

Auteur: Romeo Permentier; Jerry Xiong  
Vak: Automatisering  
Onderwerp: labo 1  
Data: 22/04/2021;

**KU LEUVEN**

Inhoud: 7/10P => 3,5/5

## Inleiding

In dit labo worden verschillende vragen opgelost aan de hand van de control library in Python. Dit is een gratis alternatief voor Matlab. De transferfunctie waarop alle oefeningen op voort bouwen is:  
*Equation 1*

$$TF = G(s) = \frac{1}{(s + 0,3)(s + 3)(s + 6)}$$

In de code is ervoor gekozen om aparte functies te maken van elke vraag. op deze manier wordt de code modulair. Aangezien vraag 4 en vraag 5 nagenoeg hetzelfde doen, is ook hier voor het gemeenschappelijke deel een functie 'pi\_pid' gemaakt.

De *output* van de code wordt per vraag vermeld.

## Vraag 1

2/2P

----- Question 1 -----  
*Poles of the system are [-6. -3. -0.3], and zeroes are []  
Since [-6. -3. -0.3] are the poles, the system is stable  
The end value of the OL step\_response is 0.18496859432531995*

In de eerste vraag onderzoeken we de polen en nullen van het gegeven systeem, alsook de eindwaarde van het OL stapantwoord. Door middel van deze factoren kunnen we bepalen of het systeem al dan niet stabiel is.

**# Gebruikte functie uit de control library voor deze vraag:**

```
control.pole(Gpr)  
control.zero(Gpr)  
control.step_response(OLTF)
```



In Figure 1 kan men het pollen-nullen diagram zien van *Equation 1*. We zien duidelijk dat er geen polen of nullen in het rechter halfvlak, of op de imaginaire as (exclusief oorsprong) liggen. Het systeem is dus stabiel. In de console output ziet men de waarden van de polen. Het systeem kent geen nullen.

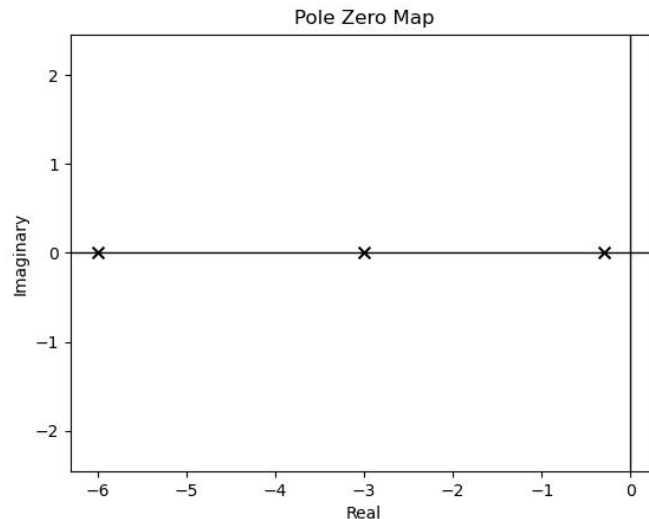


Figure 1 - Pole Zero map van G(s)

Figure 2 toont het gevraagde antwoord van het systeem op een stapfunctie aan de ingang. De functie 'control.step\_response()' geeft een tuple (x,y) terug die de x en y de coördinaten van het stapantwoord bevatten. De eindwaarde kan dus makkelijk teruggevonden worden door de laatste waarde uit de array van y te halen. Deze bedraagt 0,185 afgerond. In Figure 3 is er ingezoomd op de eindwaarde van de responsie. Hier kan men duidelijk zien dat de asymptoot op 0,185 ligt. Ook hier zien we dat het systeem stabiel is, het nadert naar een constante waarde op oneindig.

