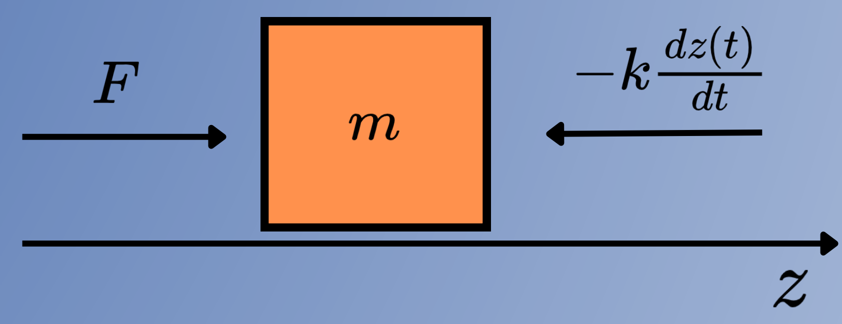
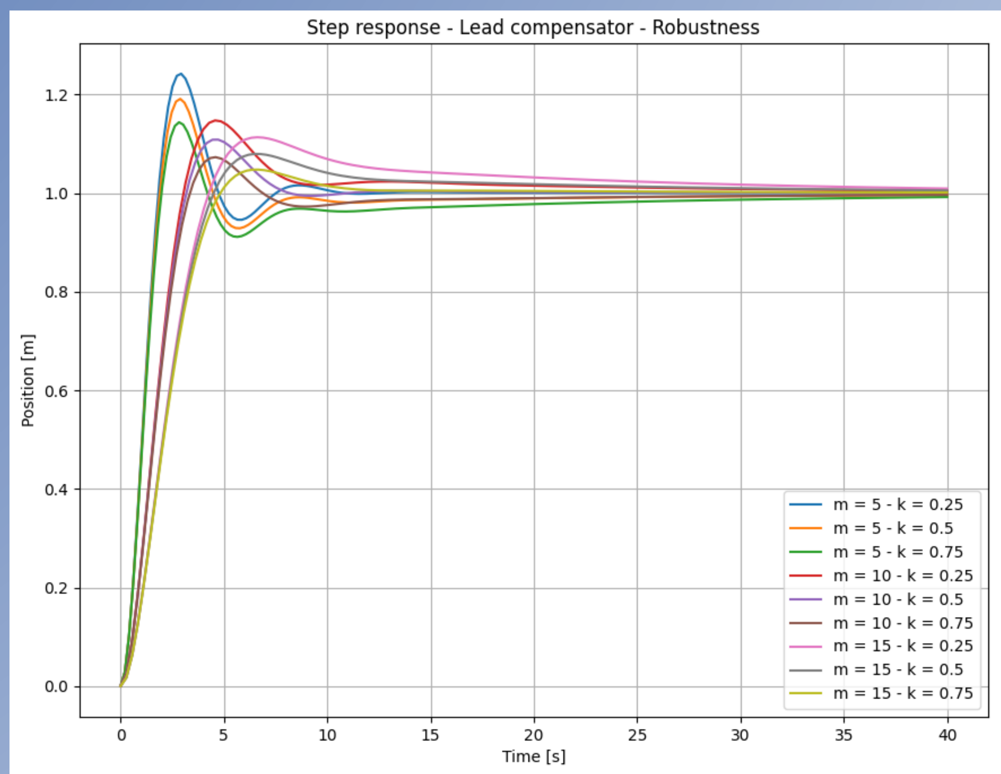
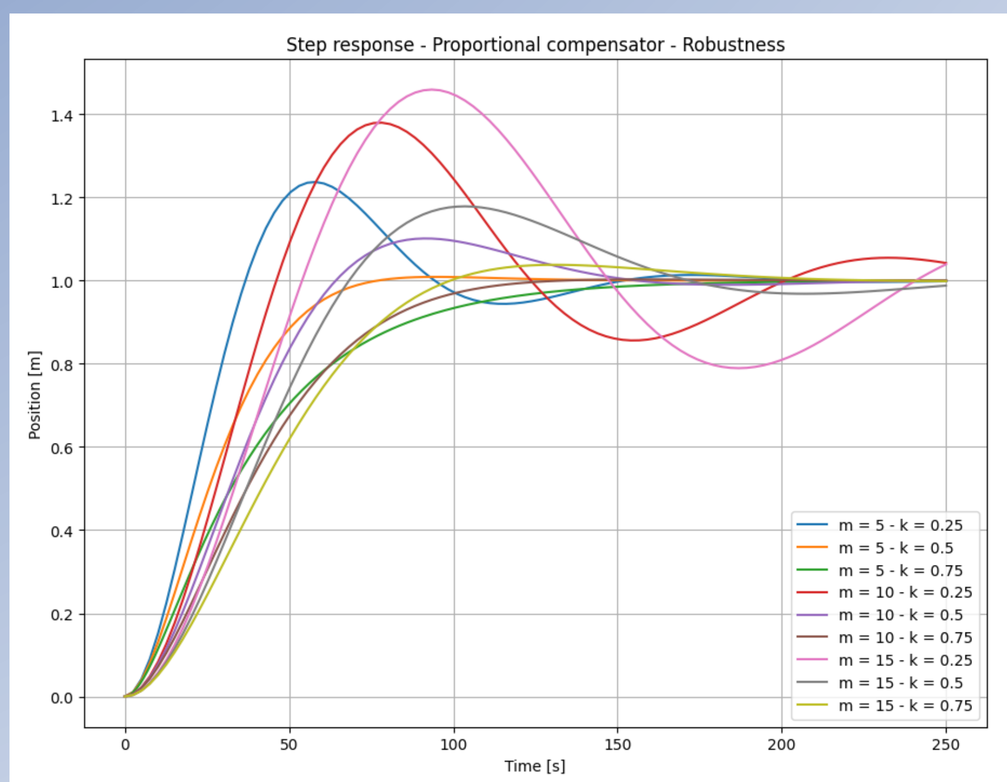
Lead Compensator

```python
import control as ct
import numpy as np
import math
import matplotlib.pyplot as plt

s = ct.TransferFunction.s

# System parameters
m = 10
k = 0.5

# System transfer function
P = 1/(s*(m*s+k))

# Proportional compensator parameters
kp = 0.018

# Proportional compensator transfer function
C_P = kp

# Close-loop transfer function
G_P = ct.feedback(C_P*P, 1)

# Plot root-locus - Proportional compensator
plt.figure()
ct.root_locus(C_P*P)

# Plot Bode diagram
plt.figure()
ct.bode(C_P*P)

# Calculate the gain and phase margins - Proportional compensator
gm, pm, gc, pc = ct.margin(C_P*P)
gm_dB = 20*np.log10(gm)

# Print margins
print(f"Gain Margin: {gm_dB:.3g} dB at frequency {gc:.3g} rad/sec")
print(f"Phase Margin: {pm:.3g} deg at frequency {pc:.3g} rad/sec")
print(f"Delay Margin: {((pm*math.pi/180)/pc):.3g} seconds")

# Ideal step response - Proportional compensator
t_P, yout_P = ct.step_response(G_P, T=250)

# Lead compensator parameters
kc = 0.4
tau_p = 1
tau_z = 18

# Lead compensator transfer function
C_L = kc*(tau_z*s+1)/(tau_p*s+1)

# Close-loop transfer function
G_L = ct.feedback(C_L*P, 1)

# Plot root-locus - Lead compensator
plt.figure()
ct.root_locus(C_L*P)

# Plot Bode diagram
plt.figure()
ct.bode(C_L*P)

# Calculate the gain and phase margins - Lead compensator
gm, pm, gc, pc = ct.margin(C_L*P)
gm_dB = 20*np.log10(gm)

# Print margins
print(f"Gain Margin: {gm_dB:.3g} dB at frequency {gc:.3g} rad/sec")
print(f"Phase Margin: {pm:.3g} deg at frequency {pc:.3g} rad/sec")
print(f"Delay Margin: {((pm*math.pi/180)/pc):.3g} seconds")
```



