

Control Theory Homework 3

Daniil Fronts, Group 3, Variant F

№2

$$System : x'' + \mu x' + kx = u$$

$$\mu = 63, k = 15$$

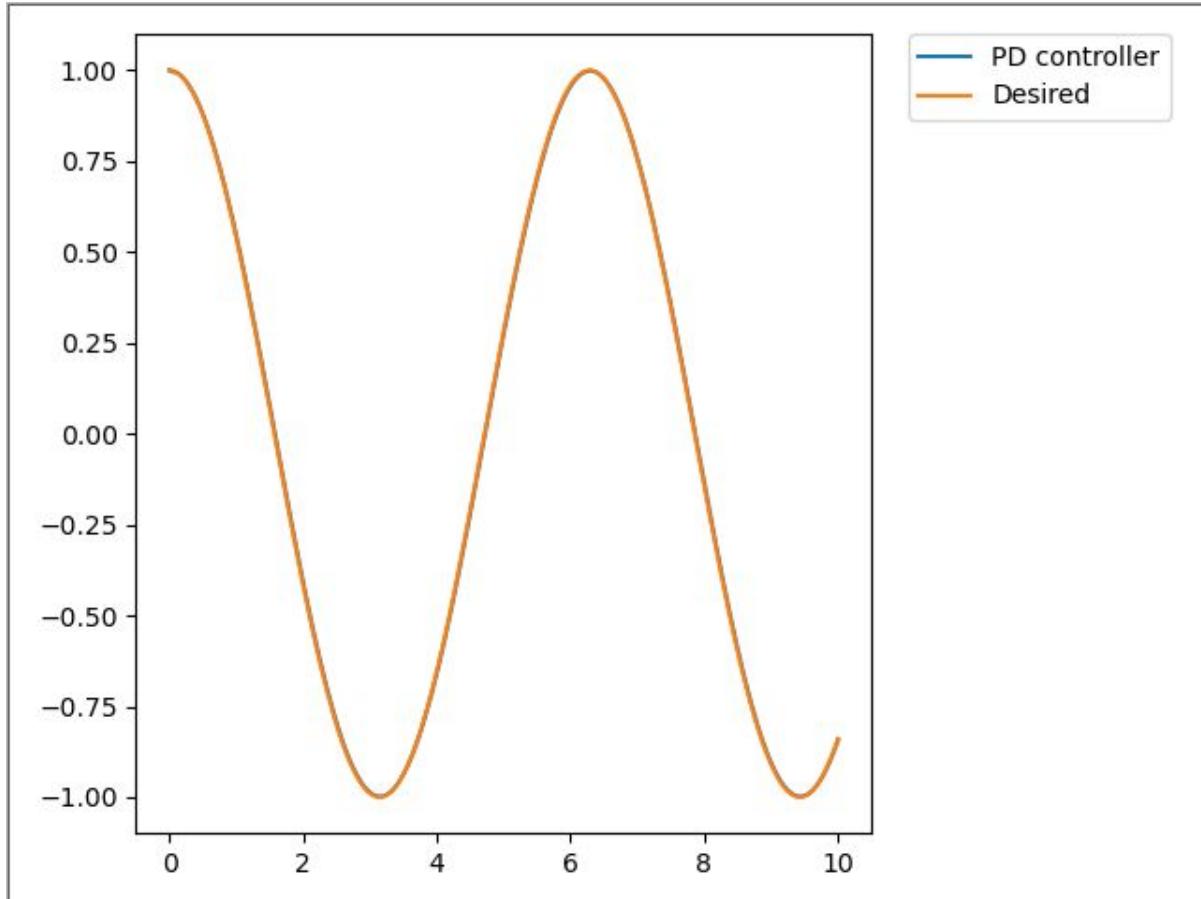
$$x'' + 63x' + 15x = u$$

$$\begin{bmatrix} x' \\ x'' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -15 & -63 \end{bmatrix} \begin{bmatrix} x \\ x' \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

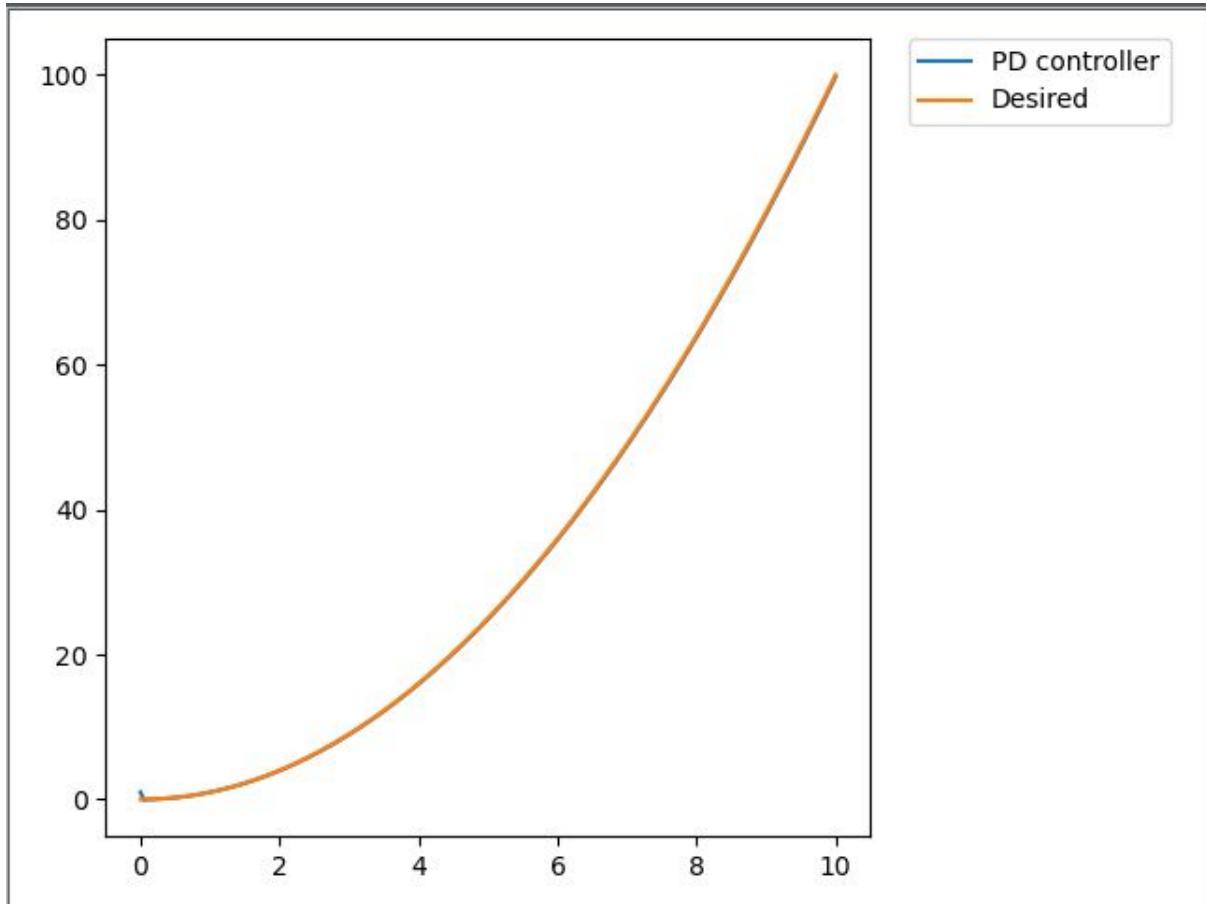
A)

