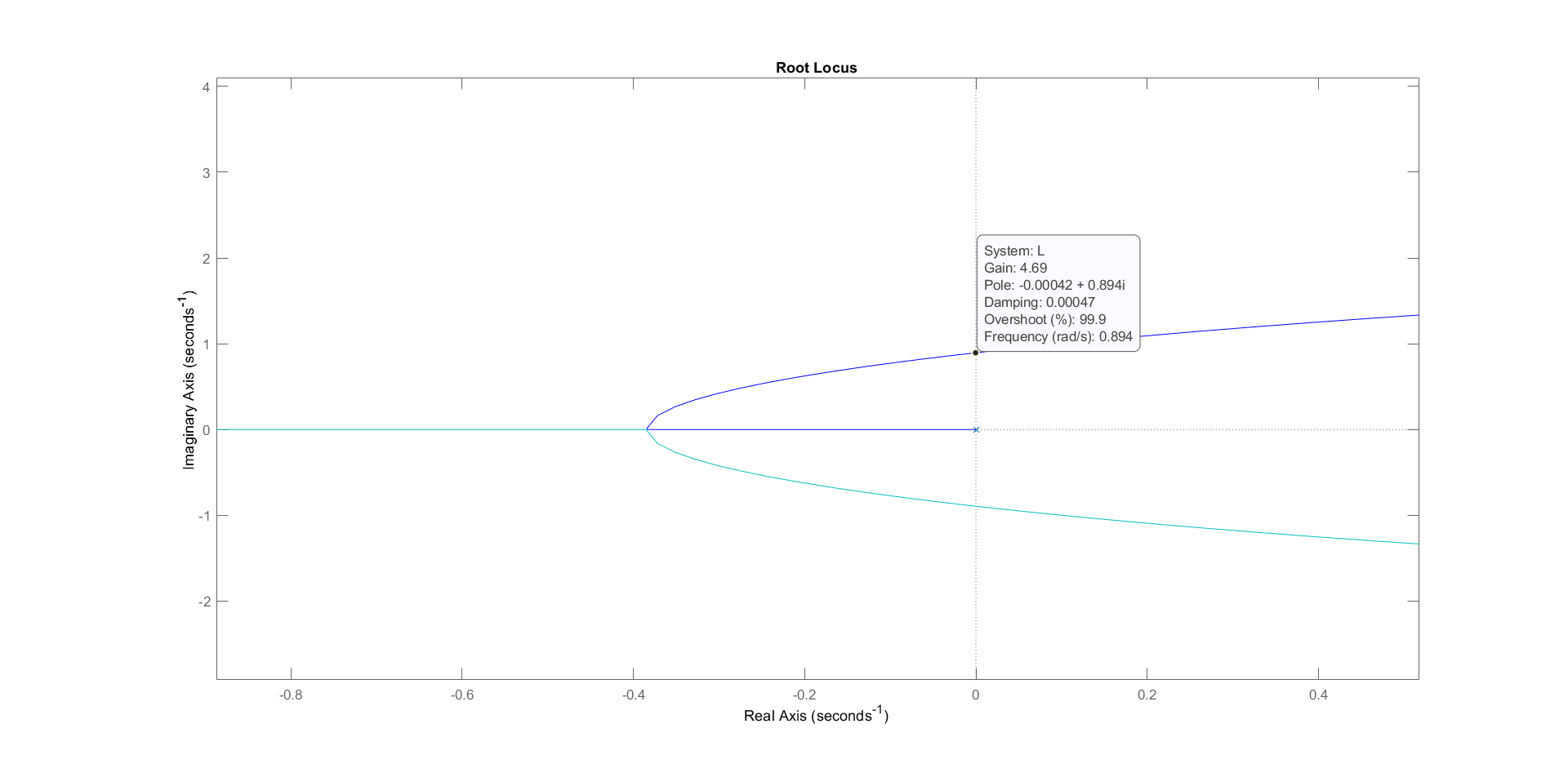


*Figure 1.2 a): Verifying break away point-1*



*Figure 1.2 b): Verifying break away point-2*

## Question 1) c) Ultimate Gain verification



*Figure 1.3: Finding the cross-over point.*

From the above plot we can see that at approximately the cross-over point, the gain is approximately 4.69. This is very close to the analytically computed value of 4.696 hence verifying it.

