Practicum verslag
SeeSaw

Jan Dekock
Carl Vermoere
2de Ir D&A

12 maart 2001

1 State Space model

Om ons state space model te kunnen opstellen moeten we eerst de energie-vergelijkingen van het systeem afleiden.

1.1 kinetische energie

We beginnen met het simpelste: de kinetische energie van de seesaw.

$$E_{kinSS} = \frac{(J\dot{\theta})^2}{2}$$

Voor het autoke is het al iets complexer: hier moeten we de algemen formule voor kinetische energie toepassen, zijnde

$$E_{kin} = \frac{mv^2}{2}$$

Dus moeten we de snelheid van het autoke bepalen. Deze snelheid is een samengestelde snelheid van de snelheid van het autoke over de wip, samengesteld met de snelheid van het autoke ten gevolge van het kantellen van de wip. Na een beetje vectorieel rekenen komen we dan uit op

$$E_{kinAutoke} = \frac{m(\dot{x} + \dot{\theta}cos\theta)^2}{2} + \frac{m(\dot{\theta}dsin\theta)^2}{2}$$

1.2 potentiele energie

Voor het opstellen van de vergelijkingen voor de potentiele enegie nemen we het tafeloppervlak als 0-punt. De potentiele energie van het autoke is dan

$$E_{potautoke} = mg(hcos\theta - xsin\theta)$$

Voor de wip is dit dan

$$E_{potSS} = Mgccos\theta$$

Hieruit bepalen we dan L

$$L = E_{kin} - E_{pot}$$

Als we deze vergelijking dan afleiden volgens de formules op pagina 3 van de manual, hebben we de volgende vergelijkingen:

$$F = m\ddot{x} + mh\ddot{\theta} - mx\dot{\theta}^2 - mgsin\theta$$

$$0 = mh\ddot{x} + m(h^2 + x^2)\ddot{\theta} + 2mx\dot{\theta}\dot{x} + J\ddot{\theta} - mghsin\theta - mgxcos\theta - Mgcsin\theta$$

Nu moeten we dit nog lineariseren. Alle hogere graadsterm van de variabelen x en θ kunnen verwaarloosd worden, en sinθ =θ en cos$\theta = 1$.

toegepast geeft dit:

$$F = m\ddot{x} + mh\ddot{\theta} - mg\theta$$

$$0 = mh\ddot{x} + mh^2\ddot{\theta} + J\ddot{\theta} - mgh\theta - mgx - Mgc\theta$$

1.3 motormodel

Na een handvol substituties komen we heel snel tot de volgende uitdrukking voor F in functie van de gegeven parameters en state $\dot{x}$

$$F = \frac{K_g K_m}{r R_m}(V - \frac{K_b K_g \dot{x}}{r})$$

1.4 het eigenlijke gelineariseerde model

zoals opgedragen nemen we de volgende states: x, $\dot{x}, \theta, \dot{\theta}$ Na wat herschrijven van onze vergelijkingen bekomen we dan het volgende state space model:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{mgh}{J} & g - \frac{Mgch}{J} & -\frac{K_g^2 K_b K_m}{r^2 R_m}(\frac{h^2 m + J}{mJ}) & 0 \\ \frac{mg}{J} & \frac{Mgc}{J} & \frac{K_g K_b K_m h}{r^2 R_m J} & 0 \end{bmatrix} \quad (1)$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{K_g K_m}{r R_m}(\frac{mh^2+J}{mJ}) \\ -\frac{K_g K_m h}{r R_m J} \end{bmatrix} \tag{2}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{3}$$

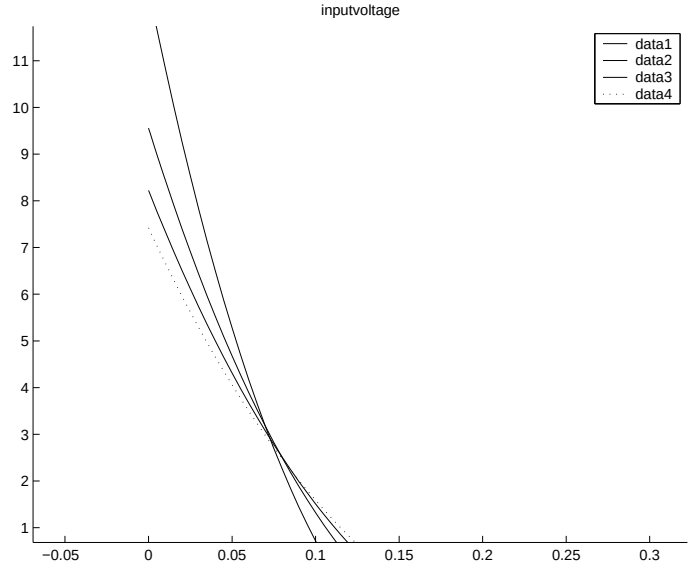
$$D = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4}$$