Design PD-controller that tracks time varying reference states i.e. $[x^*(t), \dot{x}^*(t)]$ as closely as possible. Test your controller on different trajectories, at least two. System: $\ddot{x} + \mu\dot{x} + kx = u$, see variants below.

1) Trajectory: $\cos(t)$, $k_p = 9000$, $k_d = 10$

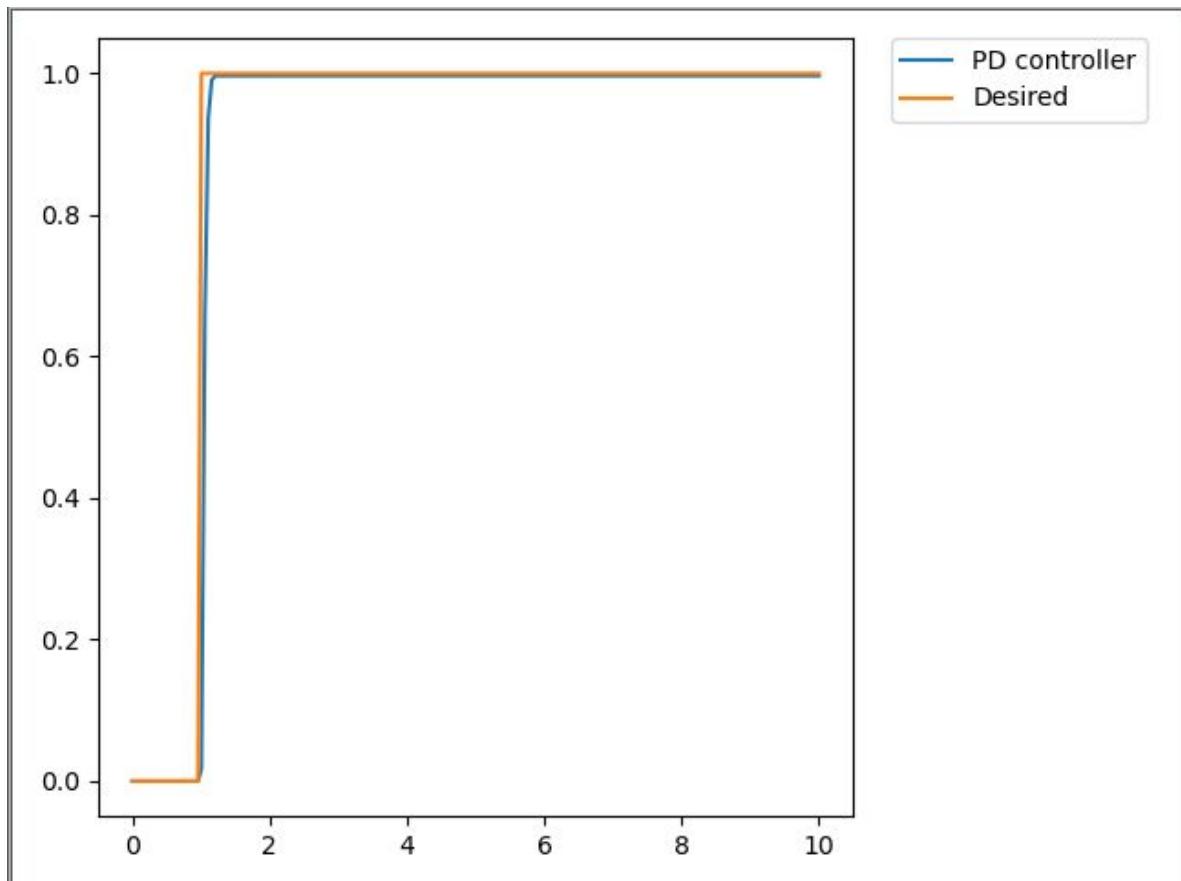


2) Trajectory: t^2 , $k_p = 9000$, $k_d = 10$



B)

Tune controller gains k_p and k_d . Find gains that provide no oscillations and no overshoot. Prove it with step input.



$$k_p = 1500, k_d = 10$$

C)

System is stable if real parts of complex eigenvalues of $AB - K$ is less than zero or real eigenvalues is less than zero.

```
eigenvalues of A - BK: [-36.5+13.51850583j -36.5-13.51850583j]
```

Thus, system is stable.