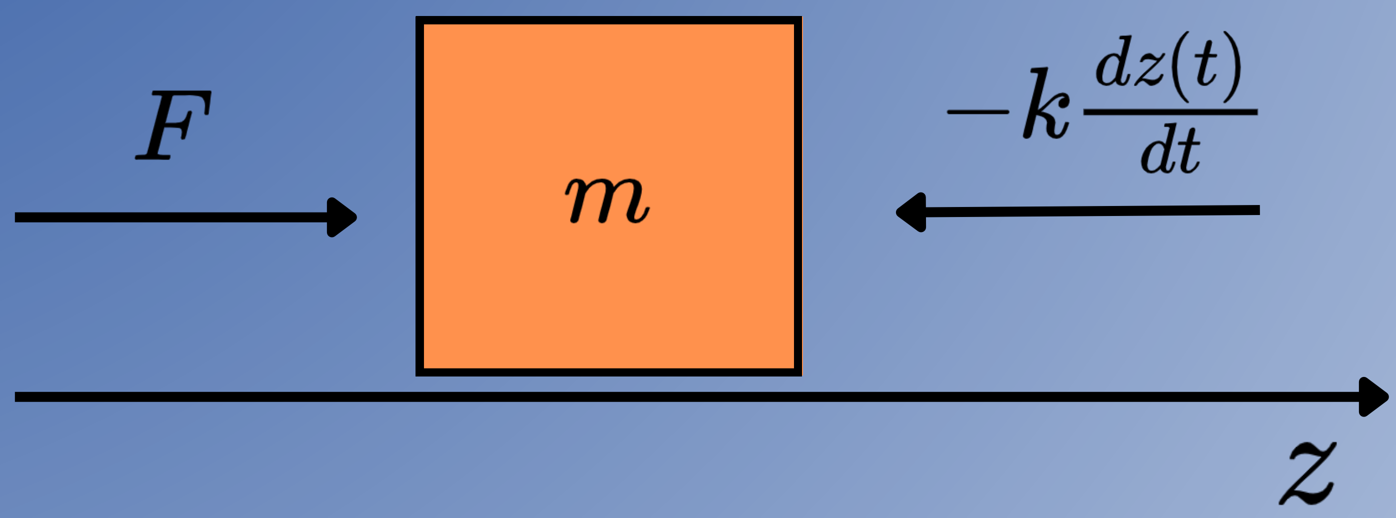




Simone Bertoni

Code
https://github.com/simorxb/Python-Lead-Compensator

Plant model



$$m\frac{d^2z(t)}{dt^2} = F - k\frac{dz(t)}{dt}$$

$$m = 10kg$$
$$k = 0.5\frac{Ns}{m}$$

$$P(s) = \frac{Z(s)}{F(s)} = \frac{1}{s(ms+k)}$$

Simone Bertoni

Objective

Design a feedback controller with maximum 10% overshoot.

We can find the damping that we need to achieve using the following formula:

$$\zeta = \frac{-\ln\left(\frac{PO}{100}\right)}{\sqrt{\pi^2+\ln^2\left(\frac{PO}{100}\right)}} = 0.59$$

