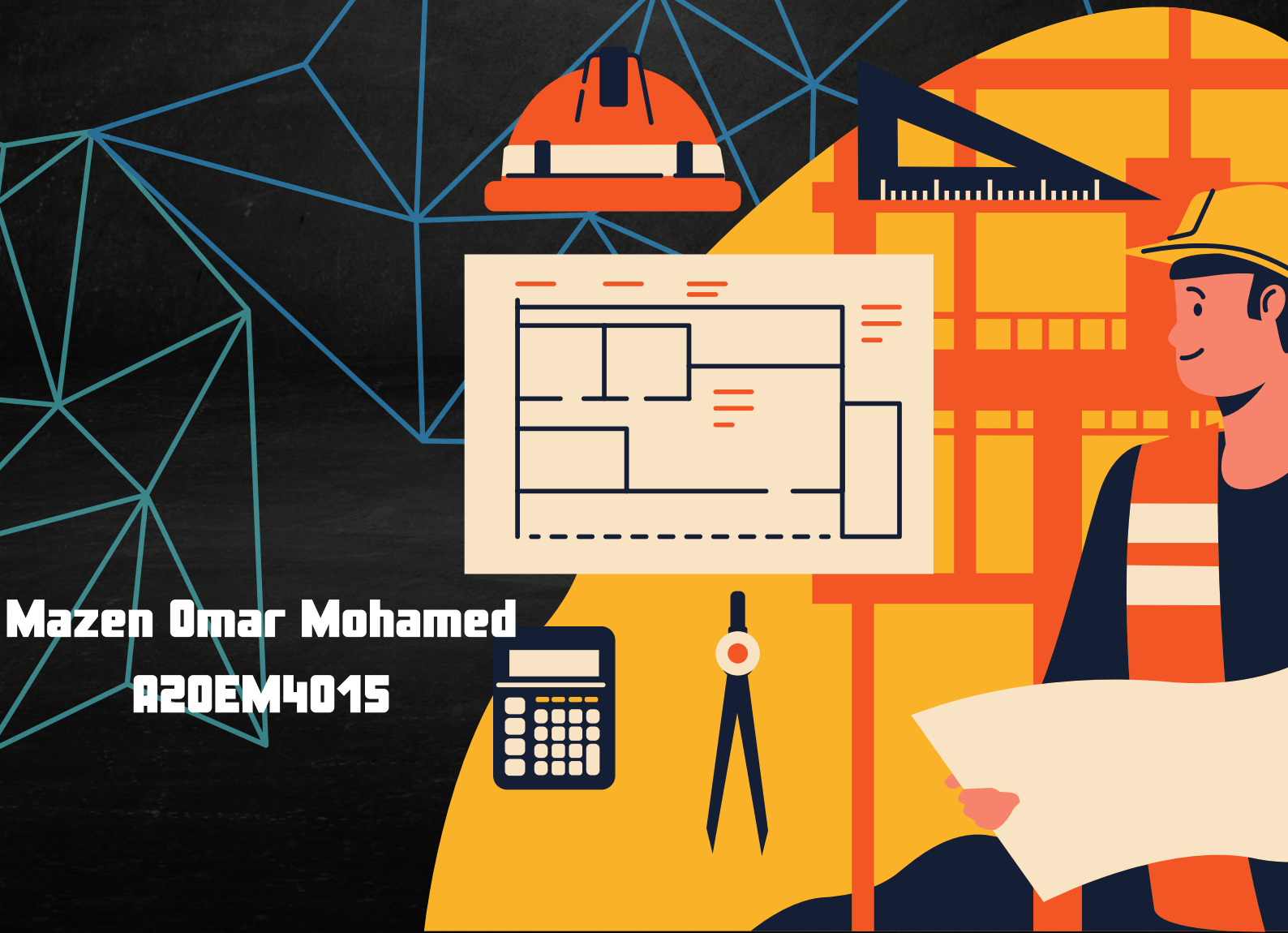
UTM
UNIVERSITI TEKNOLOGI MALAYSIA

Analysis of Time response using M-Code



Mazen Omar Mohamed
A20EM4015

Introduction

Time response and root locus allow us as engineers to review the stability and irregularity of the systems output. Typically, the time response consists of transient and

steady state response, and it enables us to identify the key factors in how the system reacts to its environments. Furthermore, by analyzing the system response, improvements regarding its steady state error and gain can be implemented using PID controllers. The objective of this short report is to analyze the time response and root locus for a system using M-code through Matlab or Octave. Hence, by analyzing the output figure, the key parameters of the time response can be calculated and conclusions regarding the system's stability can be made. The figures generated from M-Code are attached in the appendix, however, this report focuses on explaining the program and briefly delves into the calculations required to obtain the results.

Background

The M-code utilized in this exercise was designed to be open-ended, hence, its parameters can be changed at any time to model any system's time response and root locus. For this particular report, the system to be analyzed is as shown below.