D)

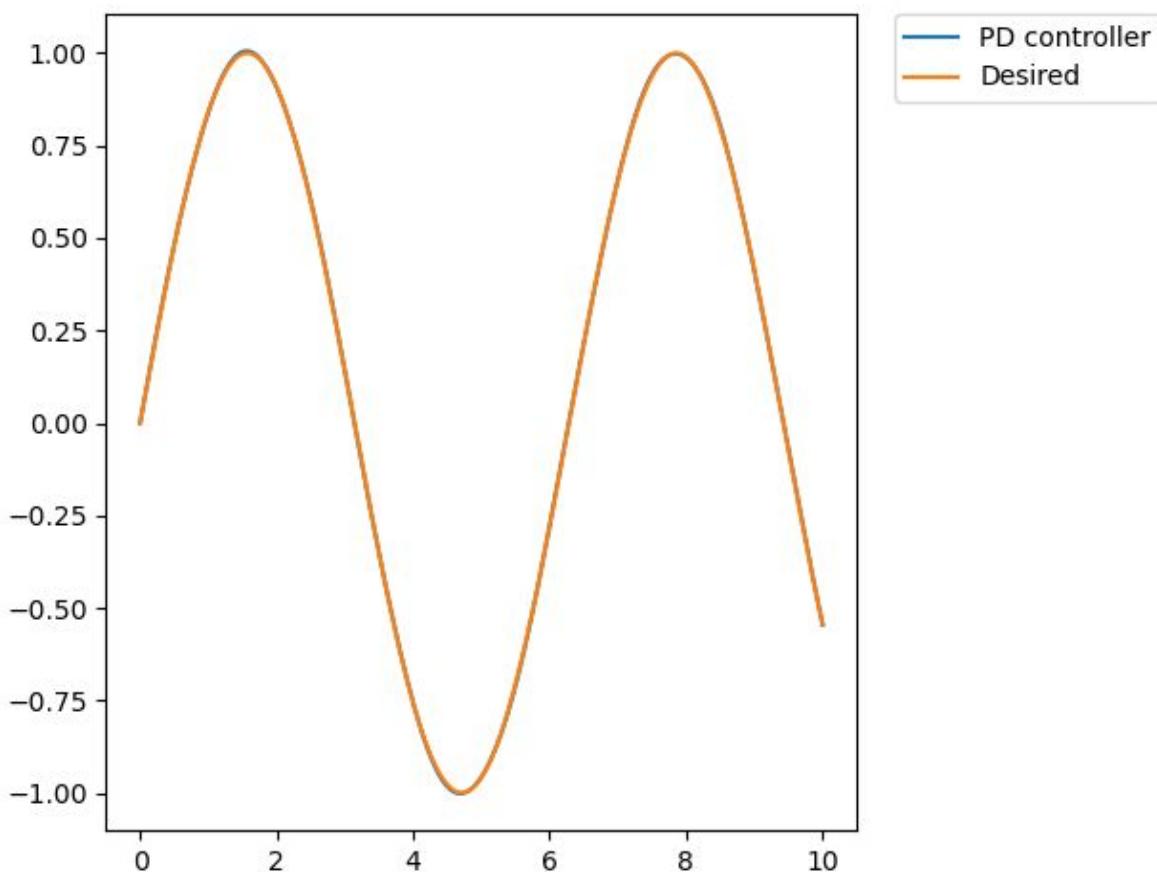
Think of how you would implement PD control for a linear system:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 10 & 3 \\ 5 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

To implement PD control for that system we should add controller.

$$\begin{bmatrix} x' \\ x'' \end{bmatrix} = \begin{bmatrix} 10 & 3 \\ 5 & -5 \end{bmatrix} \begin{bmatrix} x \\ x' \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

So, with Trajectory: $\sin(t)$, $k_p = 20000$, $k_d = 5$



E)

Implement a PI/PID controller for the system: $\ddot{x} + \mu\dot{x} + kx + 9.8 = u$ (variants are below) Test your controller on different trajectories, at least two.