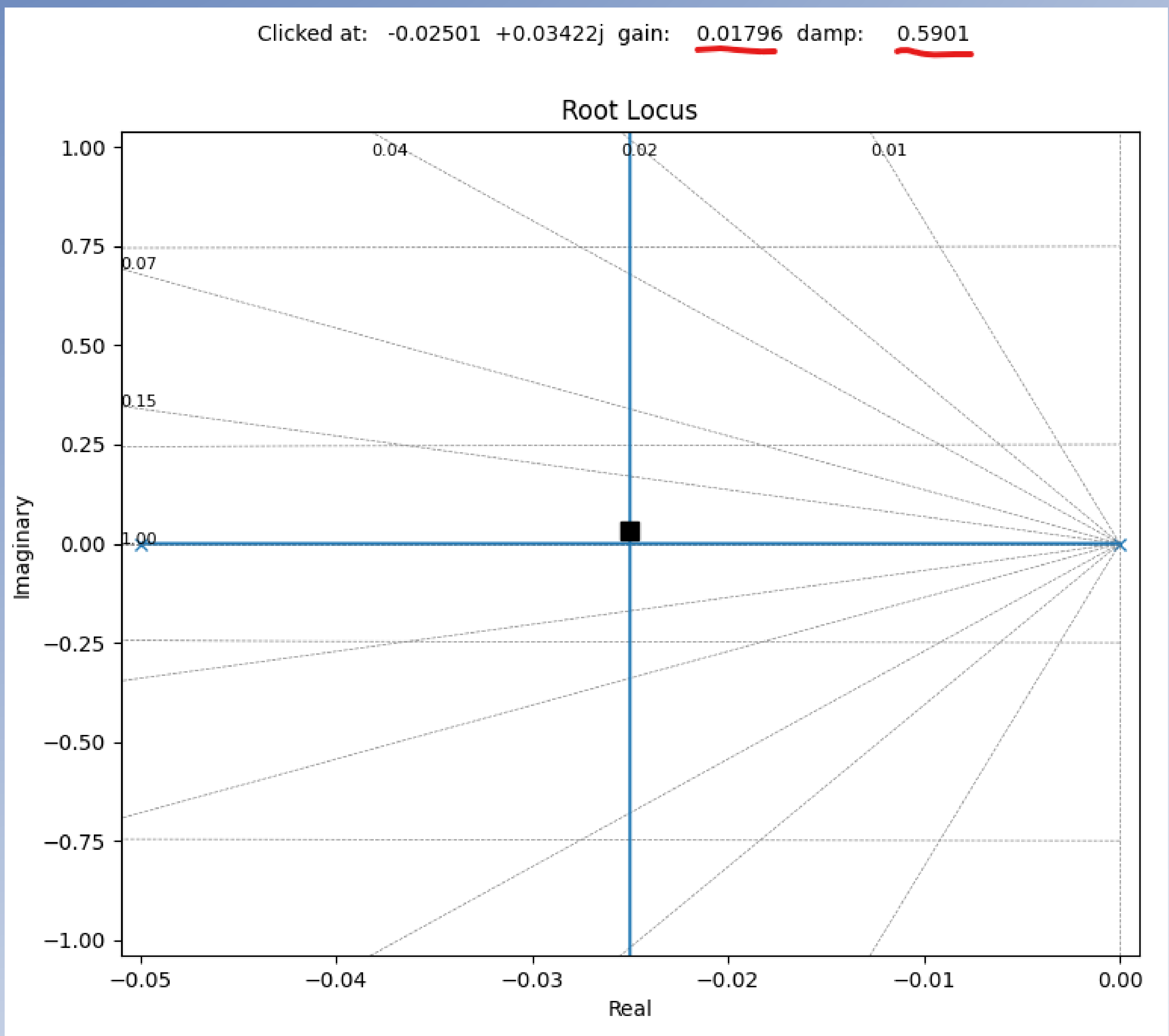
Then we can use the root locus to select the gain that we can afford not to exceed 10% overshoot.

Proportional compensator

Consider a proportional compensator:

$$C_P(s) = k_p$$

From the root locus we can see that $k_p = 0.018$ leads to $\zeta = 0.59$.