2 De controller

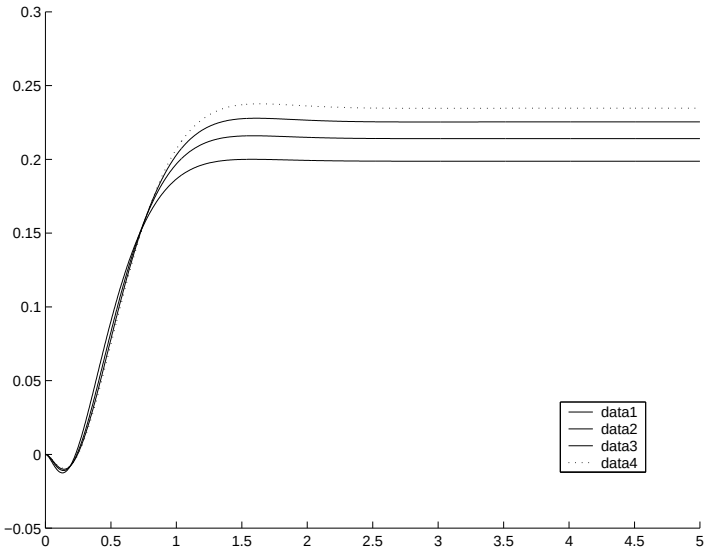
2.1 het vereenvoudigde model

In een eerste fase ontwerpen we een LQR controller voor het continue systeem. Er moet dus een terugkoppelmatrix gevonden worden, die uitgaande van de states het autoke aanstuurt. Voor deze optimisatie procedure hebben we 2 gewichtsmatrices nodig, zijnde Q en R.
De gewichten die we in Q steken hebben hun invloed op de states, de R matrix geeft gewichten mee aan de controller-output. Grote waarden in Q zullen er voor zorgen dat de states klein gehouden worden, een grote R zal er voor zorgen dat de controlleroutput, zijnde de spanning die aan het autoke wordt aangelegd, laag gehouden wordt. Om saturatie zo veel mogelijk te vermijden, zullen we dus hier een 'grote' R moeten gebruiken.
Als we aan Q_{11} een grote waarde geven, zullen we de verplaatsing van het autoke zo veel mogelijk beperken. Q_{22} groot maken zal tot gevolg hebben dat we de kantelhoek van de wip laag willen houden. De parameters $Q_{33} en Q_{44}$ houden de snelheiden van het autoke en de wip laag. Dit zou dan tot gevolg hebben dat het systeem traag zou reageren. Om een snel reagerend systeem te bekomen kunnen we deze parameters best laag houden, en dus gewoon nul laten. We kunnen wel eens experimenteren om de invloed te zien.
We beginnen met het afregelen van R. Uit de simulaties blijkt dat bij een stijgende R de steady-state fout groter wordt, maar dat de input-spanning kleiner wordt, en dus minder last gaat hebben van de saturatie. Op het ogenblik dat ik dit verslag aan het schrijven ben, heb ik de neiging om een regelaar te ontwerpen die een lage steady state fout heeft. De resultaten voor R gaande van 2 tot 8 is te zien in figuren 1 en 2 Voor onze regelaar hebben we een R gekozen van 7, om de regelaar zo weinig mogelijk in saturatie te laten gaan..