$$System : x'' + \mu x' + kx + 9.8 = u$$

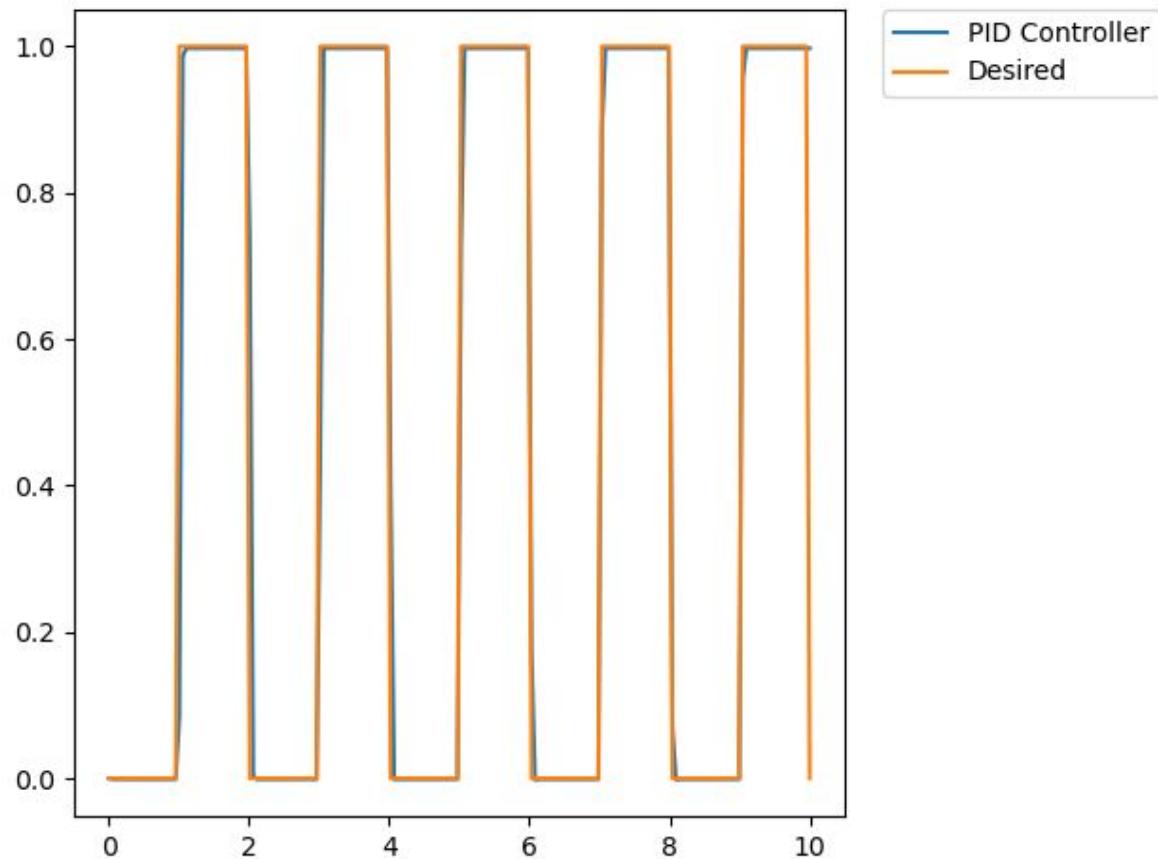
$$\mu = 63, k = 15$$

$$x'' + 63x' + 15x + 9.8 = u$$

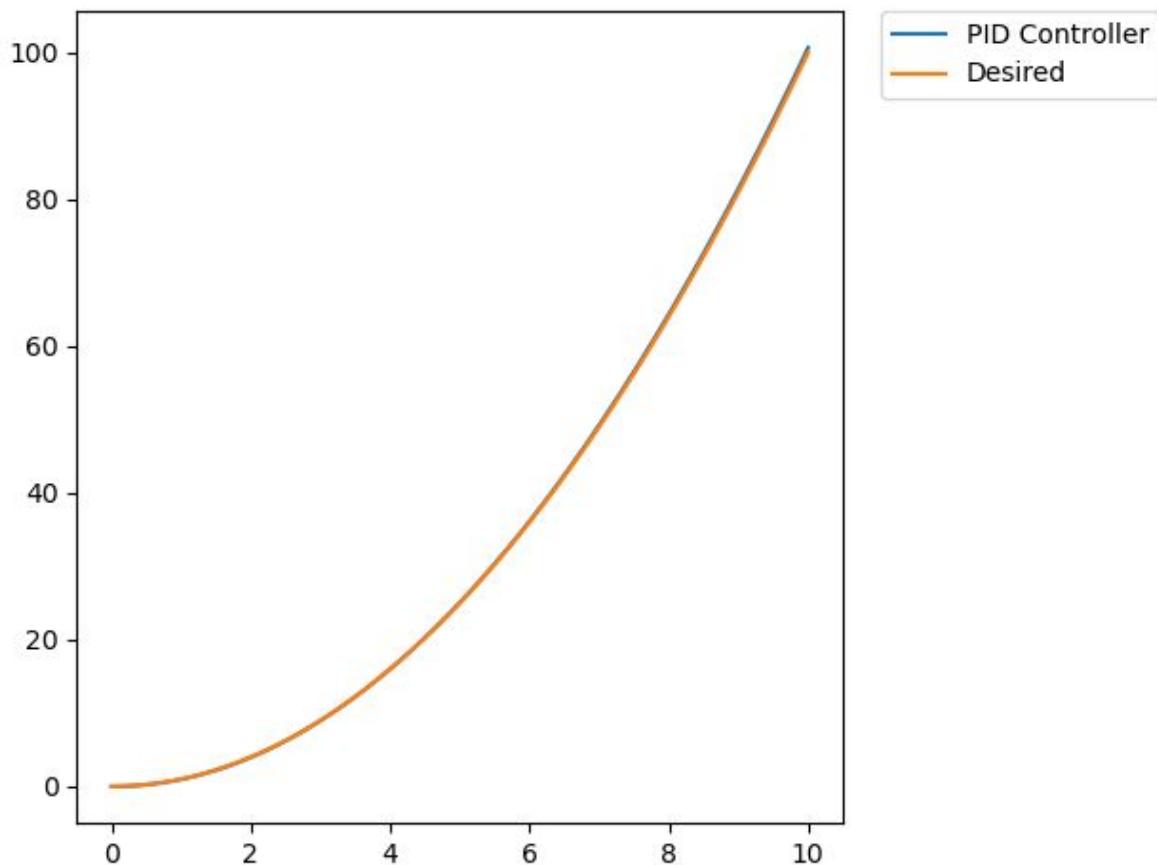
$$u_1 = u - 9.8$$

$$\begin{bmatrix} x' \\ x'' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -15 & -63 \end{bmatrix} \begin{bmatrix} x \\ x' \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_1$$

1) Trajectory: $\text{math.floor}(t) \% 2$, $k_p = 9000$, $k_d = 110$, $k_i = 0.3$



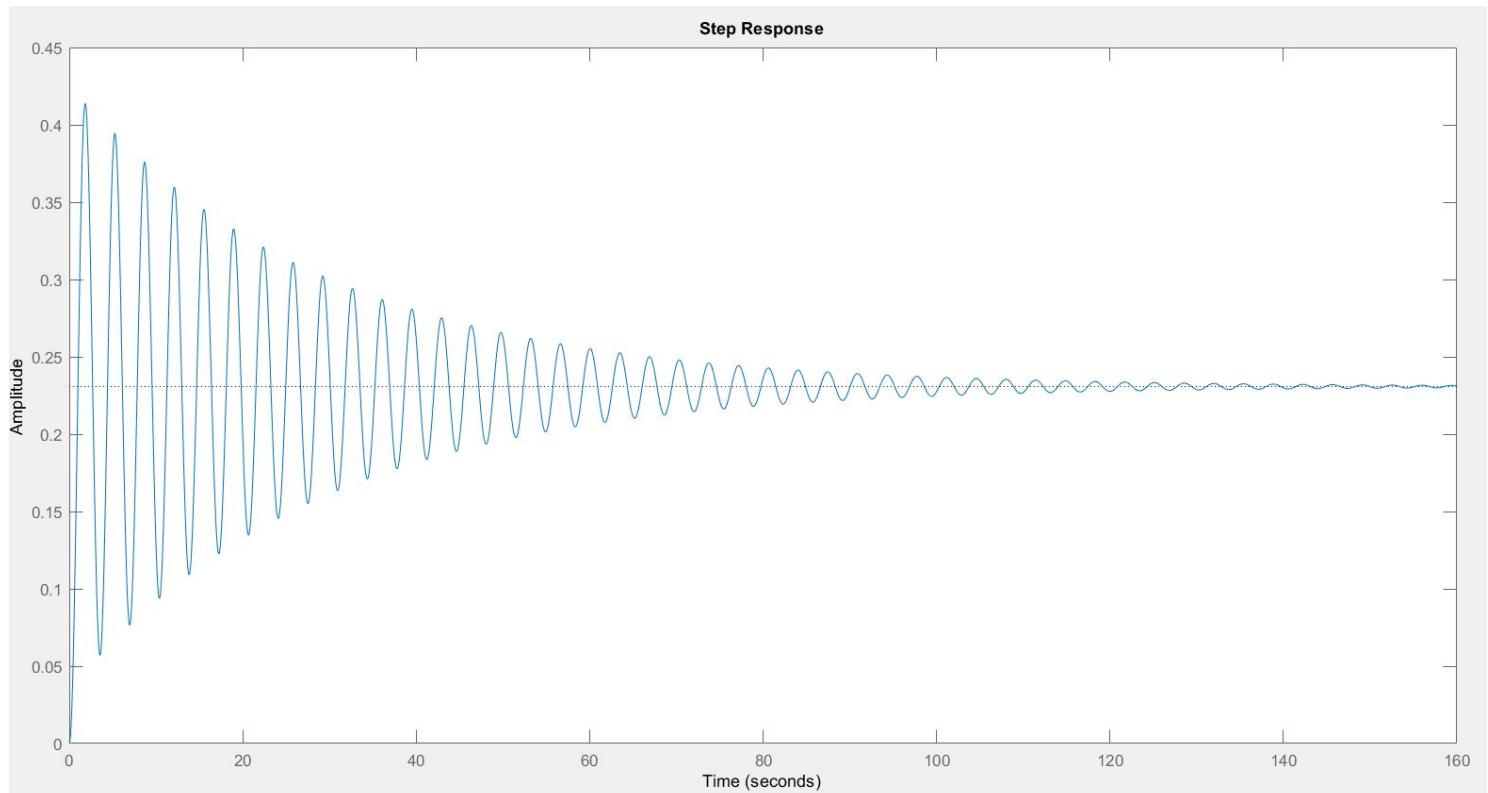
2) Trajectory: t^2 , $k_p = 9000$, $k_d = 110$, $k_i = 0.3$



