# 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* |

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

---------------------- 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 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 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.

|  |
| --- |
| Q1_-_PZ_map  Figure 1 - Pole Zero map van G(s) |

Figure 2 toont he gevraagde antwoord van het systeem op een stapfunctie aan de ingang. De functie *‘control.step\_response()’* geeft een tuple *(x,y)* terug die waarbij x en y de coordinaten 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.

|  |  |
| --- | --- |
| /home/roper/Programming/Python/EE-CRS_Spyder_Lab1/figures/Q1_-_Step_response.pngQ1_-_Step_response  Figure 2 - Stapantwoord | Q1_-_Step_response_zoom  Figure 3 - Stapantwoord ingezoomd op eindwaarde |

# Vraag 2

---------------------- 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 teveel 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’ paramater 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°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

---------------------- 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 . De nieuwe ‘OLTF - Open Loop TransferFunctie’ wordt dan:

|  |
| --- |
| *Equation 2* |

# 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 bevind op het coordinaat (-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.

|  |
| --- |
| Q3_-_Nyquist_plot  Figure 5 - Nyquist pplot van de OLTF |

# Vraag 4

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

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}

De vergelijking van de regelaar voor deze vraag is:

|  |
| --- |
| *Equation 3* |

# Vraag 5

---------------------- 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}

|  |
| --- |
| *Equation 4* |

De laatste vraag vraagt naar het ontwerp van een PID regelaar voor het systeem met dezelfde fasemarge van 45° 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:

Om een fasemarge van 45° te bekomen worden volgende waarden gebruitk:

|  |
| --- |
| *Equation 5* |

In de figuur vinden we de stapresponsie terug met de ‘rise time’ en ‘settling time’ respectievelijk 0 en 117 seconden.

|  |  |
| --- | --- |
|  |  |

# 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 regeleaar is stabieler dan de PID en heeft geen steady state eror. We kunnen de PI regelaar echter niet gebruiken voor traag bewegende procesvariabelen.

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