Simone Bertoni

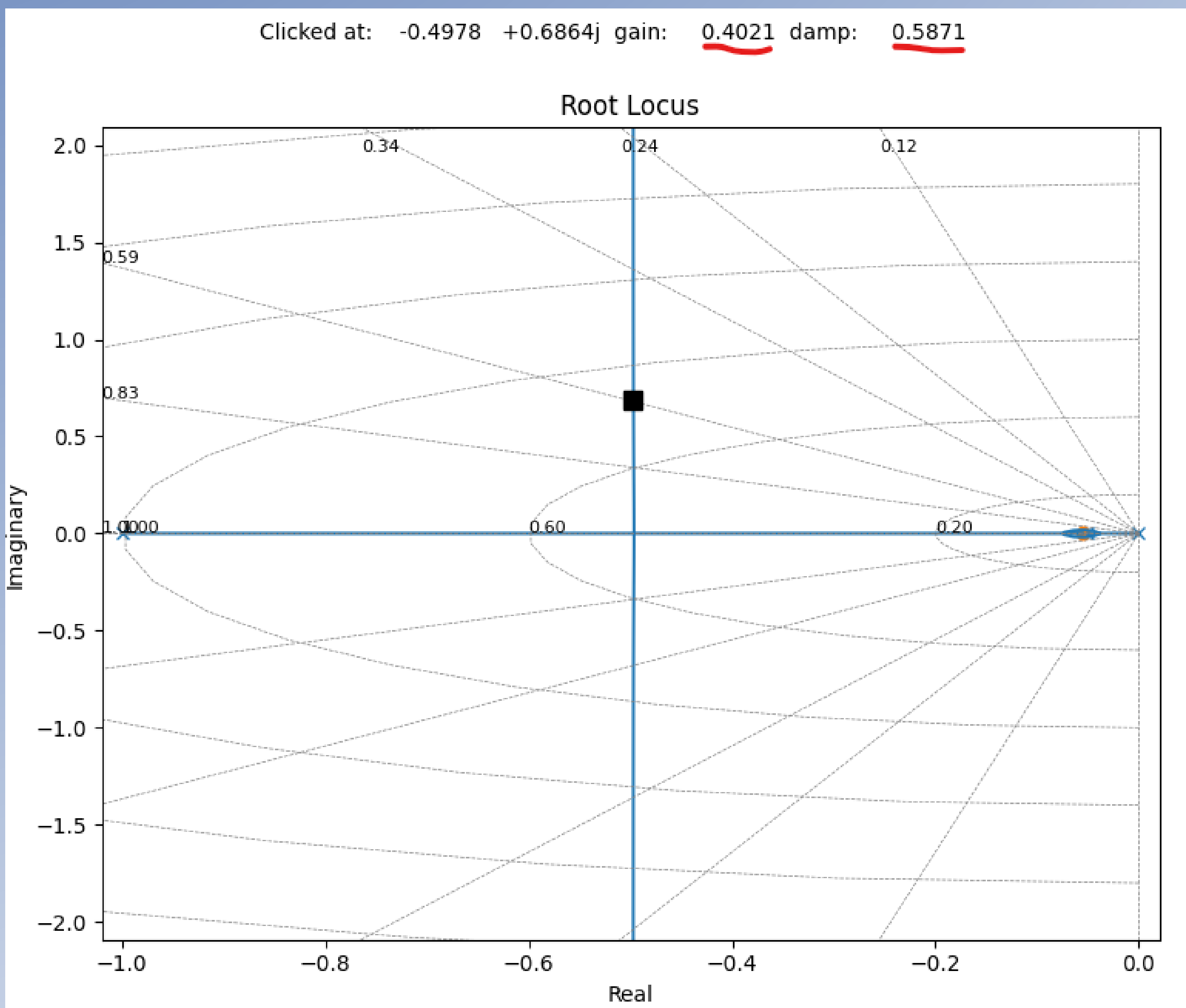
Lead compensator

Consider a lead compensator:

$$C_L(s) = k_l \frac{\tau_z s+1}{\tau_p s+1}$$

The poles of the plant are in 0 and 0.05. We choose the zero of the lead controller close to the fastest pole ($\tau_z = 18$) and the pole further to "attract" the close-loop poles ($\tau_p = 1$).

From the root locus we can see that $k_l = 0.4$ leads to $\zeta = 0.59$.