№3

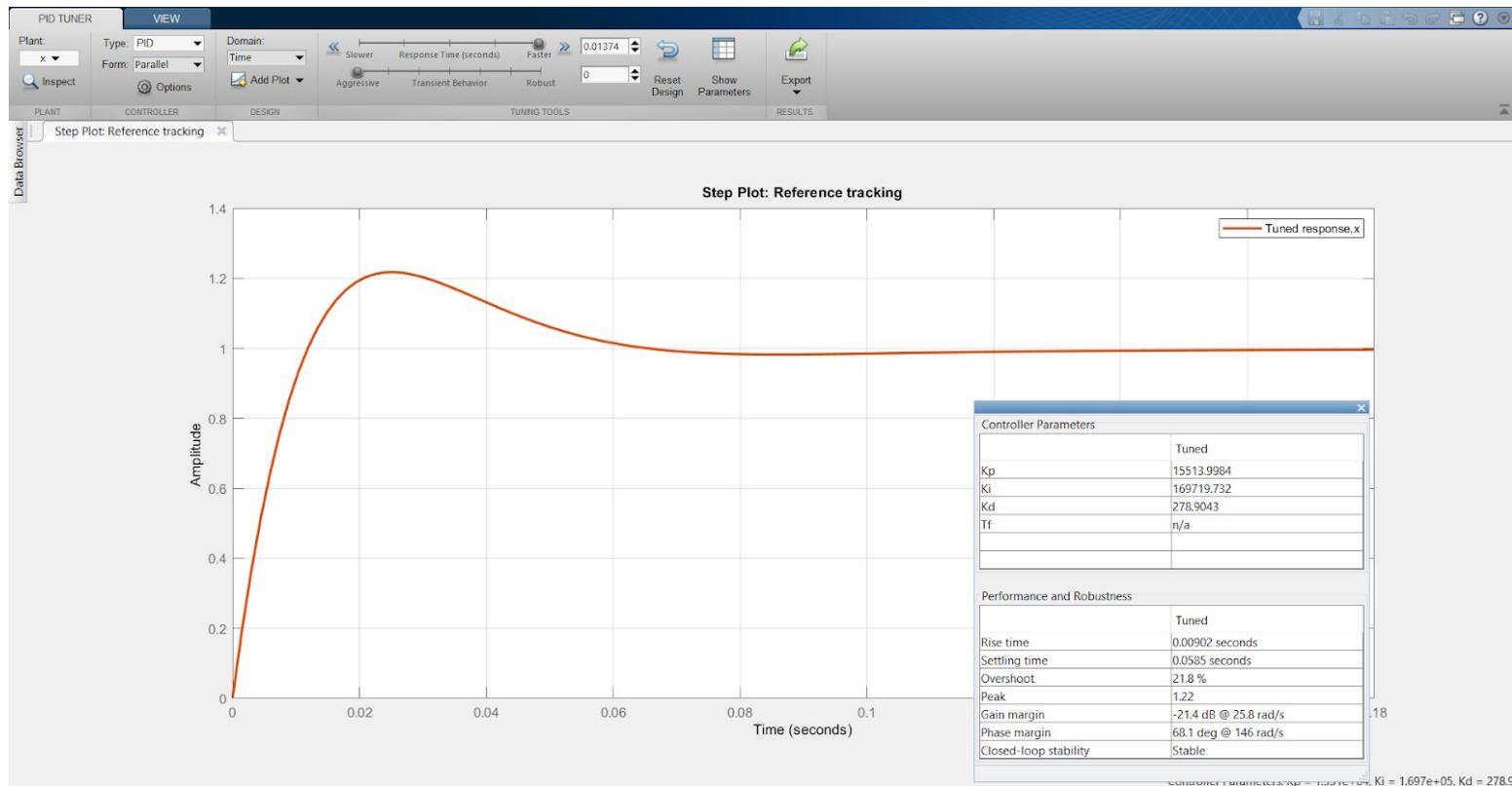
Design a PID controller. Use step input function and try to improve rise time, overshoot and steady-state error, comparing with no controller system. Describe your actions. Use pidTuner in Matlab.