## Question 1) d) Minimising settling time

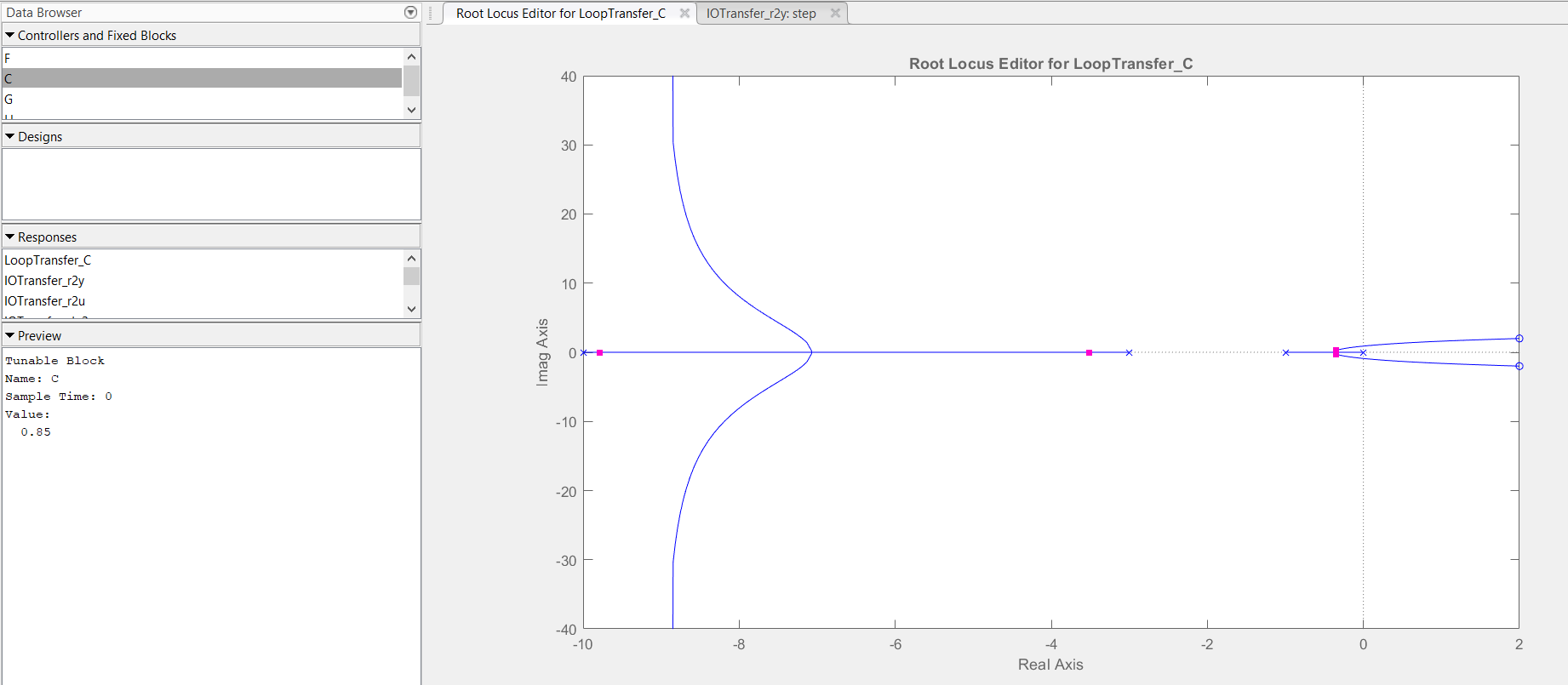
We know that KC > 0, and KC < KC,U = 4.696.

So the KC value which gives the minimum settling time must be between these 2 values. Employing the function ‘**stepinfo**’ which gives the settling time (apart from an assortment of other information about the step response), we evaluate settling time for all KC values between 0 and 4.69 (with 2 decimal place accuracy) and find the KC which gives the minimum value.