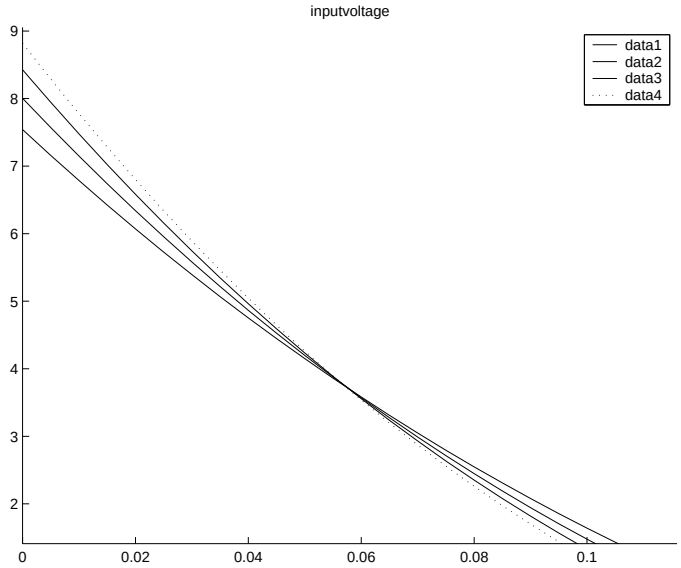
Nu kunnen we ook Q afregelen. Na wat geredeneer hebben we besloten om er voor te zorgen dat de seesaw zelf zo weining mogelijk zou schommelen, en dat



Figuur 1: spanningsverloop bij stap: boven saturatie, variabele R



Figuur 2: hoekverloop bij aanleggen stap, variabele R



Figuur 3: spanningsverloop bij stap: boven saturatie, variabele Q_{11}

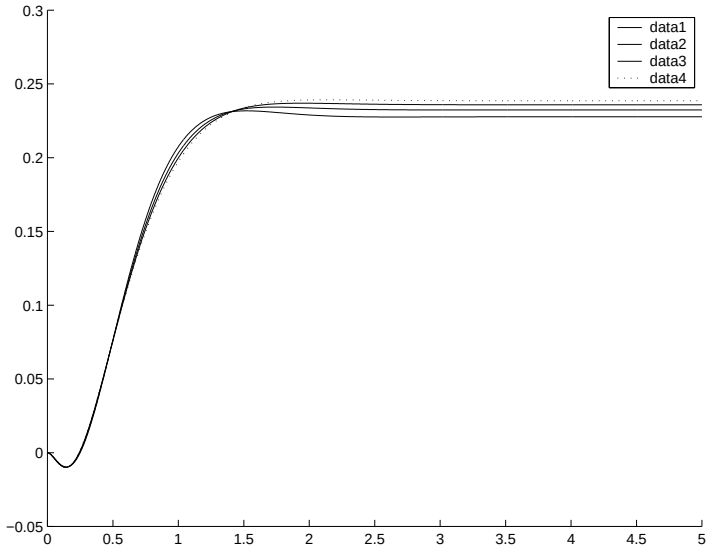
het autoke zelf veel vrijheid heeft. Wanneer we deze voor verschillendeparameters simuleren, krijgen we de volgende output: figuren 3 en 4 Wanneer we Q_{11} verhogen, krijgen we een grotere steady-state fout, en reageert het systeem ook trager. Het is dus duidelijk dat hier een lage waarde moeten nemen voor Q_{11}. Daarenboven zorgt een kleine Q_{11} voor minder saturatie. We nemen Q_{11}=1000. Voor Q_{22} ligt het iets moeilijker, hier hebben we te maken met tegenwerkende verschijnselen. Als we Q_{22} verhogen, krijgen we meer last van saturatie, maar wel een kleinere steady state fout. We kiezen hier dan ook best voor een middelmatige waarde voor Q_{22}, bijvoorbeeld 5000. Simulatieresultaten te zien in figuren 5 en 6 Om te controlleren of ons systeem stabiel is, kunnen we ook eens de polen van ons systeem{7,100,5000} plotten: figuur 7

Voor K hebben we dan de volgende waarden (in SI-eenheden)