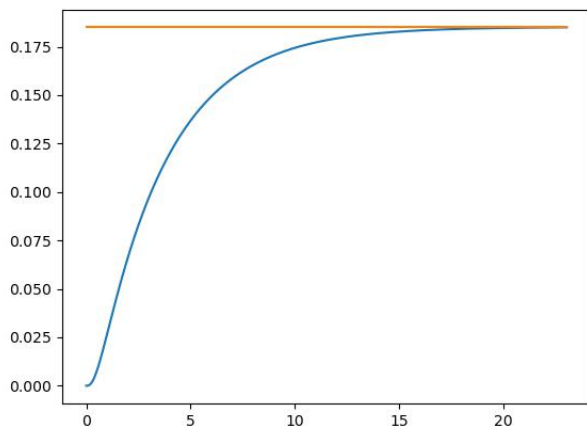


Figure 2 - Stapantwoord

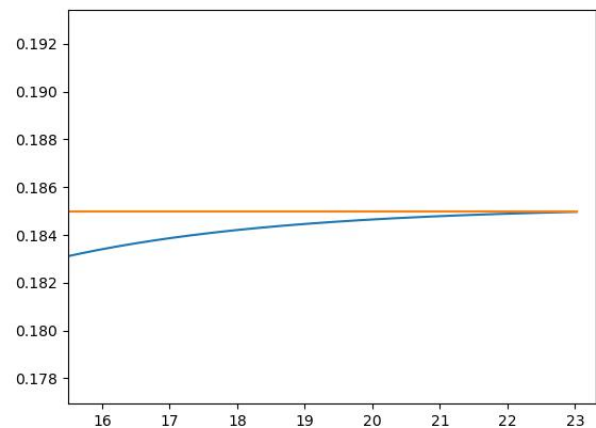


Figure 3 - Stapantwoord ingezoomd op eindwaarde

## Vraag 2

2/2P

----- Question 2 -----  
 The maximum gain in Hz is (187.10999999999999, inf, 0.9846586606431792, 4.54972526643093, nan, 1.6922205114360809) and in dB is [45.44193998 inf -0.1342859 13.15970345 nan 4.5691391 ]

In de tweede opgave wordt gevraagd te onderzoeken wat de maximale waarde voor een proportionele versterking is, opdat een marginaal stabiele regelkring bekomen wordt. Indien een systeem te veel versterkt wordt, dreigt het instabiel te worden.

Met behulp van de bode plot kunnen we bepalen hoeveel het systeem dus versterkt mag worden zonder dat het instabiel wordt.

**# Gebruikte functie uit de control library voor deze vraag:**

```
control.bode_plot(OLTF, dB=True, margins=True)
```

```
gainHz = control.stability_margins(OLTF)
```

```
gaindB = control.mag2db(gainHz)
```

Door de 'margins' parameter op "True" te zetten, zijn we in staat de maximale winst ~~eenvoudig~~ te berekenen en visualiseren. De maximale winst kan gevonden worden door het snijpunt van de fase grafiek met de  $-180^\circ$  lijn verticaal naar boven te gaan (verticale stippellijn op fase grafiek) tot men op de magnitude grafiek komt. De maximale winst is dus het verschil van de magnitude waarde en de horizontale lijn van 0 dB (aangeduid in het paars op Figure 4)