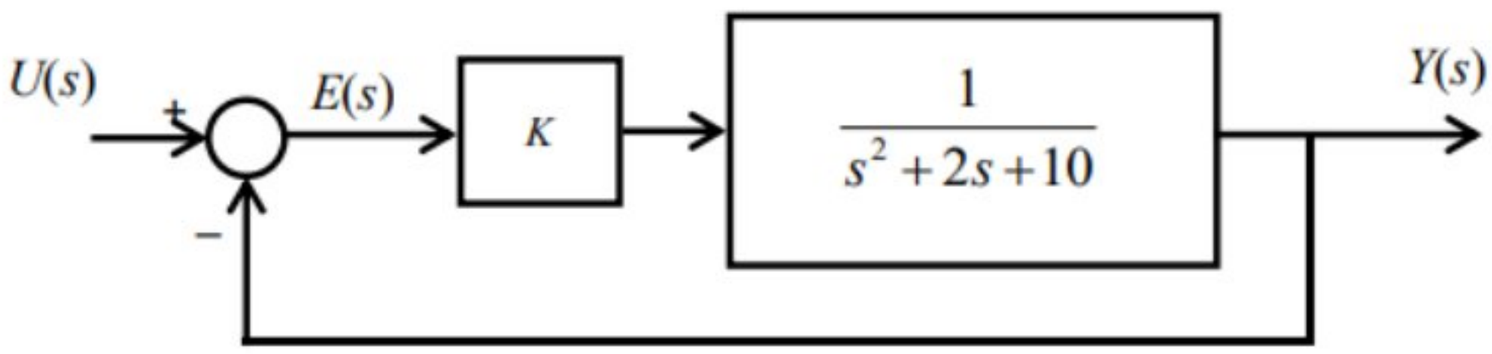
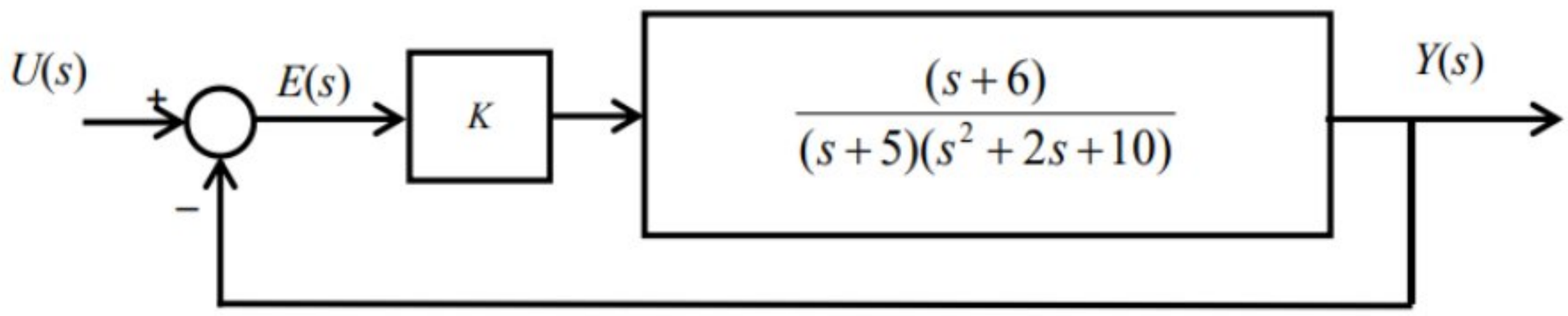


Figure 1. Second order transfer function with step input.

As shown in figure 1, once the time response figure is generated, it is used to calculate the damping ratio, natural frequency, settling time, maximum overshoot and time for the first peak.

Next, several PID controllers are implemented and the new system response is plotted to determine the changes introduced. In this particular program, the controllers tested are P, PI, PD and PID controllers.

Lastly, an enhanced system model is used to generate a root locus and compared with manual root locus plotting, as shown in the figure below.



The program allows changing numerators and denominators at will to generate different root locus and supports more than 2nd order systems.

M-Code

This section focuses on explaining the M-code used to generate the figures. The program begins with asking for several inputs.

```
Rep=1;

num=[input("Please type out the Gain of 2nd Order Systemm (numbers only) : ")];
den=[input("Please type out the coefficients of Denominator, Separated by Commas. : ","s")];
m=cellfun("str2num",strsplit(den,","));

s = tf('s');
m_prime = m(:)';
G = (num*1/(m_prime(1,1)*s^2+m_prime(1,2)*s+m_prime(1,3)*1))

sys = tf(num,m_prime);
```

This input consists of the numerator and denominator for a second order system, however only the coefficients and numbers can be used. For example, the transfer function as shown in figure (1), would only require a numerator input of 1, and a denominator input of 1,2,10. At line 5, the denominator input is separated and converted into a matrix, followed by conversion into a row vector to be used with the transfer function command in Octave.

Once a transfer function (sys) or (s) is made, this function can be used to generate graphs.

```
damp(sys);

figure (1);
grid
t= 0:0.01:10;
step(sys,t)
title(" Time Response of 2nd Order System with Step Input");
legend("Time Response");
```

The code above generates figure (1), with a time range of 0 to 10 seconds, with a step size of 0.01. As can be seen, the transfer function is plotted according to a step input, indicated by step(sys,t) command.

Once the first figure is generated, PID controllers are implemented, however the gain of each controller is subject to the user, and hence can be input using the code shown below.

```
##PID Controllers
print("For Root Locus")
num1=[input("Please type out the Coefficients of Numerator for Transfer Function separated by commas : ","s")];
den1=[input("Please type out the coefficients of Denominator, Separated by Commas. : ","s")];
m1=cellfun("str2num",strsplit(den1,","));
n1=cellfun("str2num",strsplit(num1,","));

m_prime1 = m1(:)';
n_prime1 = n1(:)';
G=tf(n_prime1,m_prime1);
```

A new transfer function is generated as the user inputs the numerator and denominator for root locus plotting and to implement PID controllers.

```
while Rep==1
    %Controllers
    Kp=input('Enter proportional gain: ');
    Ki=input('Enter integral gain: ');
    Kd=input('Enter derivative gain: ');

    ## Different Types of Controllers, P, PI, PD, PID
    A= pid(Kp);
    B= pid(Kp,Ki);
    C = pid(Kp,Ki,Kd);
    D = pid(Kp,Kd);

    T1= feedback(A*G,1);
    T2= feedback(B*G,1);
    T3 = feedback(C*G,1);
    T4 = feedback(D*G,1);
```

Separate variables are made to represent P, PI, PD and PID controllers.

```
figure(2);
%% rootlocus
grid;
rlocus(G);
title("Root Locus")

figure (3);
subplot(2,1,1);
t = 0:0.01:50;
step(T1,t)
title("Time Response after P Controller")

%% rootlocus
subplot(2,1,2);
grid;
rlocus(G);
title("Root Locus Using P Controller")

figure(4);
subplot(2,1,1);
t = 0:0.01:50;
step(T2,t)
title("Time Response after PI Controller")

%% rootlocus
subplot(2,1,2);
grid;
rlocus(G);
title("Root Locus Using PI Controller")

figure(6);
subplot(2,1,1);
```

Figures(2) to Figures(6) are generated for each controller's time response and root locus on a subplot.

```
  Rep=input("Would you like to try different controller gains? (yes=1,no=0) ");
end
```

Lastly, the code ends by asking for Rep input, which signifies the while loop the program runs on. If user inputs 1, the loop is repeated and everything is reset as the user can use the program to redo any figures or generate new ones, if 0 is input, the program ends here while the figures generated can be saved to pdf or png files to be referenced later.