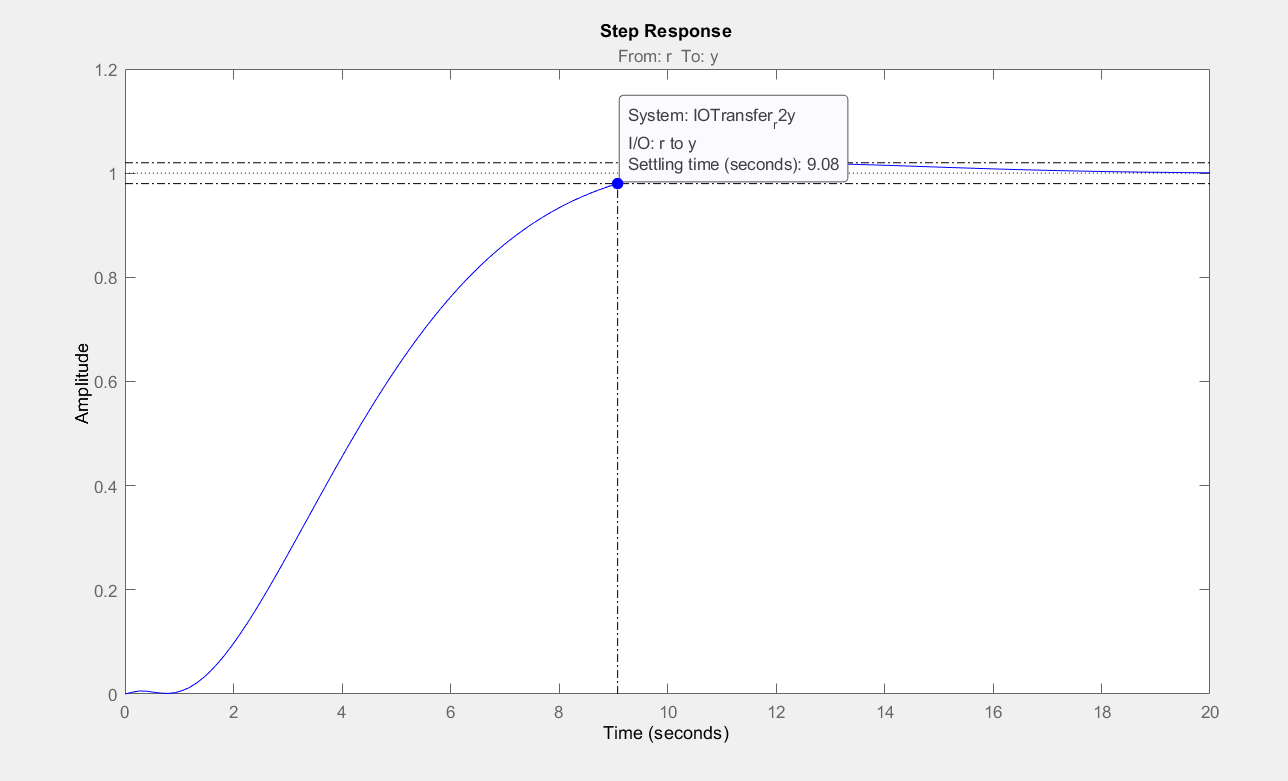
The required **KC =** **0.85** and it has a settling time of 9.0798 units.

This is further verified using rltool. We need to get the dominant pole away from origin as far as possible to quicken the dynamics of the system. (And that the same time all poles should be in LHP)

This is achieved in this configuration:



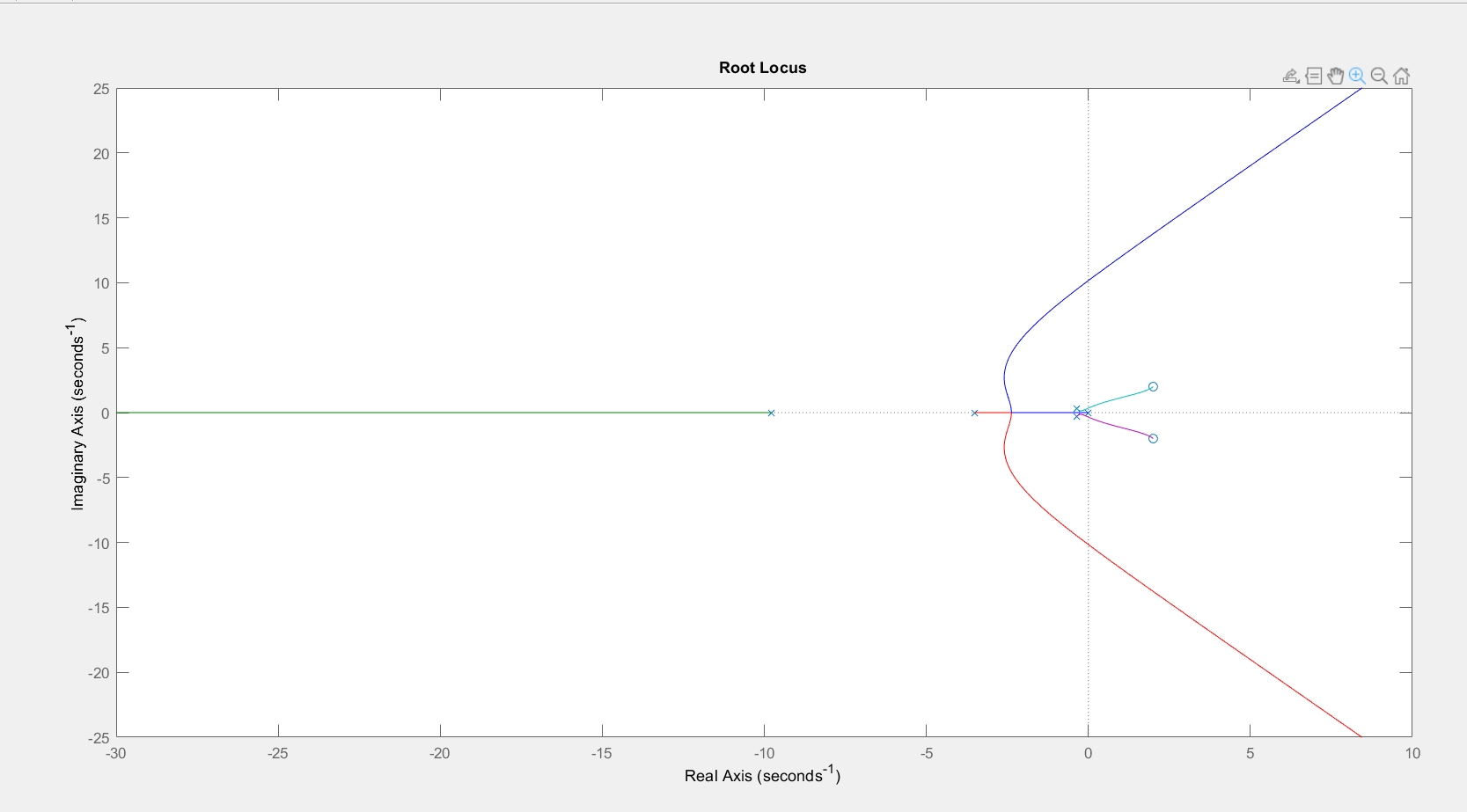
*Figure 1.4: RL plot using rltool at minimum settling time*