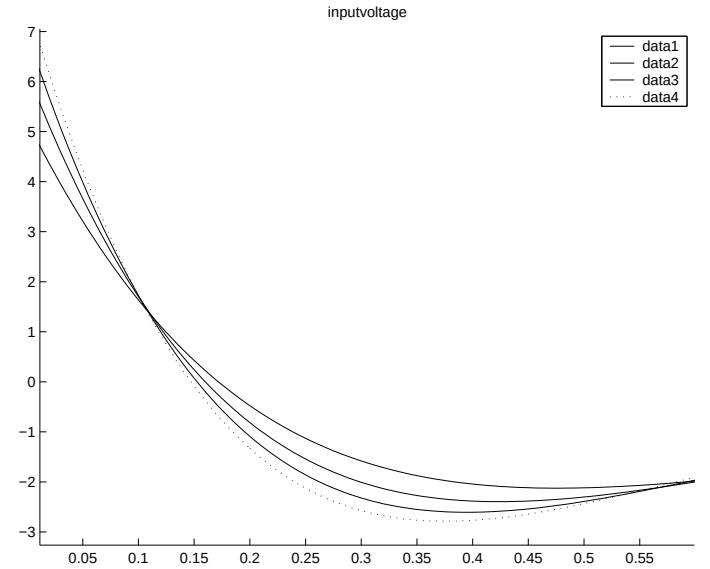
$$\begin{bmatrix} 63.7112 & 65.9926 & 5.9995 & 23.8778 \end{bmatrix}$$

2.2 het meer realistische model

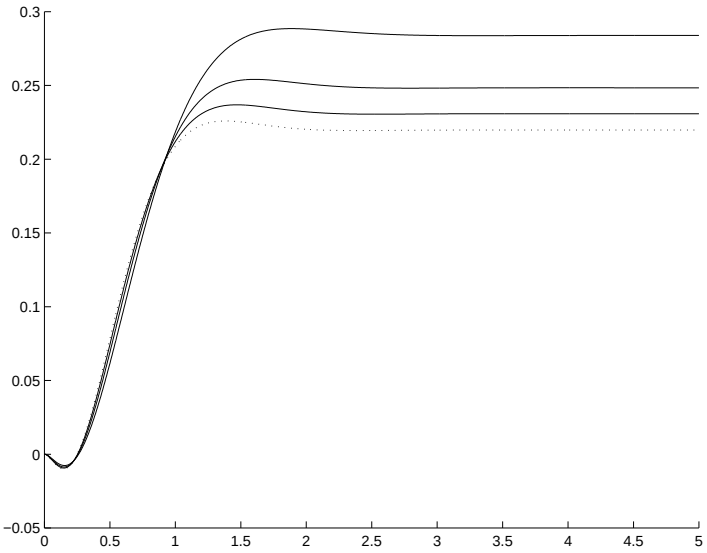
We kunnen ons model realistischer maken door de invloed van de computer en ruis mee in ons model op te nemen. De computer sampelt de inputs aan een frequentie van 200 Hz, en er wordt een laagdoolaatfilter gebruikt van 2Hz. De output van de controller is ook begrenst tussen [-5V 5V]. Het model is gesimuleerd geweest geworden voor de volgende parameters: R=7 Q=[1000 5000 0 0]
R=5 Q=[2000 5000 0 0]



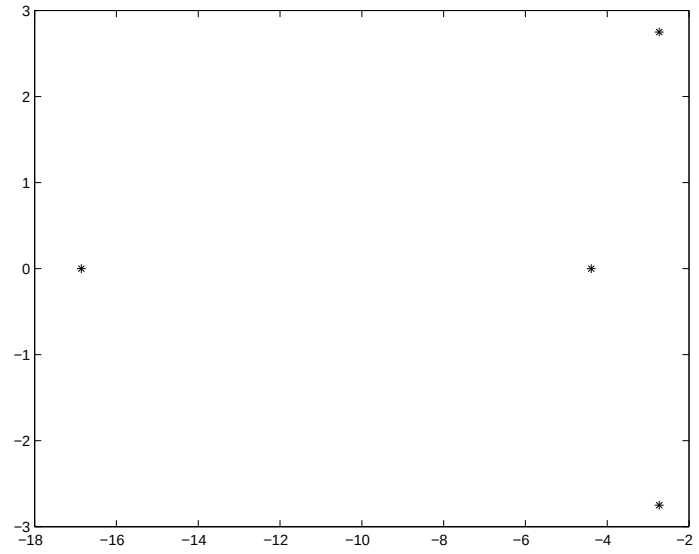
Figuur 4: hoekverloop bij aanleggen stap, variable Q_{11}