Results

This section aims to demonstrate the aid of this Matlab code in the calculation and analysis of second order systems mainly. The following block diagram visualizes the transfer function that has been used to generate the results interpreted in this report.

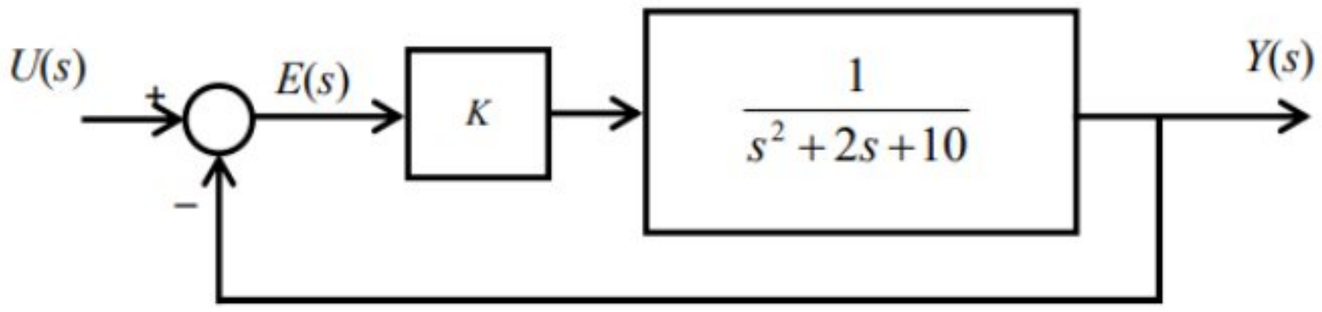


Figure 1. Second order transfer function with step input.

Utilizing the program running on the .m code previously explained; the time response graph was plotted and displayed.

In Figure 2 (appendix), the graph shows a stable time response with decaying oscillations.

For the analysis of said graph, certain parameters were obtained accurately after finding the coordinates by hovering the cursor over the desired point on the generated Figure's window, as seen at the bottom left of Figure 2.

Hovering over the first peak of the graph, the coordinates obtained were (1.044, 0.135). Hence, this indicated our time to first peak time, t1 to be 1.044s. To get the maximum overshoot (Mp) the difference between the maximum amplitude and the amplitude at which the system settled. Hovering over the horizontal (stable) part of the line, the amplitude was found to be 0.1. Therefore, the Maximum Overshoot, Mp is $0.135 - 0.1 = 0.035$; hence the percentage Mp is obtained to be 3.5%.

To determine the settling time (ts) , the time at which the system is almost settled - 98% of the settled amplitude: $[\frac{98}{100} \times 0.1 = 0.098]$ - is found. Using Figure 2, the time at which the amplitude is 0.098 was found to be approximately 4.176s. This is confirmed by the mathematical method that can be used to obtain the settling time, ts.

$$ts = \frac{4}{\xi\omega}$$

Upon obtaining the poles of the equation from the program output, the pole was obtained to be $[-1 \pm 3i]$, which can be used to determine that $\xi\omega = 1$ according to the format of the poles.

$$s_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$$

Hence $ts = \frac{4}{1} = 4s$ mathematically.

The damping ratio and natural frequency was generated by the matlab program to be 0.316ξ and 3.16 respectively. Therefore, the damped frequency can be calculated using the following equation.

$$\omega_d = \omega_n\sqrt{1-\xi^2}$$

$$\omega_d = 3.16\sqrt{1 - 0.316^2}$$

$$\omega_d = 2.998$$

This is once more confirmed by the poles generated by the program; which was found to be 3 (the imaginary part of the complex pole).

Upon introducing P, PI, PD and PID controllers, the time response of the function was regenerated to show their effect. Refer to the Appendix.

Controllers help reduce error to the point of elimination with enough applied gain. An interactive controller combining proportional and integral controllers was used to drastically reduce the error in the system without affecting its transient characteristics.

Upon testing, which was made easy by the repeat functionality integrated in the matlab program, the error was found to greatly diminish when the proportional gain is reduced to Kp=1 while maintaining the integral gain at Ki=5. This does not completely eliminate the error, however, greatly reduces it.

Following the consideration of environmental disturbance; a new transfer function is to be analyzed.

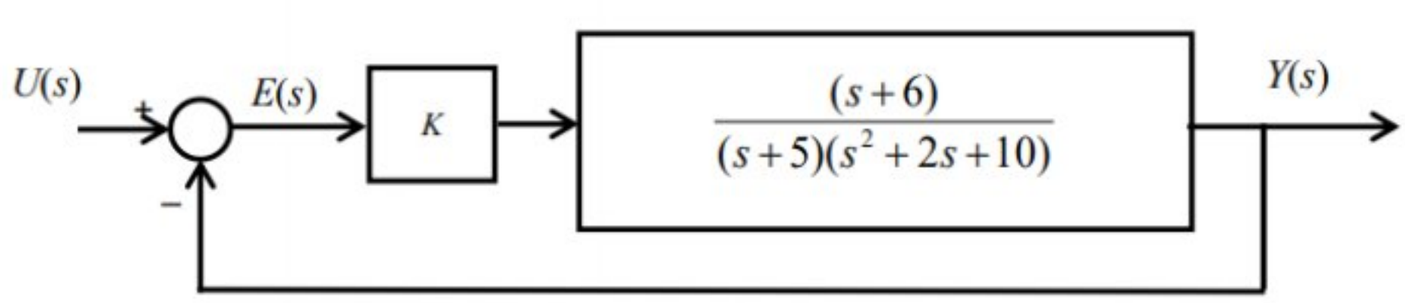


Figure 8. Transfer function block diagram after environmental disturbance consideration.

The new coefficients were input into the program and the root locus of the transfer function was generated. The root locus of this function was also sketched manually in Figure 10 attached in the Appendix.

Discussion

1. Time Response Graph Analysis:

The system's behavior in terms of time is displayed through the Time Response graph. This graphic provides clues about various KPIs related to a control process. In this case, the system under consideration has an overshoot time of 0.098s settling period; a maximum over-reaction (Mp) equal to 3.5%; a Mp %26 and t1 makes for: The USSTS occurs within seconds while OTP is in milliseconds range with only one thousandth higher accuracy than TOC.