Simone Bertoni

Stability margins

Proportional compensator

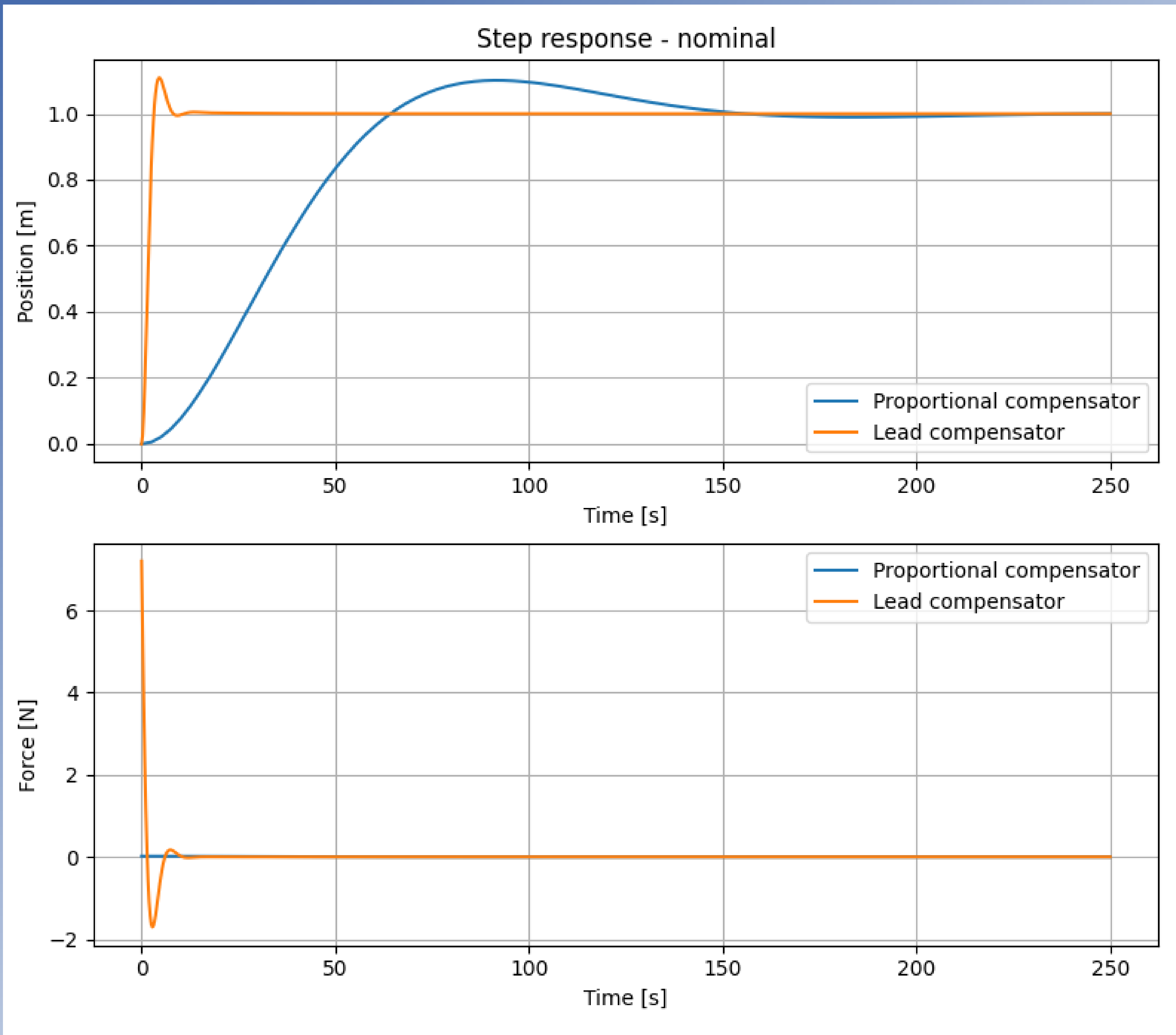
Gain Margin: inf
Phase Margin: 58.5 deg at frequency 0.0307 rad/sec
Delay Margin: 33.3 seconds

Lead compensator

Gain Margin: inf
Phase Margin: 57.9 deg at frequency 0.614 rad/sec
Delay Margin: 1.65 seconds