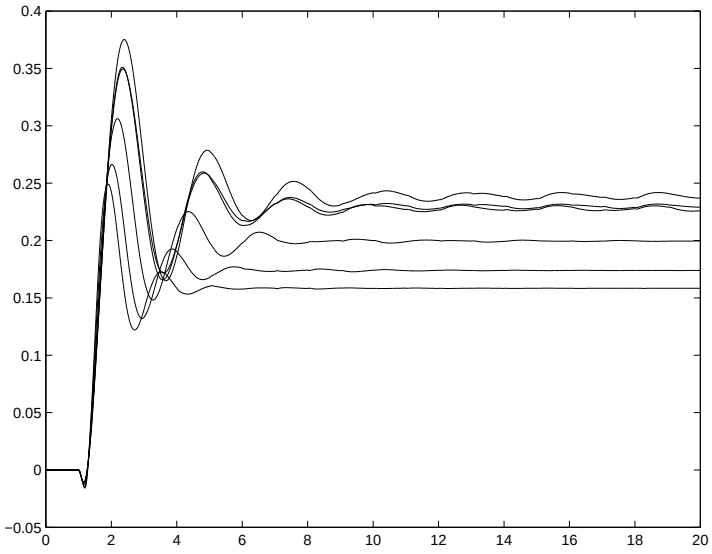
Figuur 5: spanningsverloop bij stap: boven saturatie, variabele Q_{22}



Figuur 6: hoekverloop bij aanleggen stap, variable Q_2



Figuur 7: polen-plot voor R=7 en Q=[1000 5000 0 0]



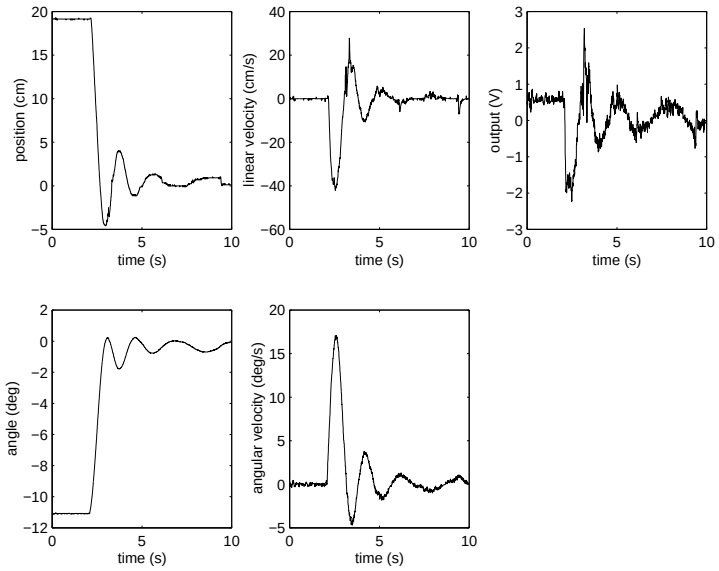
Figuur 8: simulaties van het realistischere model
Er kan ook eens gecontroleerd worden wat de invloed is van de meetruis op het resultaat van de controller. Hieruit blijkt dat als de meetruis een variantie heeft die groter is dan $5 * 10^{-}7$, dan blijkt de controller onstabiel te worden en osciilleert de output naar oneindig.

R=7 Q=[2000 7000 0 0]
R=3 Q=[1000 7000 0 0]
R=3 Q=[1000 20000 0 0]
R=3 Q=[1000 200000 0 0]
R=1 Q=[1000 20000 0 0]

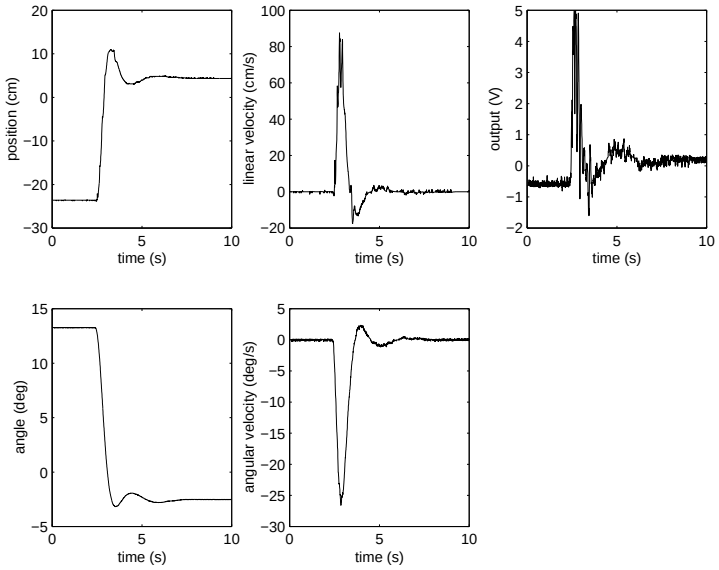
De simulaties voor Q=[1000 200000 0 0] blijken onstabiel te zijn. De polen van dat systeem zijn nogthans duidelijk negatief. Uit de andere simulaties blijkt het laatste geval de beste oplossing te geven: snelste settling tijd en kleinste steady state error. De resultaten zijn te zien figuur 8.