In the relative time response graph, settling is indicated by the amount of accessory period during which this process (steady state) took place before it established final value-that usually should be within range 5%of this ultimate. In this instance, the system relaxes in 0.98 seconds. The maximum overshoot (Mp) signifies the maximum amplitude reached by the system's response, which, in this scenario, is 0.035. The percentage maximum overshoot (Mp%) expresses this overshoot as a percentage of the final steady-state value, revealing the system's response's relative peak. Here, it is 3.5%. The time for the first peak (t1) is the duration it takes for the system's output to reach the first peak in its response, occurring at 1.044 seconds.

2. Time Response with Different Controllers:

a. Proportional (P) Controller:

The P controller introduces gain that directly influences the amplitude of the response. The shape of the system response remains comparable to the original graph, as seen in Figure 2 of the appendix. Analyzing the graph in terms of settling time and overshoot provides a basis for comparison. The impact of the proportional controller on settling time and overshoot is crucial for understanding its effectiveness in achieving desired system performance.

b. Proportional-Integral (PI) Controller:

The integral controller in the PI configuration alters the shape of the response graph. The first peak is no longer the highest, indicating a change in the system's behavior. The longer settling time and oscillatory behavior before stability suggests that the integral controller introduces additional dynamics. Evaluating the impact on settling time and peak amplitude provides insights into the trade-offs involved in employing integral control.

c. Proportional-Derivative (PD) Controller:

The PD controller's response, resembling the PI controller, requires a nuanced examination. Differences in settling time and overshoot, though subtle, may signify the influence of derivative control. Scrutinizing the system's response to the PD controller allows comparative analysis, revealing the unique characteristics introduced by derivative control.

d. Proportional-Integral-Derivative (PID) Controller:

The PID controller demonstrates decreased oscillations and lower errors. Despite the increased settling time, the system's response highlights improved stability and reduced overshoot. Evaluating how the PID controller achieves these improvements in comparison to other controller's sheds light on the synergistic effects of proportional, integral, and derivative control.

3. Controller Comparison:

In comparing the controllers, settling time, overshoot, and other relevant characteristics are crucial factors. The P controller, being the simplest, may exhibit faster settling times but may suffer from overshooting. The PI controller introduces integral action to reduce steady-state error but may extend settling times. The PD controller, emphasizing damping, could exhibit similar characteristics to PI. The PID controller aims to strike a balance between these aspects. A comprehensive comparison facilitates selecting the most suitable controller based on the system's requirements.

4. Proportional Gain Adjustment for PI Controller:

a. After Adjusting Gain to Reduce Error:

Fine-tuning the proportional gain in the PI controller refines the system's response. The reduction in oscillations and amplitude suggests an improved balance between responsiveness and stability. While the settling time is extended, the transition from 0 to 1 is notably smoother. The near elimination of error indicates the effectiveness of the gain adjustment. Analyzing the time to settle post-adjustment provides insights into the trade-offs involved in achieving improved performance.

b. Time to Settle:

Determining whether the system took more or less time to settle after the gain adjustment allows for a quantitative assessment of the trade-offs made. Balancing the desire for rapid settling with the need for stability is a key aspect of control system design.

5. Locus Figure Analysis:

The locus figure provides a detailed depiction of the system's poles and zeros in the complex plane. A pole in the real axis signifies a system response with no oscillations, while complex conjugate poles introduce oscillatory behavior. The trajectory from pole to zero in the real axis and the angle of departure for conjugate complex poles depict the system's transient response and overall stability.

As the locus moves along asymptotes, it indicates the system's response under varying parameters. Understanding the locus figure is crucial for predicting and optimizing the system's behavior under different conditions.

In summary, the time response graphs, and locus figures are pivotal tools for understanding and optimizing the dynamics of a second-order control system under different controllers. Each graph and controller configuration tells a unique story about the system's behavior, providing valuable insights for control engineers seeking to design systems with specific performance characteristics.