$$(f) \quad W = \frac{s+3}{2s^3+4s^2+7s+13}$$

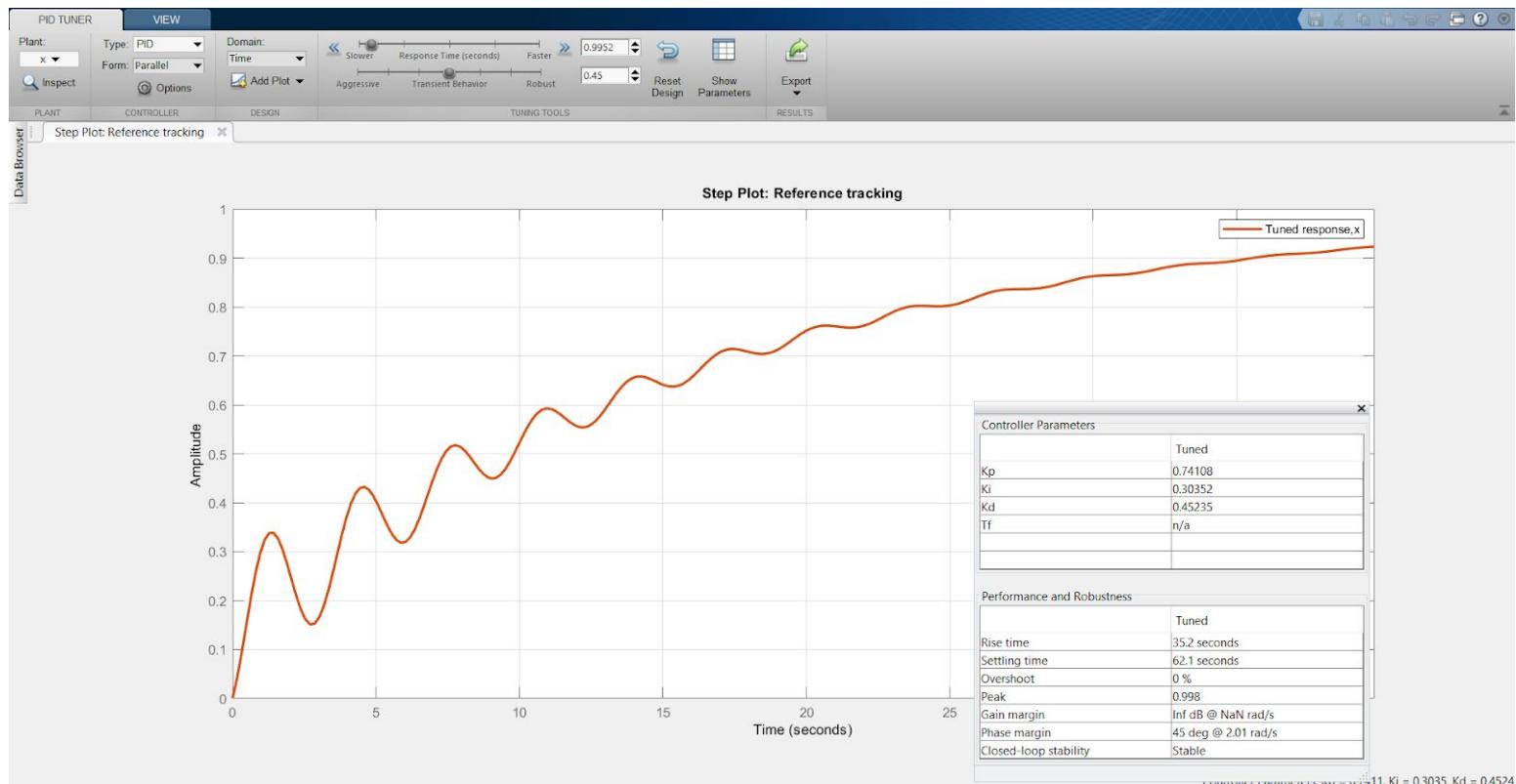


Plot of function without any control.

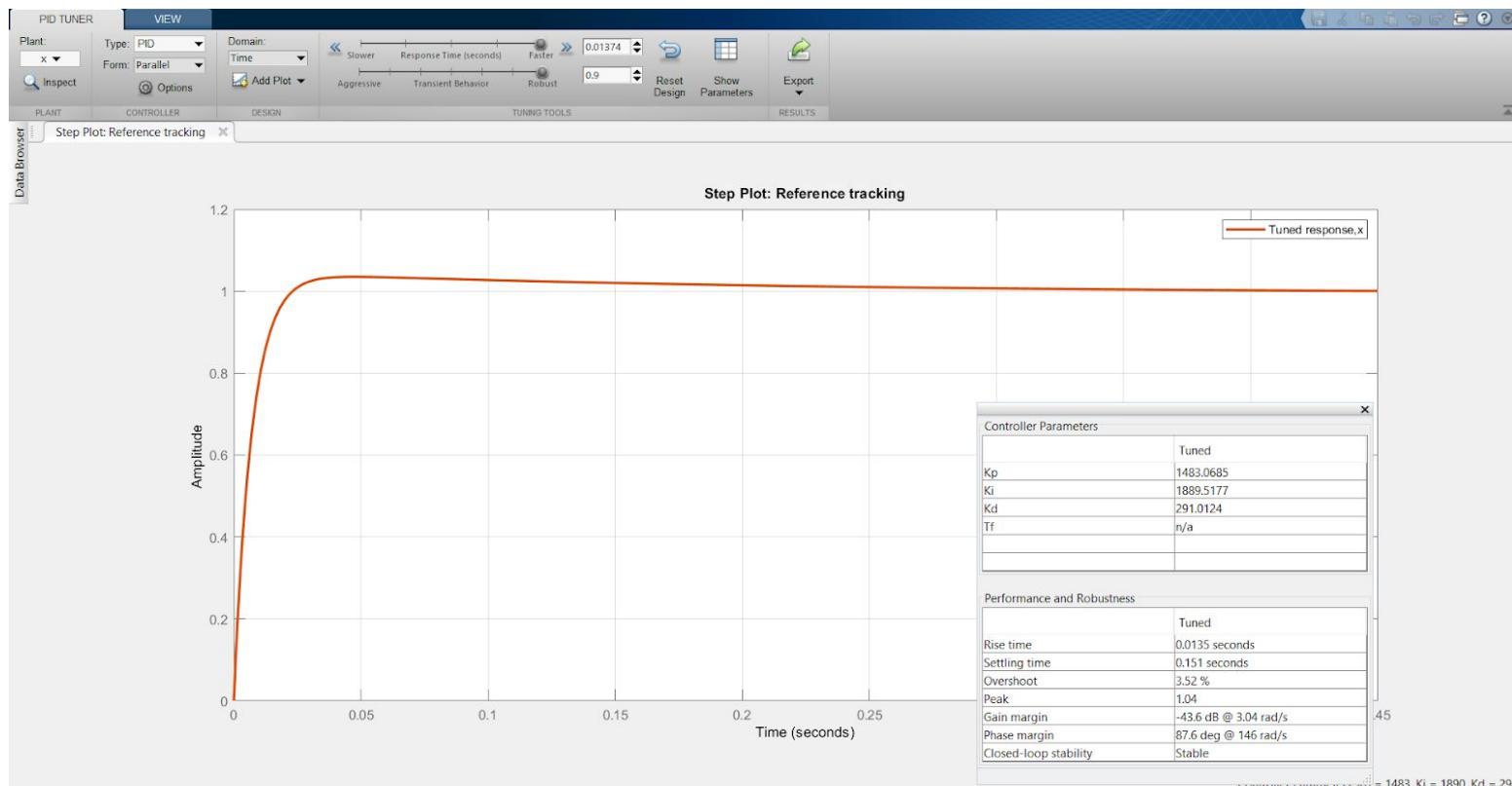
To have best rise time I set fastest response time in pidTuner and smallest transient behavior.