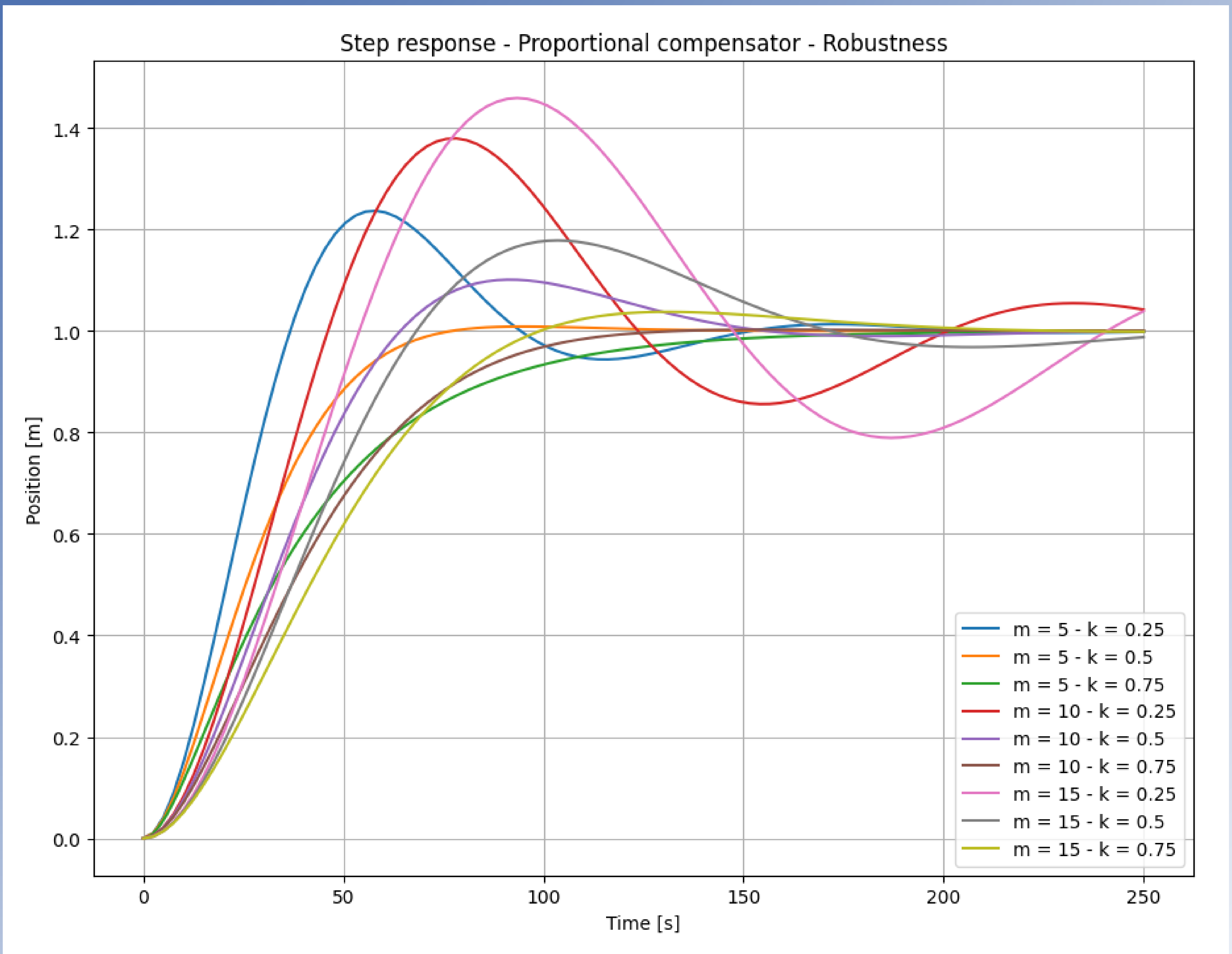
Simone Bertoni

Nominal step response



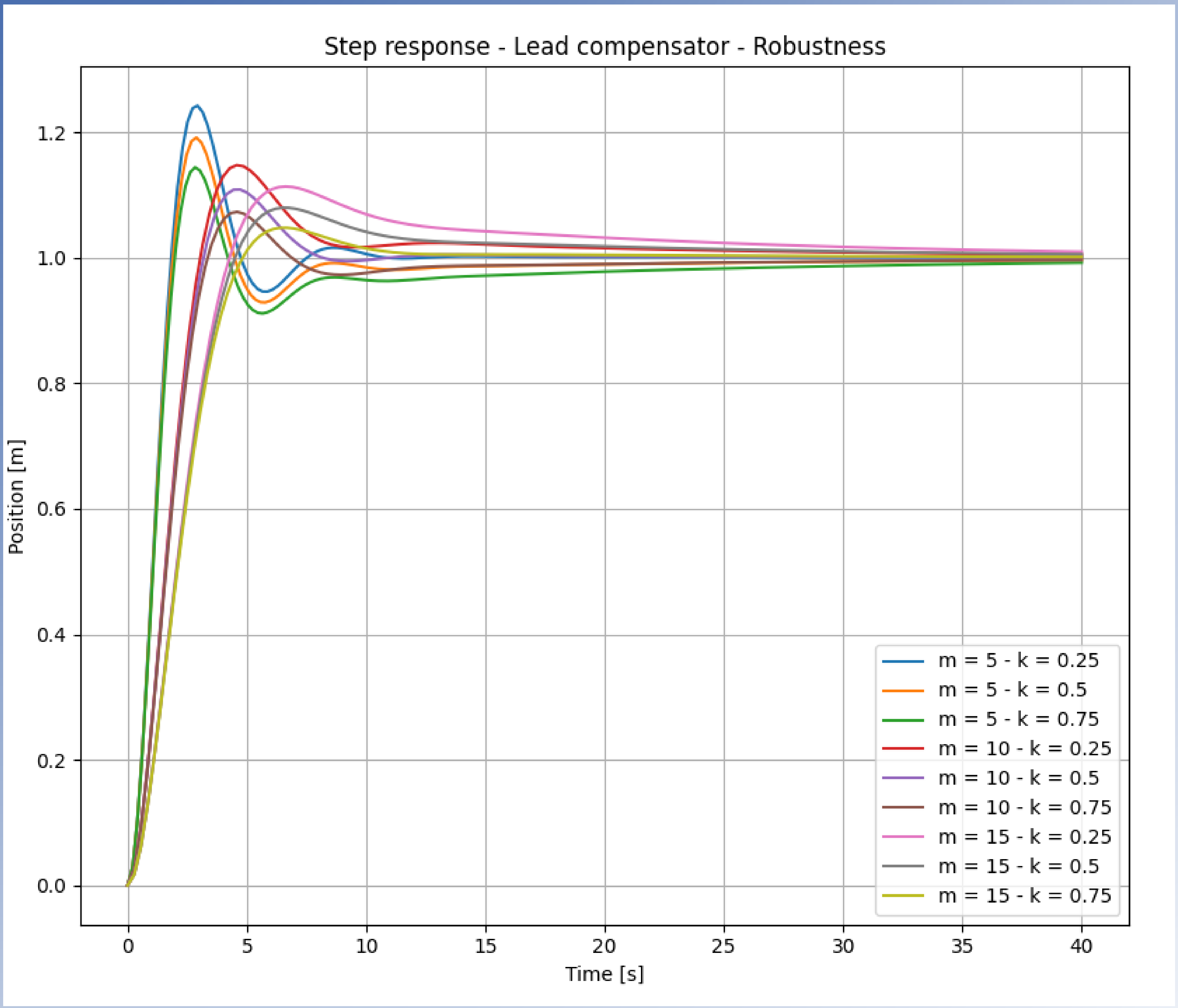
Simone Bertoni

Proportional compensator – Robustness



Simone Bertoni