**

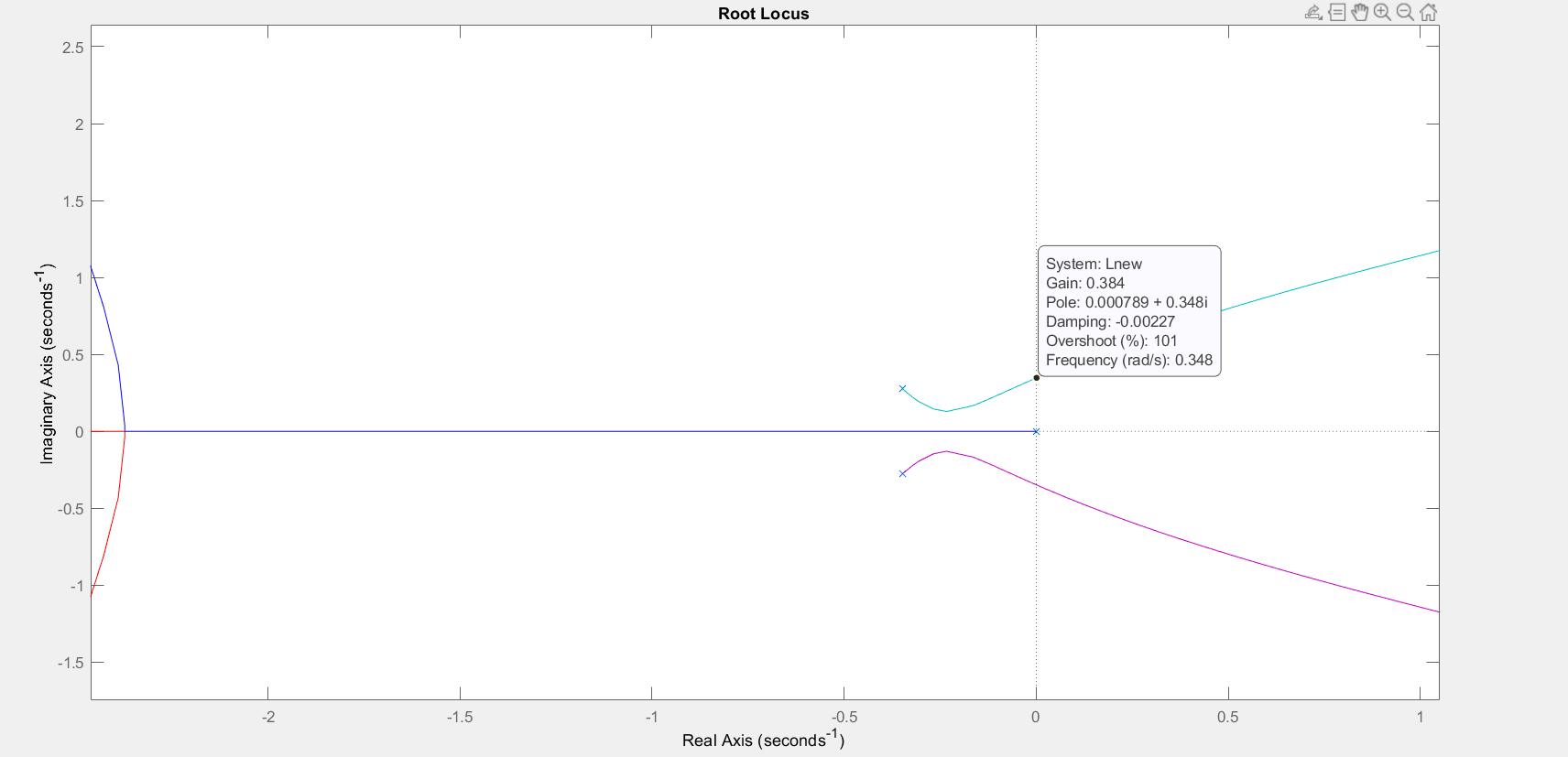
*Figure 1.5: Settling time for the optimal Kc.*

## Question 1) e) Ultimate gain for KI

Given that the equation for solving s = jw is complicated, it is much easier to use the root-locus plot generated by MATLAB.



*Figure 1.6 a): The required root locus plot*



*Figure 1.6 b): Finding the cross-over point*

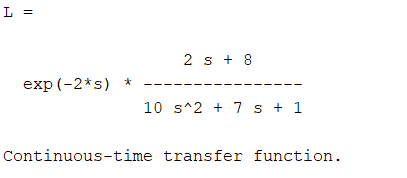
KI,ultimate = 0.384.