For smallest overshoot I just decreased response time.



For best rise time, overshoot and steady-state error I chose fastest response time and robust transient behavior.



As a result, I see this correlation:

Effects of increasing a parameter independently [22][23]

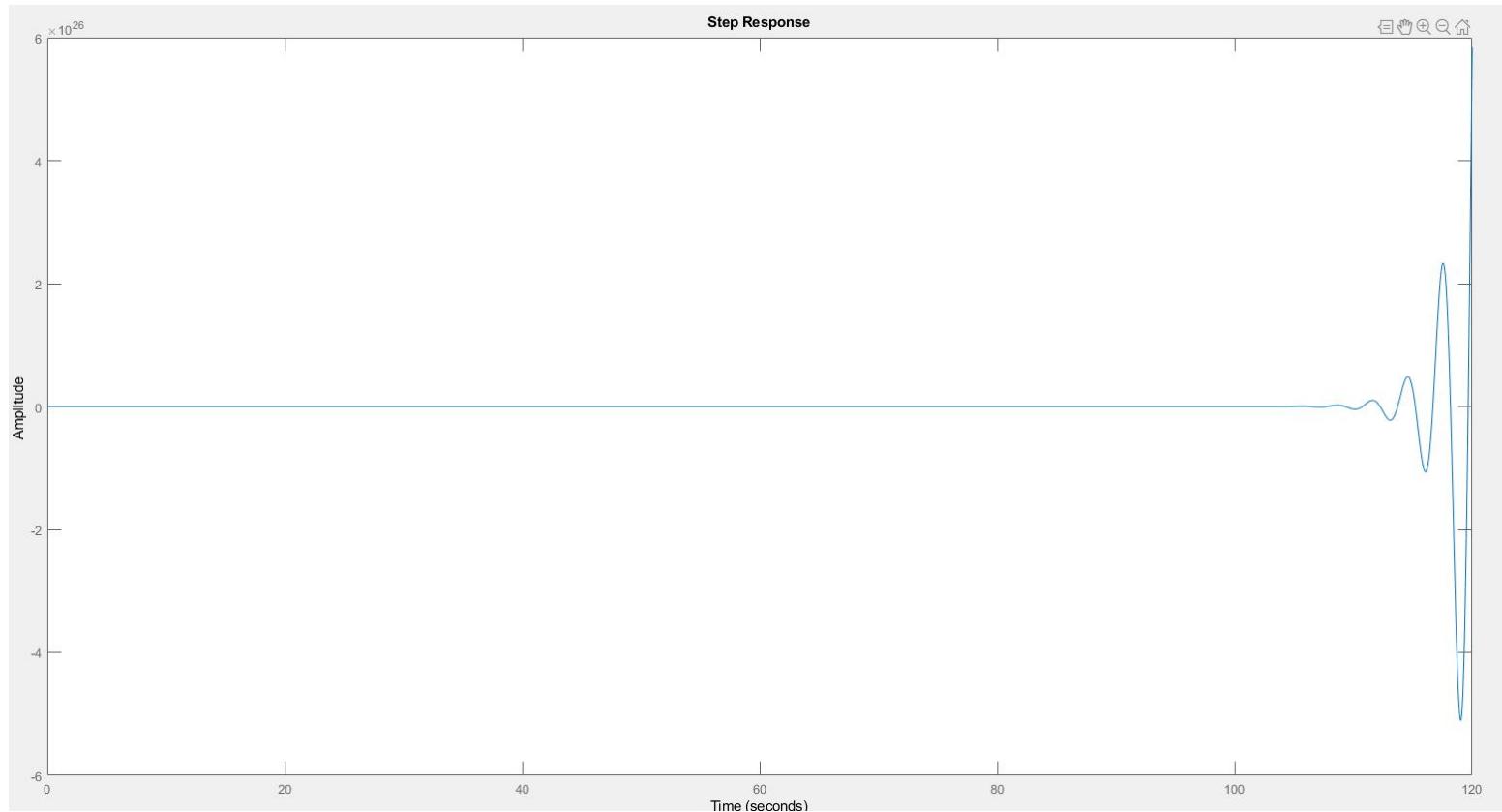
| Parameter | Rise time | Overshoot | Settling time | Steady-state error | Stability |
|-----------|--------------|-----------|---------------|---------------------|------------------------|
| K_p | Decrease | Increase | Small change | Decrease | Degrade |
| K_i | Decrease | Increase | Increase | Eliminate | Degrade |
| K_d | Minor change | Decrease | Decrease | No effect in theory | Improve if K_d small |