Lead compensator – Robustness

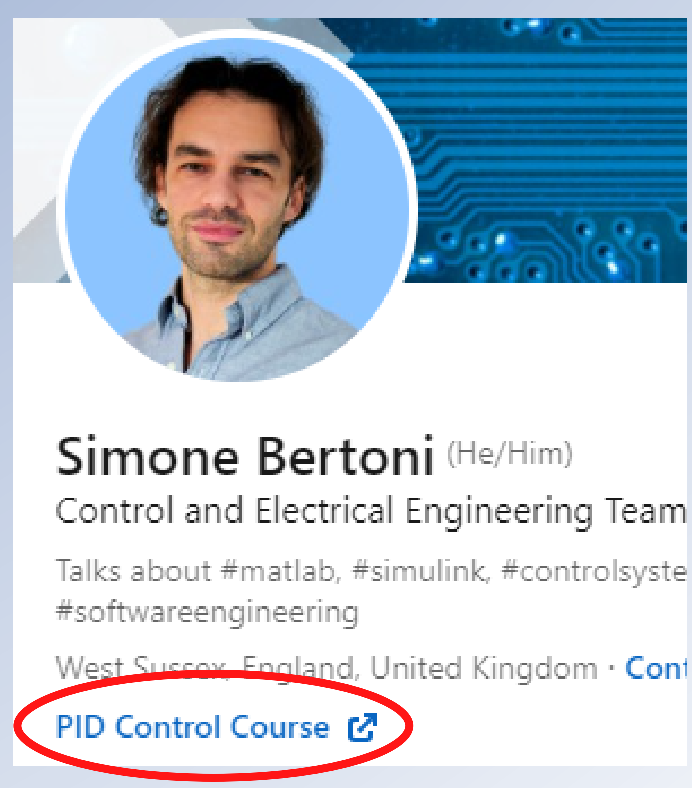


Simone Bertoni

PID Control

Interested in PID Control? Check out my digital course:

Find the link here!



Simone Bertoni