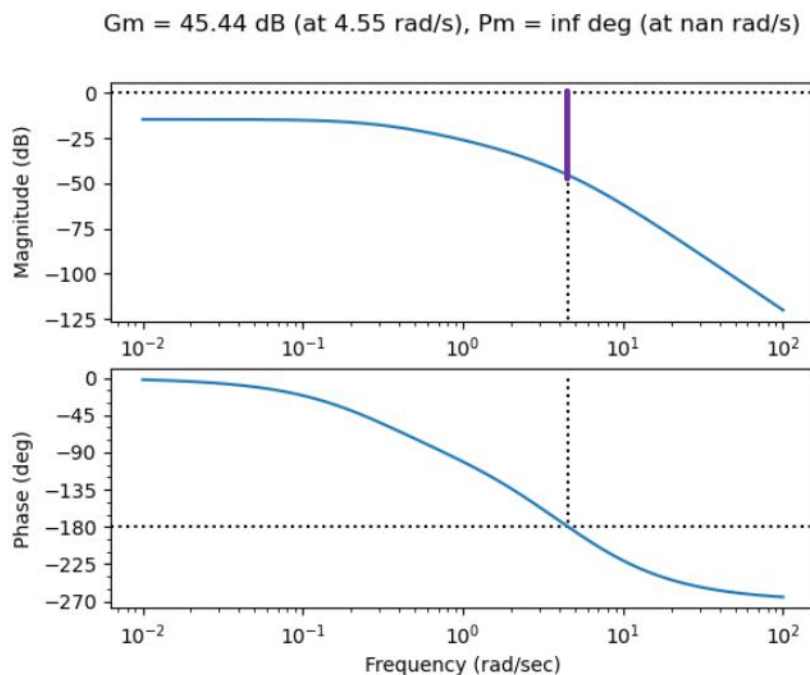


Figure 4 - Bode plot van de OLTF

Tenslotte bekomen we via de '`control.stability_margins(OLTF)`' de exacte waarde van de winst (gain). De maximale winst voor dit systeem is 187.11 Hz / 45.44dB (staat ook in Figure 4).

## Vraag 3

0,5/1P

----- Question 3 -----  
Kp is 187.10999999999999

Vraag drie heeft als doel het controleren van de maximale winst uit vraag 2. Dit kan gebeuren aan de hand van het Nyquist diagram voor  $K_p = 187,11$ . De nieuwe 'OLTF - Open Loop TransferFunctie' wordt dan:

Equation 2

$$TF = K_p * G(s) = \frac{187,11}{(s + 0,3)(s + 3)(s + 6)}$$

# Gebruikte functie uit de control library voor deze vraag:  
`control.nyquist_plot(OLTF, plot=True)`

Door het plotten van de Nyquist plot (Figure 5) zien we duidelijk dat de singulariteit zich bevindt op het coördinaat (-1,0). Dit ligt heel dicht bij het Nyquist-pad van de OLTF. Uit deze waarde kan besloten worden dat de waarde uit vraag 2 een logische waarde kan zijn. Het systeem verdraagt echter niet veel extra versterking zonder instabiel te worden.

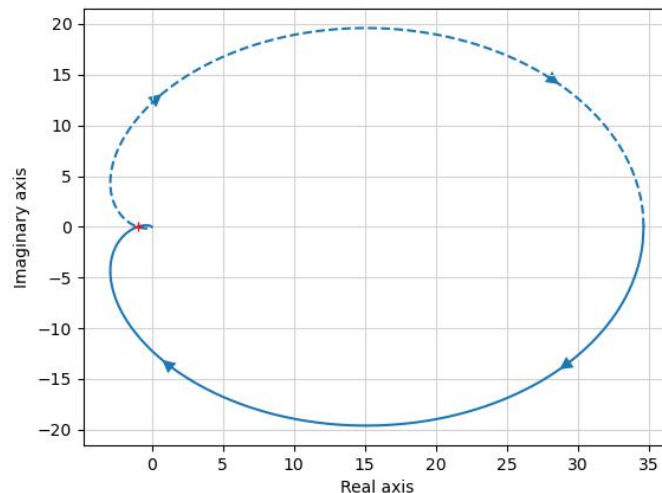


Figure 5 - Nyquist plot van de OLTF

## Vraag 4

1,5/2P

----- Question 4 -----

$$\frac{216.5 s + 43.3}{5 s}$$

```
{'RiseTime': 0.6374518928711776, 'SettlingTime': 5.868307131431724,
'SettlingMin': 0.8910308503418393, 'SettlingMax': 1.244562757226298,
'Overshoot': 24.456275722629805, 'Undershoot': 0, 'Peak': 1.244562757226298,
'PeakTime': 1.5186353918401587, 'SteadyStateValue': 1.0}
```

gevraagd  
methode  
resultaten  
interpretatie

In de vierde opgave wordt het ontwerp van een PI regelaar bekeken. De PI regelaar moet een fasemarge hebben van 45 graden. De 'rise time' en 'settling time' zullen gebruikt worden om deze regelaar te vergelijken met een PID regelaar in vraag 5. De vergelijking van de regelaar voor deze vraag is:

Equation 3

$$G_c = K_p * \left( \frac{1 + T_i s}{T_i s} \right)$$

# Gebruikte functie uit de control library voor deze vraag:

CLTF = control.feedback(OLTF)

t,c=control.step\_response(CLTF)

# Gebruikte functie uit de control,matlab library voor deze vraag:

control.matlab.stepinfo(CLTF)

De CLTF kan ~~makkelijk~~ berekend worden met de “control.feedback” functie uit de control library. Van deze CLTF bekijken we de stap responsie en onderzoeken we de verschillende tijden (waaronder rise- en settling-time).

Na trial-and-error werden volgende waarden gevonden om een fasemarge van 45° te bekomen:

Equation 4

$$K_p = 43,3$$

$$T_i = 5$$

Uit de stapresponsie in de Figure 6 lezen we volgende resultaten af: 0.637 sec (groen, rise time) en 0,891 sec (blauw, settling time). De rise time geeft aan hoelang het systeem nodig heeft om van 10% naar 90% van de steady state (= 1,0) waarde te gaan. De settling time geeft aan wanneer het systeem het moment bereikt waarop het signaal niet meer dan 2% verschilt van de steady state waarde.

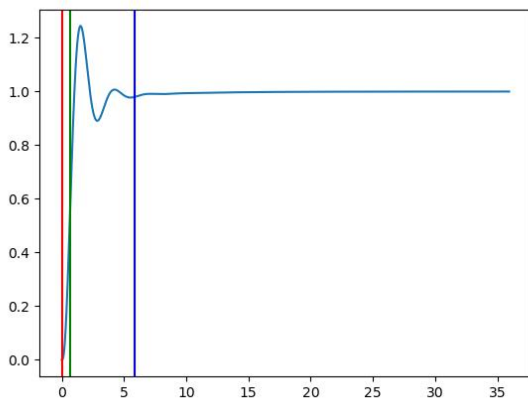


Figure 6 - Stapresponsie PI regelaar

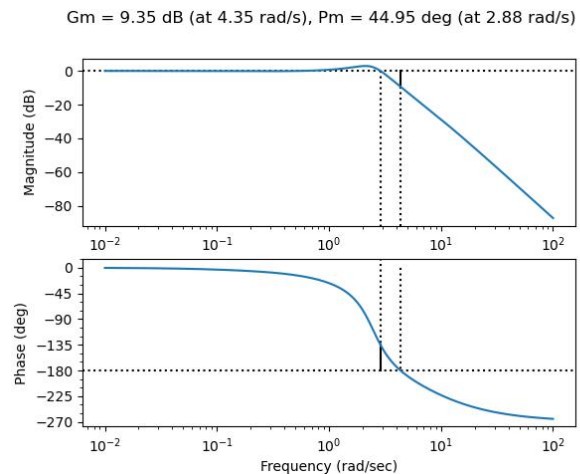


Figure 7 - Bode plot PI regelaar

## Vraag 5

1/3P

----- Question 5 -----

300.2 s + 5

1

```
{'RiseTime': 0.04259297048782012, 'SettlingTime': 118.96216657248159,
'SettlingMin': 0.48122525238089703, 'SettlingMax': 1.3295666152994254,
'Overshoot': 176.54985598228043, 'Undershoot': 0, 'Peak': 1.3295666152994254,
'PeakTime': 0.17037188195128047, 'SteadyStateValue': 0.48076923076923084}
```

De laatste vraag vraagt naar het ontwerp van een PID regelaar voor het systeem met dezelfde fasemarge van  $45^\circ$  als in vraag 4. Hierdoor zullen we de PID regelaar goed kunnen vergelijken met de PI regelaar uit vraag 4. De vergelijking van de regelaar voor deze vraag is:

Equation 5

$$G_c = K_p * \left( 1 + \frac{1}{T_i s} + T_d s \right)$$

Om een fasemarge van  $45^\circ$  te bekomen worden volgende waarden gebruikt:

Equation 6

$$K_p = 5$$

$$T_i = 30$$

$$T_d = 60$$

In Figure 8 vinden we de stapresponsie terug met de 'rise time' en 'settling time' respectievelijk 0.0426 sec (groen) en 118.962 sec (blauw). Omdat de PID regelaar een zeer kleine 'rise time' kent, is er in Figure 9 ingezoomd op de stijgende flank.