Appendix

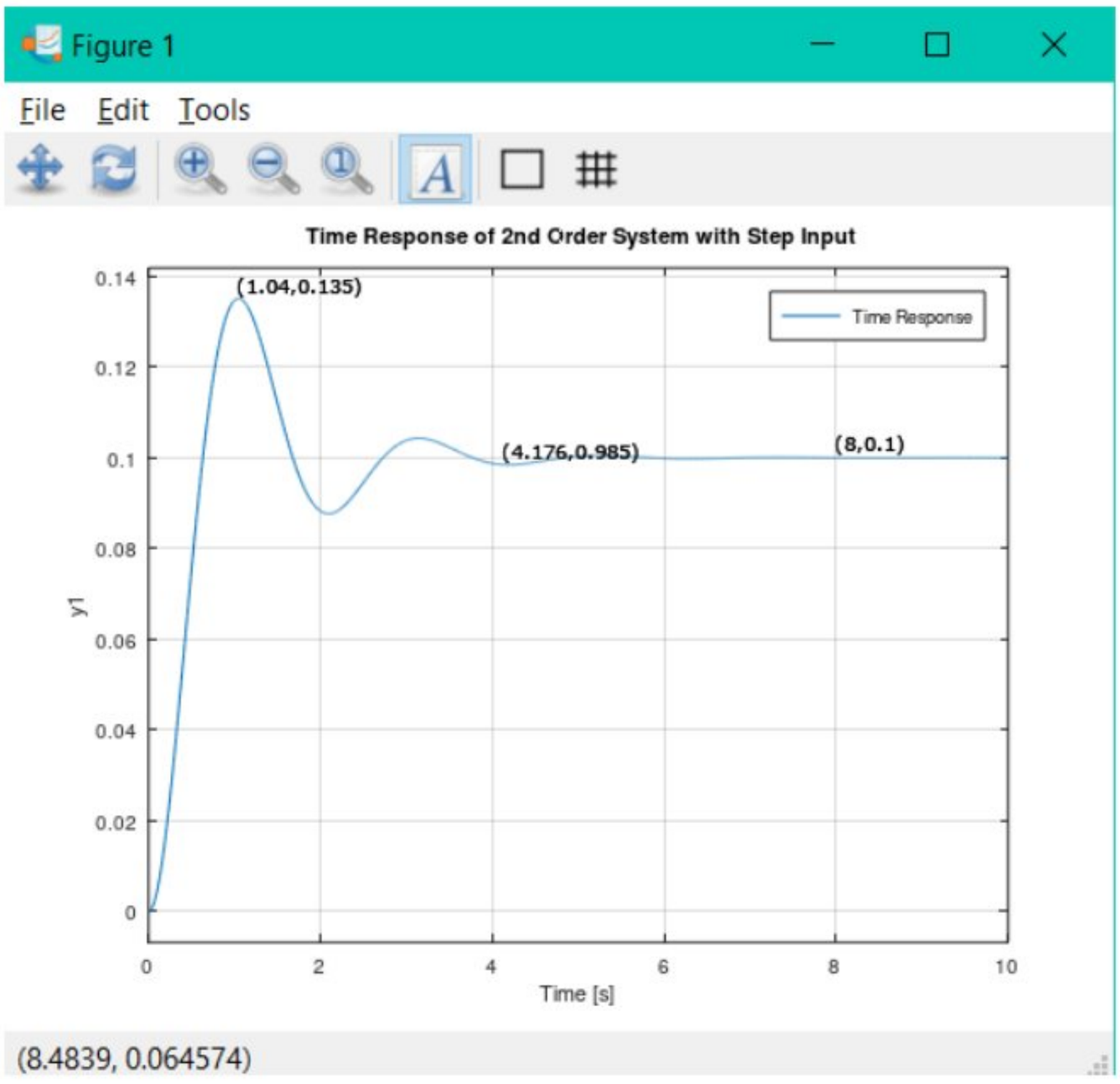


Figure 2. Time response of 2nd order system with step input and gain, K=1.

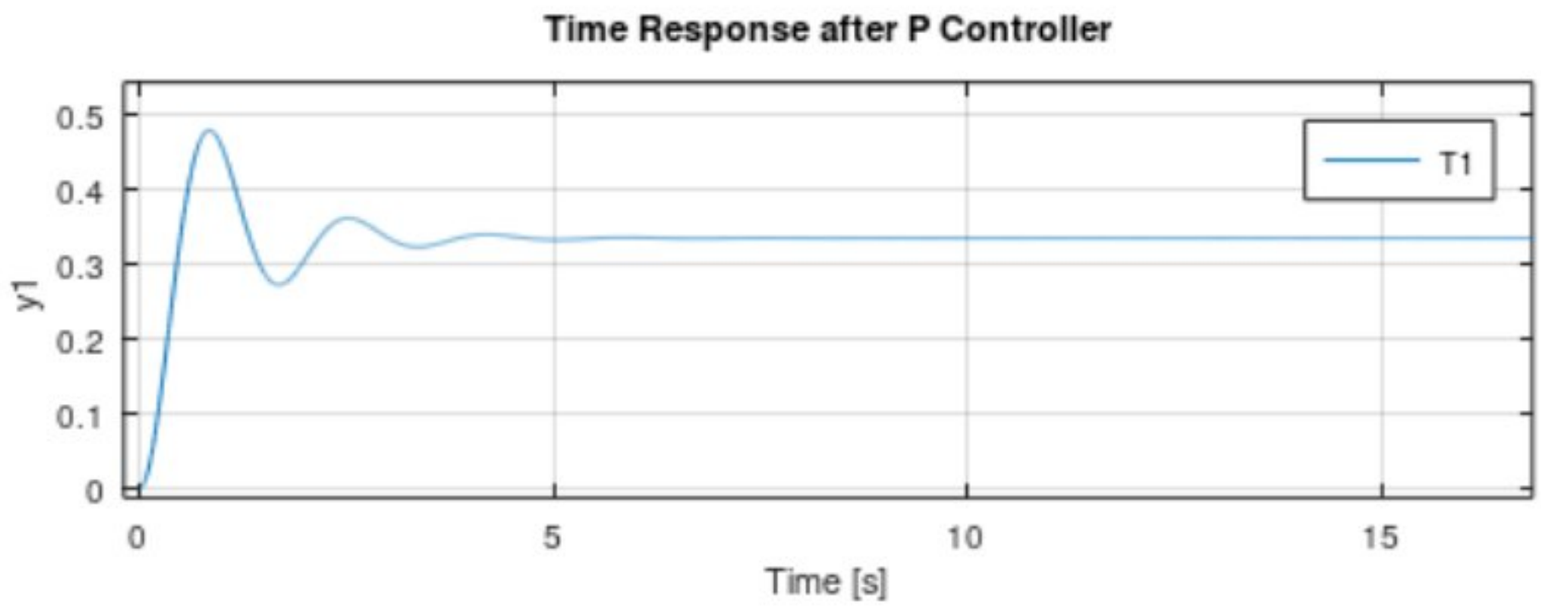


Figure 3. Transfer function time response after applying P controller with gain, Kp=5.