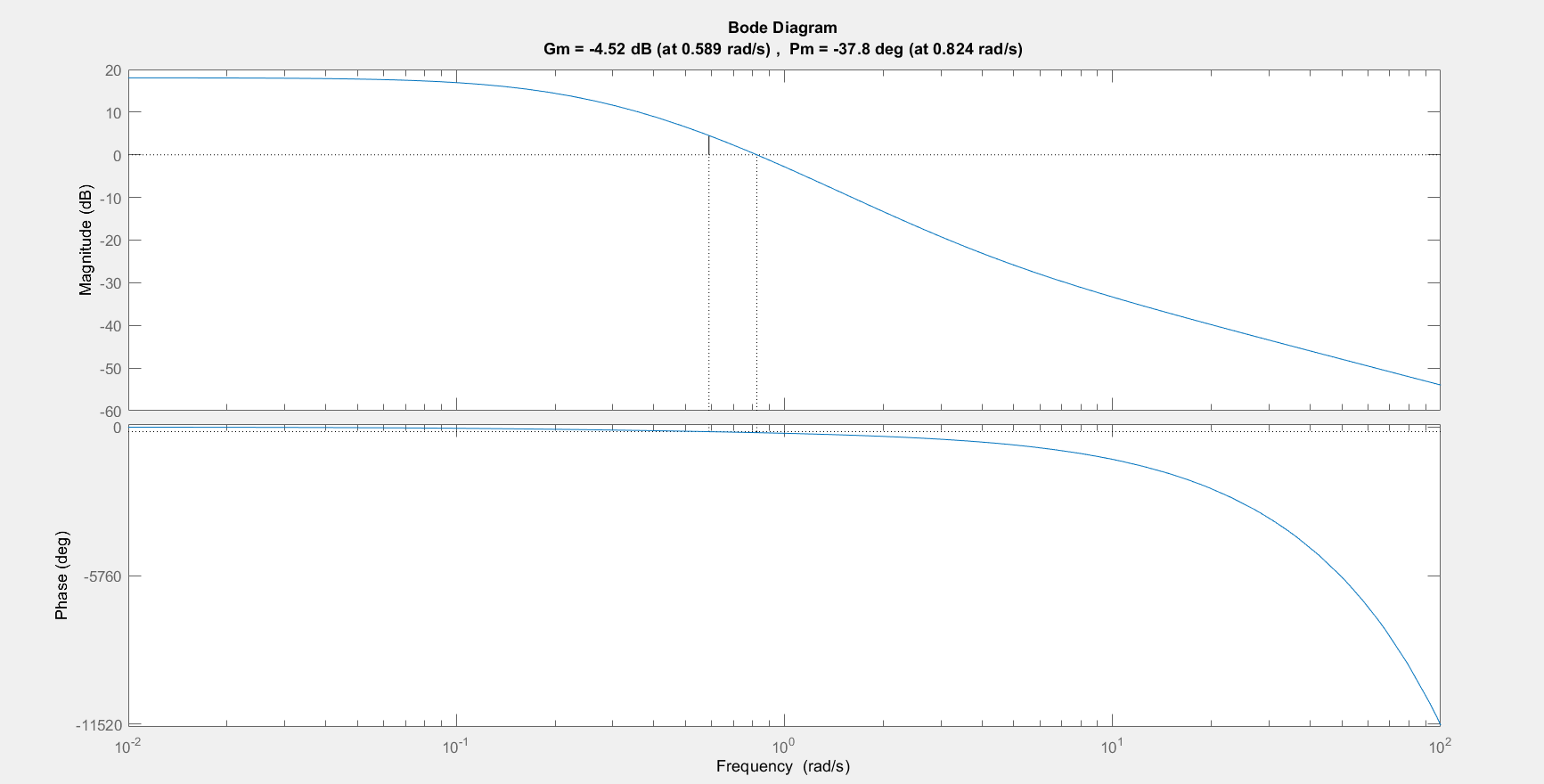
Note: there is a another cross-over point at 1.540\*10^3. If KI>0.38 there will be RHP poles (although there will be some LHP poles also). So the ultimate gain is the gain at the first cross-over point.

# Question 2)

## Question 2) a) P-Control design for given GM

We can use the Bode stability criteria and obtain the Gain margin for KC = 1. So L is simply equal to GP as given below.