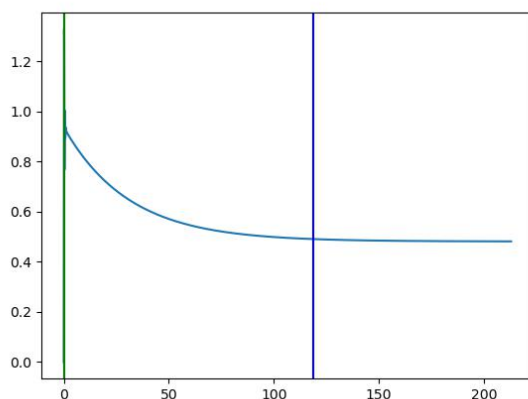


Figure 8 - Stapresponsie PID

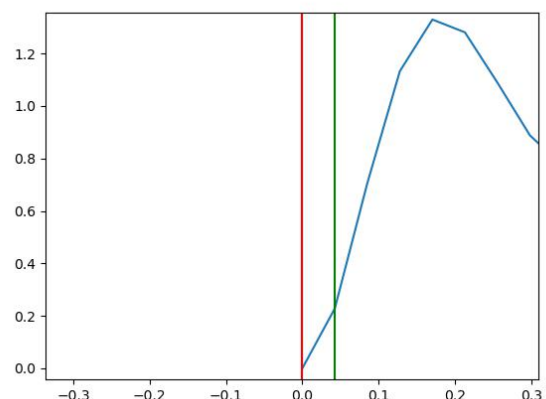


Figure 9 - Stapresponsie PID ingezoomd

Gm = inf dB (at nan rad/s), Pm = 45.42 deg (at 23.05 rad/s)

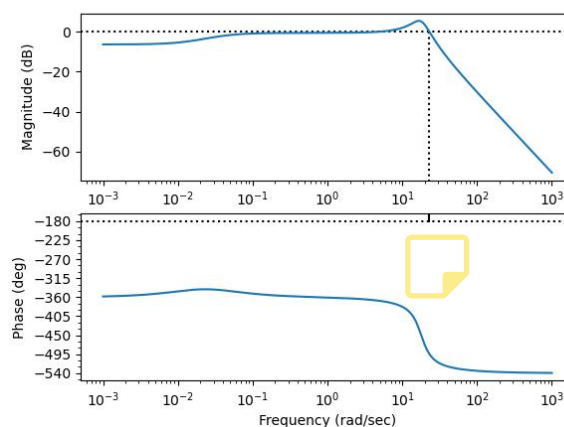



Figure 10 - Bode plot PID

## Besluit vraag 4 en vraag 5

Zowel de PI regelaar uit vraag 4 als de PID regelaar uit vraag 5 hebben hun eigen voor- en nadelen. De PI regelaar is stabiel dan de PID en heeft geen steady state error. We kunnen de PI regelaar echter niet gebruiken voor traag bewegende procesvariabelen.

De PID regelaar is minder stabiel dan de PI regelaar maar kan ook gebruikt worden voor snelle procesvariabelen. Nadelig is dan weer de verhoogde complexiteit door een verhoging van het aantal parameters ten opzichte van de PI regelaar.

Men kan besluiten dat de PID regelaar voor een  wijd gebruik kan ingezet worden, en daar waar mogelijk kan men de eenvoudigere PI regelaar gebruiken.

## Code

De code kan geraadpleegd worden op Github: <https://github.com/ro-per/EE-CRS-Lab1>