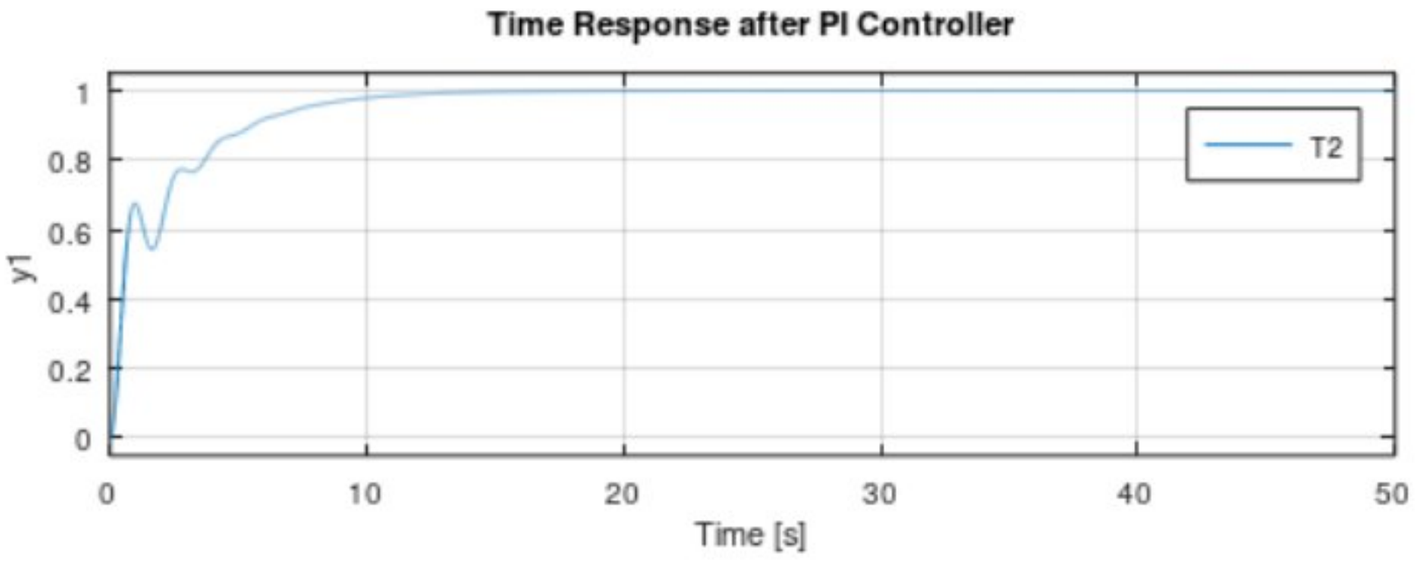


Figure 4. Transfer function time response after applying PI controller with proportional and integral gains, Kp=5 and Ki=5, respectively.

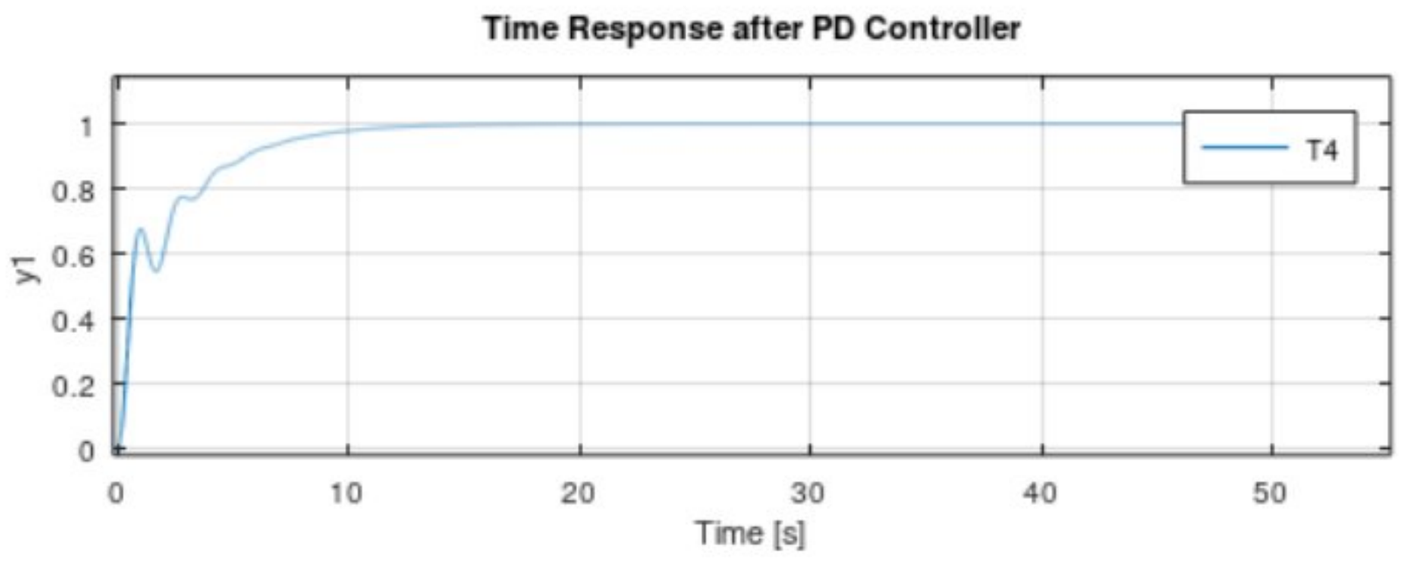


Figure 5. Transfer function time response after applying PD controller with proportional and derivative gains, Kp=5 and Kd=5, respectively.