No4

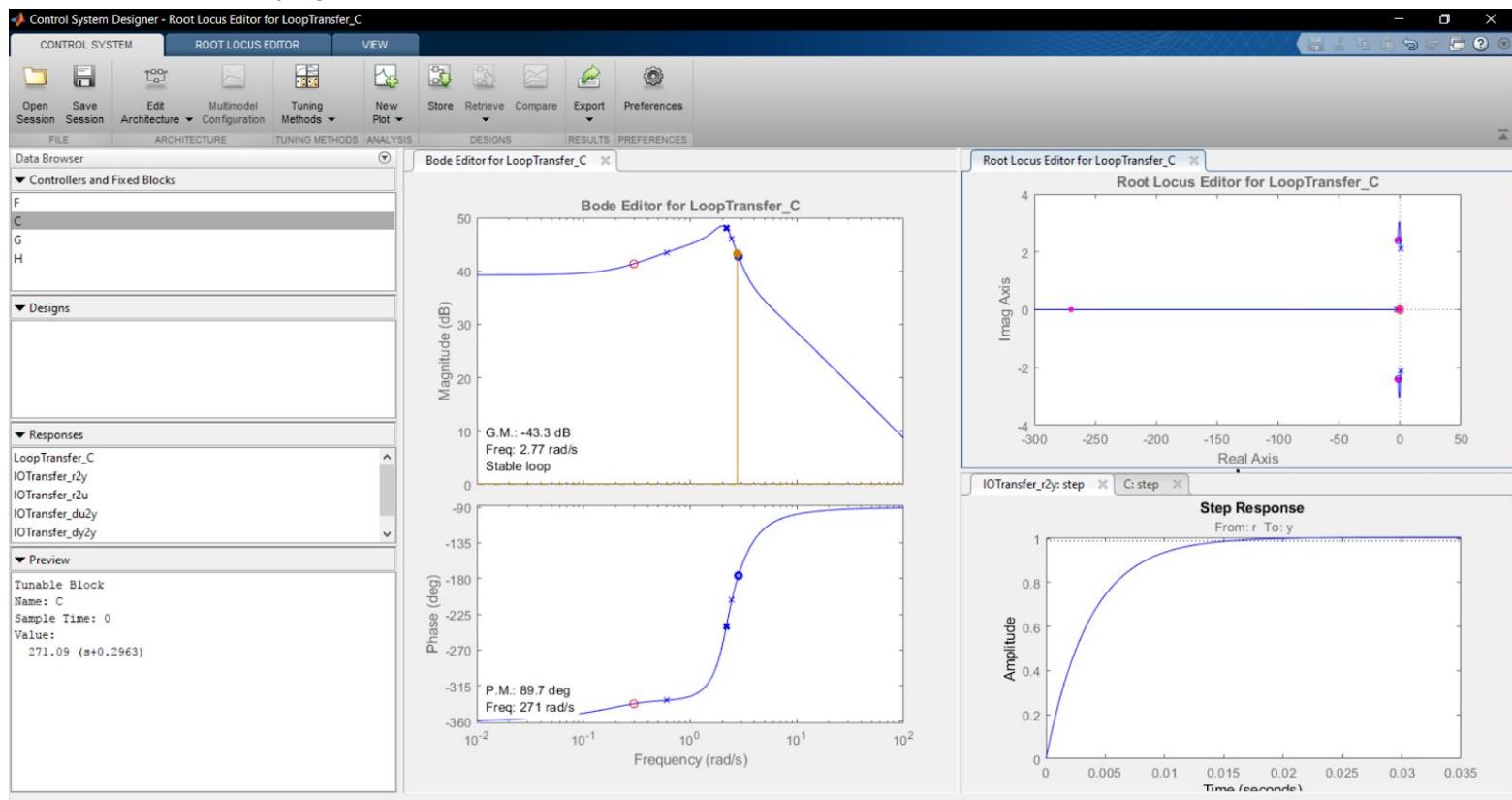
Design a lag or lead compensator (if applicable), play with zero and pole to find optimal values (of overshoot, peak time, transient process time, stationary error, etc.) for transient process. Use editors in Matlab Control System Designer.

$$(f) \quad W(s) = \frac{s^2 + 3s + 8}{s^4 + 2s^3 + 3s^2 + 13s + 7}$$

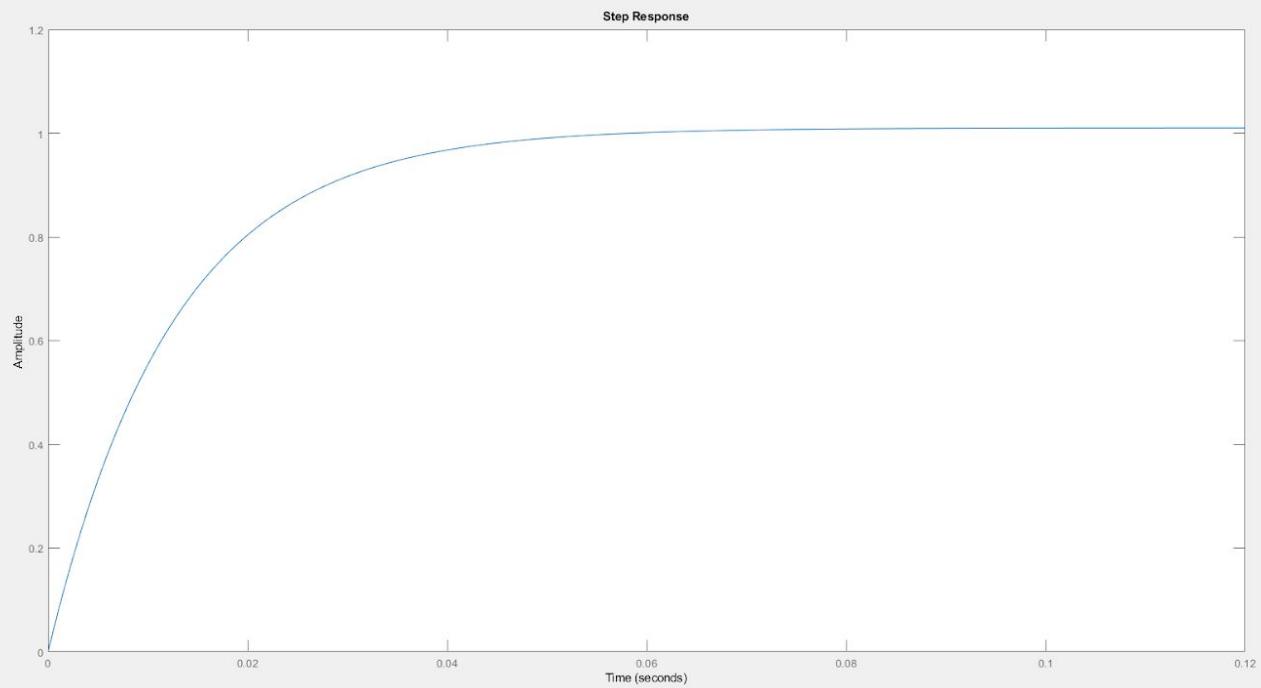
Plot without controller:



After playing with sisotools I found best coefficients for overshoot and peak time.



Plot with controller:



Matlab code:

```
1 - num = [1, 3, 8];
2 - denum = [1, 2, 3, 13, 7];
3 - G = tf(num, denum);
4
5 - z = -0.29628;
6 - k = 80.317;
7 - C = tf(k *[1 z], [1]);
8 - T = feedback(C*G, 1);
9 - step(T);
10
```