3 Proof of the pudding

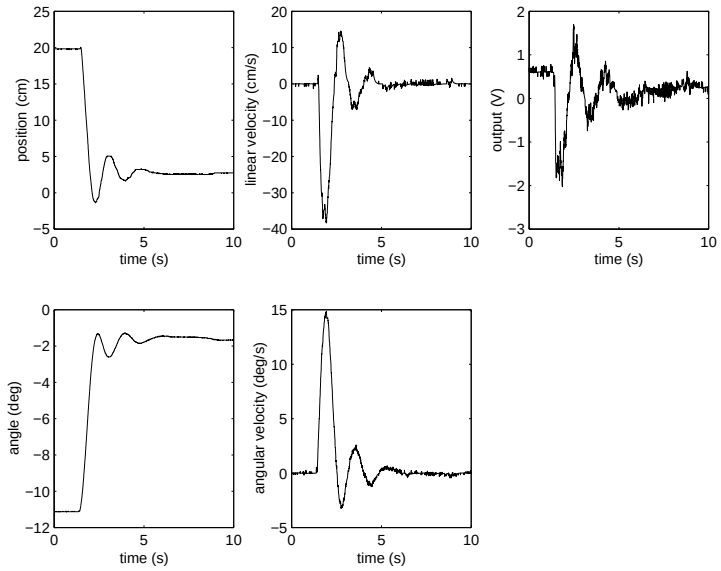
We hebben op de opstelling getest met verschillende R en Q waarden. De resultaten van 3 van die simulaties vind je hieronder:
figuur 9: R=7 en Q=[1000 5000 0 0]
figuur 10: R=5 en Q=[2000 5000 0 0]
figuur 11; R=7 en Q=[2000 7000 0 0]



Figuur 9: praktijksimulaties voor R=7 en Q=[1000 5000 0 0]



Figuur 10: praktijksimulaties voor R=5 en Q=[2000 5000 0 0]



Figuur 11: praktijksimulaties voor R=7 en Q=[2000 7000 0 0]

Als we deze figuren bekijken, zien we dat we de kleinste steady state error krijgen bij onze gekozen waarden (figuur 9, maar de settling time is wel redelijk groot. We krijgen de snelste settling in het 2de geval, maar de steady state error is groter dan in het 3de geval. In het geval van R=7 en Q=[2000 7000 0 0] krijgen we dan een redelijke settling time en een beperkte steady-state fout. Nu hangt het af van wat het doel is van de regeling. Als een kleine steady-state error nodig is, is het eerste geval te verkiezen. Als we echter een vlugge respons willen, is het 2de geval de beste oplossing. En Als we een snelle respons willen en een beperkte steady state error, dan moeten we daar iets tussen kiezen.

4 conclusie

Bij het ontwerp van een controller is het in den beginne een beetje werken met de natte vinger. Na enige tijd krijg je echter wel wat meer voeling met het probleem, en kan je ook veel gerichter beginnen werken. De manier waarop je de controller ontwerpt, hangt sterk af van wat de verwachtingen zijn. De oplossing van het probleem zal anders zijn wanneer je streeft naar een zo snel mogelijk bereiken van de steady-state, of als je een zo klein mogelijke steady state fout wil bekomen. In den beginne hebben we redelijk veel geexperimenteerd, zonder echt naar een specifieke oplossing te zoeken. De resultaten van onze practicum-testen zijn dan ook niet echt wat we zouden bekomen wanneer we al wat meer ervaring zouden gehad hebben in de regel-materie. Zoals steeds komt de wijsheid nadat je ze had

kunnnen gebruiken. Maar aangezien we studenten zijn, zal dit wel de bedoeling zijn zeker. Er gaat niks boven een praktijkvoorbeeld om inzicht te verwerven.