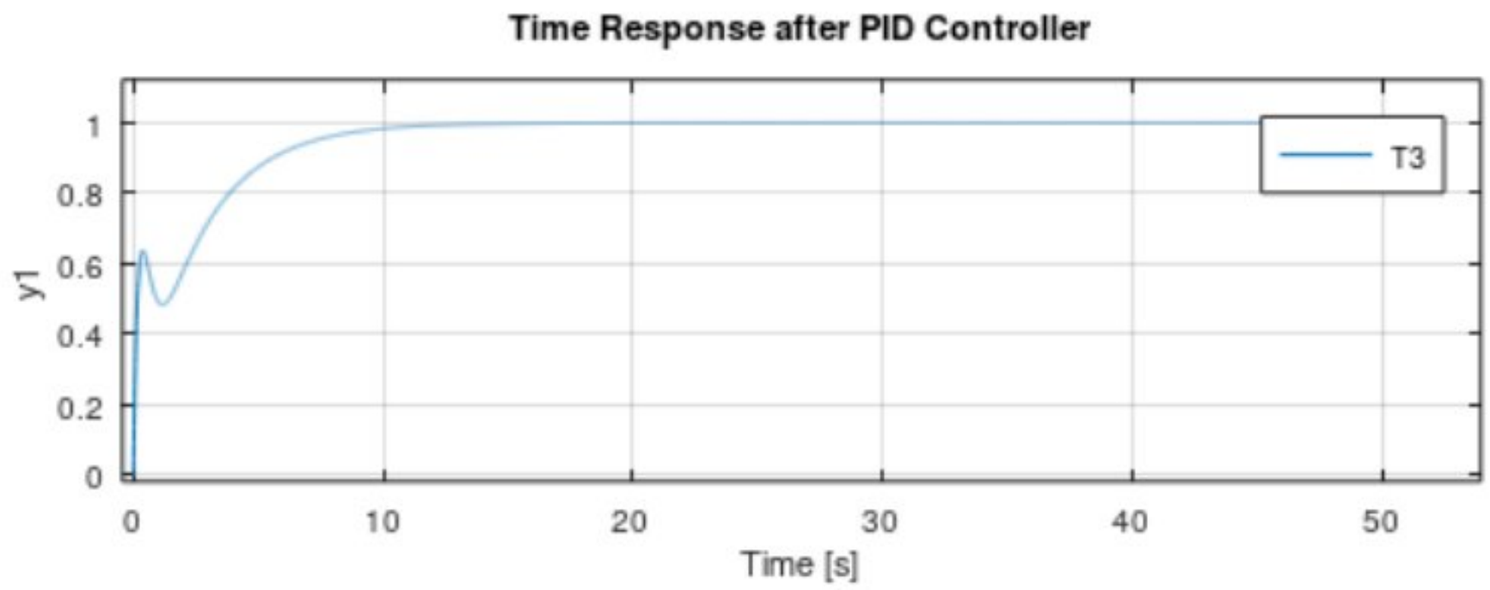


Figure 6. Transfer function time response after applying PID controller with proportional, integral and derivative gains, Kp=5, Ki=5 and Kd=5, respectively.

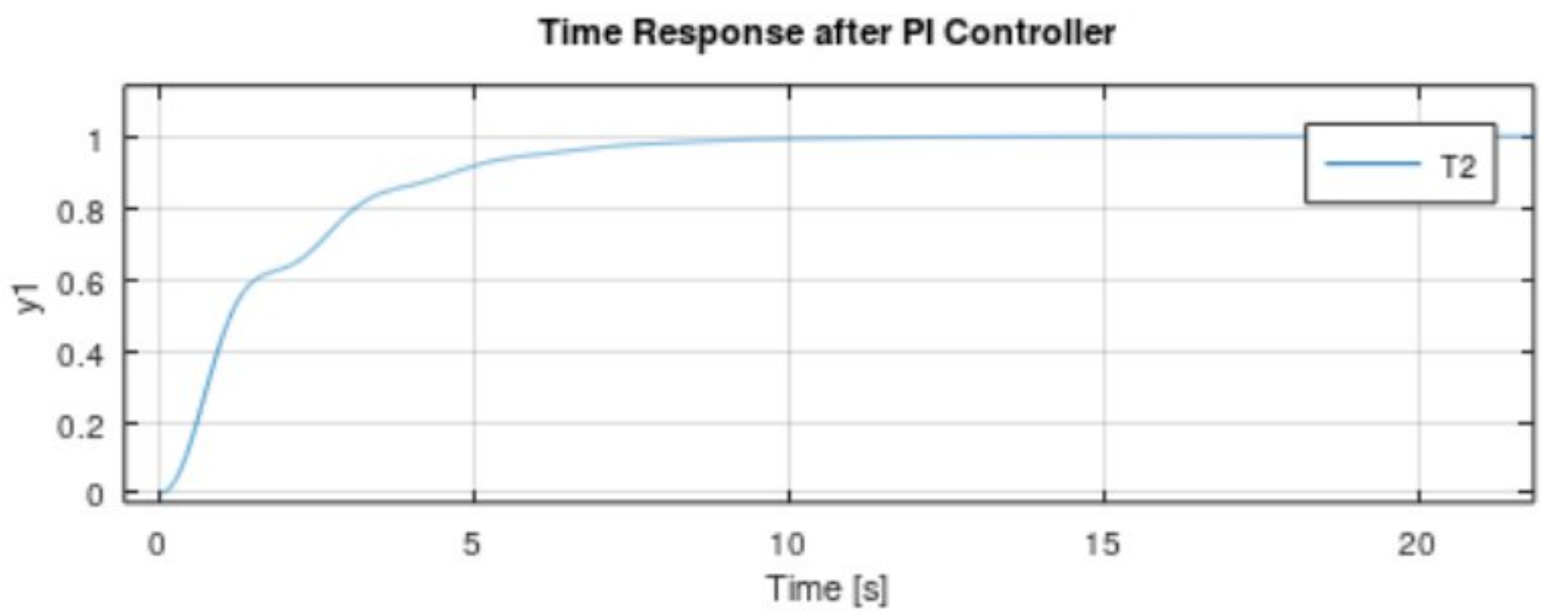


Figure 7. Transfer function time response after applying PI controller with proportional and derivative gains, Kp=1 and Kd=5, respectively.