*Figure 2.1: Bode Diagram (using ‘margin’ function).*

**GM = -4.52 dB**

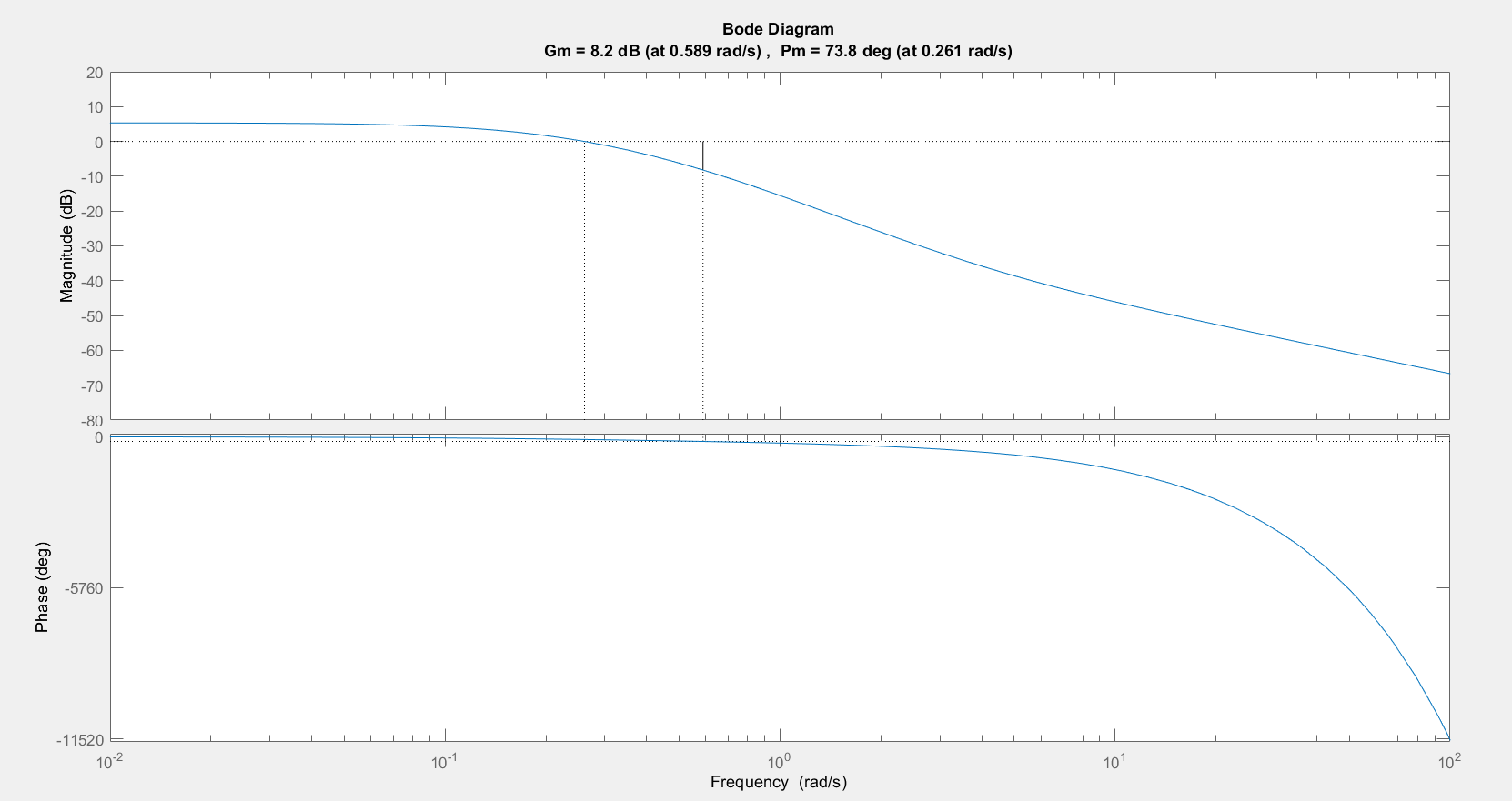
So using this gain margin and the definition of gain margin,

and LG = GCGP = KC\*GP

We can get the required KC as

where GM,required = 8.2 dB

KC was found to be **0.2311**. The corresponding phase margin = **73.815 degrees**

****

*Figure 2.2: Margins of the required P-Controller*

## Question 2) b) Delay uncertainty

Value of delay uncertainty = Phase Margin/gain cross-over frequency

(since, Phase is directly proportional to –w\*Delay)

Gain cross-over frequency = 0.2607 rad/s

=> **Delay Uncertainity = 4.942 units**

## Question 2) c) PI Controller

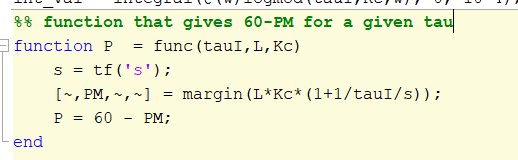
Unlike part a), changing τI changes both magnitude and phase responses of the system.

So one option is to solve for gain cross over frequency and the controller parameter τI simultaneously by imposing

(1)