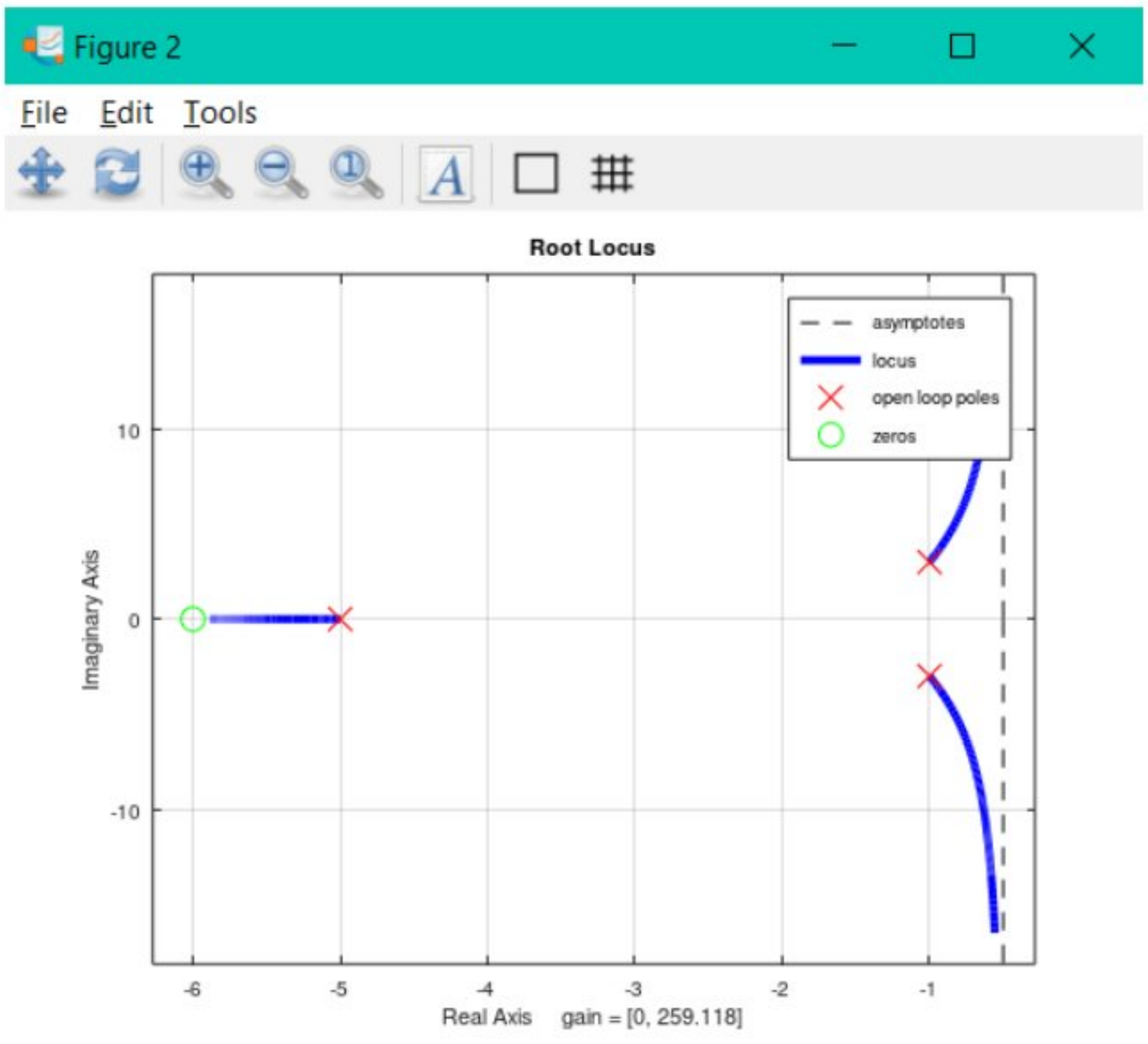


Figure 9. Root locus of new transfer function in figure 8, as K varies from 0 to ∞.

$$TF = \frac{kG(s)}{1 + kG(s)H(s)}$$

$$kG(s) = \frac{k(s+6)}{(s+5)(s^2+2s+10)}$$

poles:

$s = -5$

$s = -1+3i$ ← Complex Poles

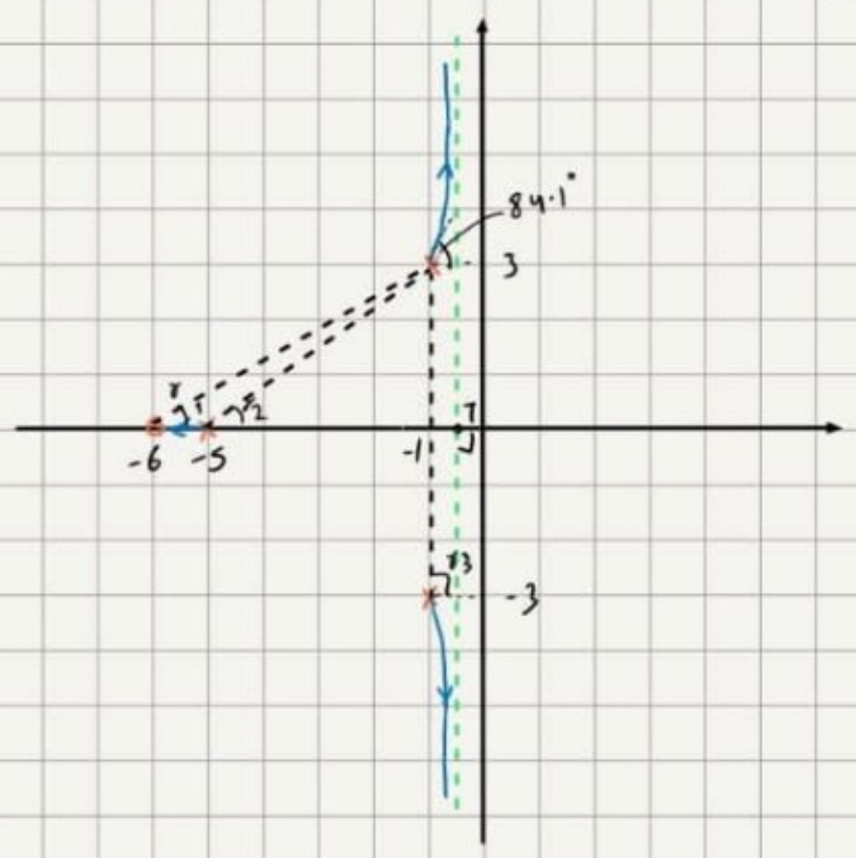
$s = -1-3i$ ←

zeros:

$s = -6$

$n = 3$

$m = 1$



* Get Asymptote angles:

$$\alpha = \frac{180(2k+1)}{n-m} \qquad k = 0, 1, \ldots \; {}^{3-1-1}_{n-m-1}$$

$= 0, 1$

$$\alpha_1 = \frac{180(2(0)+1)}{3-1}$$

$= 90°$

$$\alpha_2 = \frac{180(2(1)+1)}{3-1}$$

$= 270°$ or $-90°$

* Get Asymptote centroid:

$$cp = \frac{\Sigma n - \Sigma m}{n-m}$$

$$= \frac{(-5 + -1+3i + -1-3i) - (-6)}{3-1}$$

$$= \frac{-7+6}{2}$$

$$= -\frac{1}{2} = -0.5$$

* Angle of departure:

$\theta_1 = \tan^{-1}\left(\frac{3}{5}\right)$

$= 30.96°$ zero

$\theta_2 = \tan^{-1}\left(\frac{3}{4}\right)$

$= 36.87°$ pole

$\theta_3 = 90°$ pole

angle of departure $= 180 - 90 - 36.87 + 30.96$

$= 84.09°$

Figure 10. Manual calculation and plotting of root locus of transfer function in figure 8.