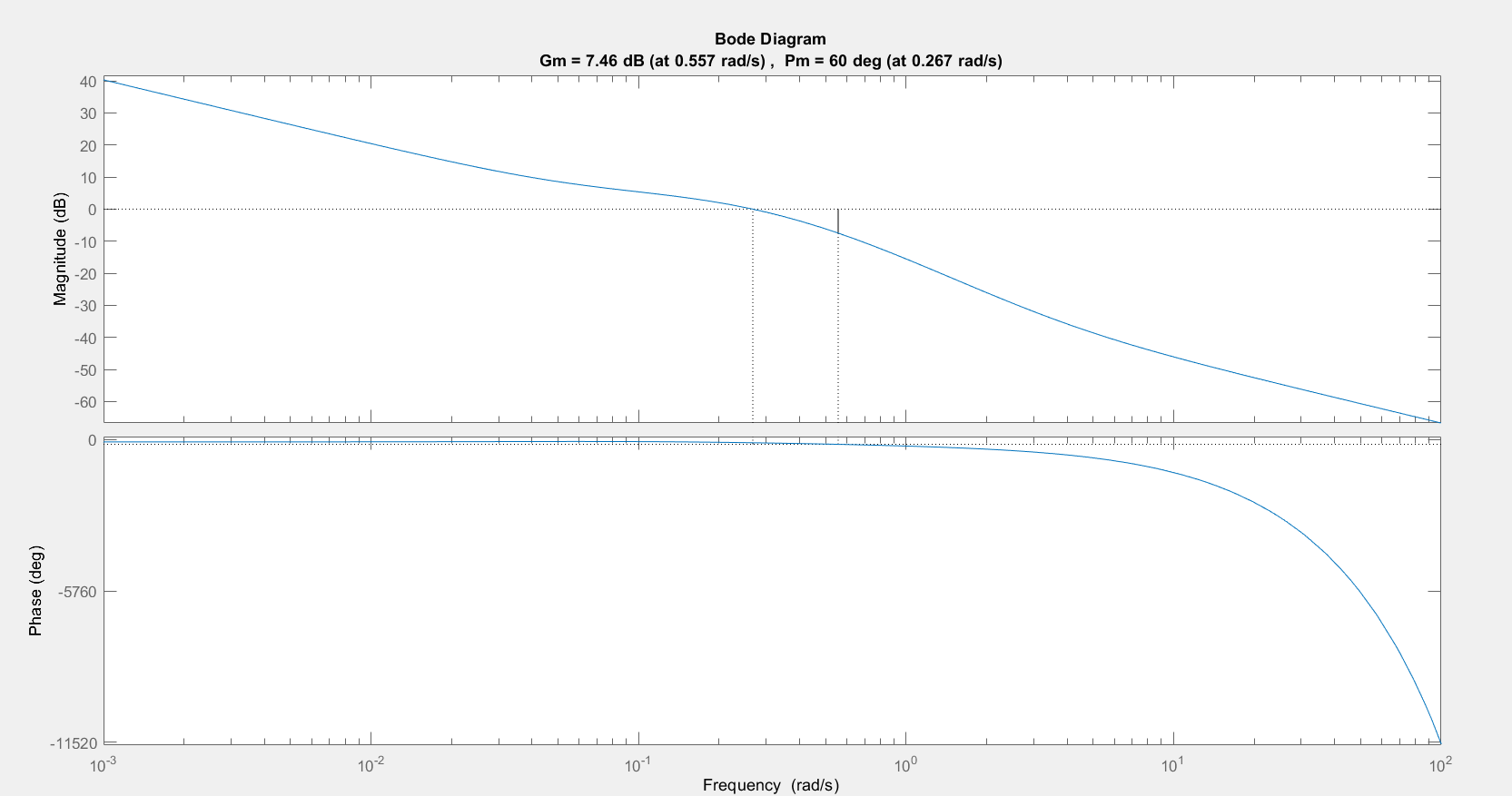
(2)

Instead, I used the function ‘margin’ to compute the phase margin for the closed loop system given a τI. This results in simply solving for tau such that phase is 60.



PI controller parameter τI was found to be **17.8327** units

The Gain margin was found be **7.46 dB**

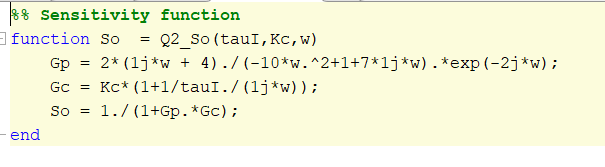
****

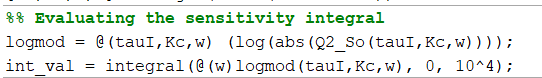
*Figure 2.3: Bode diagram for the closed loop system with the PI controller*

## Question 2) d) Verifying Bode’s Senstitivity Integral

where

The above integral is verified using the following script:





Since it is numerical integration, I replaced infinity with a high value instead. Higher the value, closer it gets to zero. But for too high values (even 105), MATLAB shows up a warning where the error in the estimate is greater than the order of the computed value, so I am restricting to 104

The integral turns out to be, int\_val = -1.8790e-06 which is sufficiently close to 0!

Thus, the Bode’s sensitivity integral holds!