Vrije Universiteit Brussel

FACULTEIT INGENIEURSWETENSCHAPPEN
Vakgroep Toegepaste Mechanica

Identificatie van de Softarm

Eindwerk voorgelegd voor het behalen van de academische graad van
Burgerlijk Werktuigkundig - Elektrotechnisch ingenieur door

Willem Deconinck

Academiejaar 2005-2006

Promotor : Prof. dr. ir. D. Lefeber

Begeleider : ir. Michaël Van Damme

Vrije Universiteit Brussel

FACULTEIT INGENIEURSWETENSCHAPPEN
Vakgroep Toegepaste Mechanica

Identificatie van de Softarm

Eindwerk voorgelegd voor het behalen van de academische graad van
Burgerlijk Werktuigkundig - Elektrotechnisch ingenieur door

Willem Deconinck

Academiejaar 2005-2006

Promotor : Prof. dr. ir. D. Lefeber

Begeleider : ir. Michaël Van Damme

Samenvatting

Identificatie van de softarm

De softarm is een pneumatische manipulator die als doel heeft een operator te helpen bij het verplaatsen van een last. De softarm moet gestuurd worden door de operator via direct contact met de te verplaatsen last. De softarm moet kunnen aanvoelen in welke richting de last wordt verplaatst en zal een belangrijk deel hiervan op zich nemen.

De softarm bestaat uit 2 geledingen die geactueerd worden door drukgestuurde geplooide pneumatische artificiële spieren. Het sturen van zo een systeem vergt kennis van het gedrag van zowel de actuatoren als de manipulator.
Kennis van het dynamisch model van de manipulator is nodig om de te actueren koppels te berekenen die nodig zijn om de softarm een bepaald traject te doen volgen. Kennis van de actuatoren laat toe te berekenen welke drukken nodig zijn in de geplooide pneumatische artificiële spieren om deze koppels te bewerkstelligen.

Voor het dynamisch model dat het gedrag van de softarm beschrijft, zijn de dynamische parameters niet precies gekend. Er zitten daarom fouten op de te actueren koppels. Voor het gedrag van de geplooide pneumatische artificiële spieren zijn de spierparameters ook niet precies gekend. Er zitten daarom fouten op de koppels die zullen ingesteld worden door de actuatoren.

Het doel van dit eindwerk is om de fouten die optreden in de sturing te minimaliseren door voor de manipulator zijn dynamische parameters, en voor de actuatoren hun spierparameters te schatten. De schatting gebeurt via kracht-, druk- en hoekmetingen voor de actuatoren, en kracht-, hoek- en hoeksnelheidsmetingen voor de manipulator.

Er zijn verschillende iteratieve methoden getest om de spierparameters voor de actuatoren te schatten: een ééndimensionale methode, gradiëntmethode en Levenberg-Marquardt methode.
Voor de manipulator is een offline-kleinste kwadratenschatter getest en een online recursieve kleinste kwadratenschatter.
Goede resultaten werden bekomen voor de identificatie van de spierparameters. Verdere studie is nodig voor het optimaliseren van de identificatie van de manipulatorarm.

Abstract

Identification of the softarm

The softarm is a pneumatic manipulator with a purpose of helping an operator to move a load. The softarm must be controlled by an operator through direct contact with the load that has to be moved. The softarm must be able to sense in which direction the load is being moved and will assist in this action by moving and carrying an important part of this load itself.

The softarm exists out of 2 links which are actuated by pressure controlled pleated pneumatic artificial muscles. The controlling of such a system requires knowledge of the behaviour of the actuators as well as the manipulator.
It is necesarry to know the dynamic model of the manipulator to calculate the torques that are required to make the softarm follow a certain trajectory. Knowledge of the actuators allows to calculate the amount of pressure required inside the pleated pneumatic artificial muscles to create these torques.

The dynamic parameters are not known precisely for the dynamic model that describes the behaviour of the softarm. Hence there are deviations on the torques that have to be actuated. For the describing of the behaviour of the pleated pneumatic artifical muscles, the muscle-parameters are also not precisely known. Hence there will be deviations on the created torques.

The purpose of this thesis is to minimalize the deviations that occur during the controlling by estimating the dynamic parameters for the dynamic model and estimating the muscle-parameters for the actuators. The estimation uses force-, pressure- and angle measurements for the actuators, and force-, angle- and angle velocity measurements for the manipulator.

Several iterative methods have been tested to estimate the muscle-parameters for the actuators: a one-dimensional method, the gradient method and the Levenberg-Marquardt method.
For the manipulator an offline least squares estimator and an online recursive least squares estimator has been tested.
Good results have been achieved for the identification of the muscle-parameters. Further study is required for the optimalization of the identification of the manipulator arm.

Résumé

Identification du softarm

Le “softarm” est un manipulateur pneumatique destiné à l'aide d'un opérateur dans le déplacement d'une charge. Le “softarm” doit etre guidé par l'opérateur par un contact direct avec la charge à déplacer. Le manipulateur doit sentir la direction dans laquelle la charge sera déplacée, et reprendre une partie importante de la force.

Le softarm est composé de deux articles actués par des muscles pneumatiques artificiels pliés controlés par la pression. Le contrôle d'un tel système nécessite la connaissance du comportement dynamique des actuateurs et du manipulateur.
La connaissance du modèle dynamique du manipulateur est nécessaire pour le calcul des couples qui doivent etre exercés si on veut que le manipulateur suivra un trajet imposé.
La connaissance du comportement dynamique des actuateurs permet de calculer les pressions nécessaires dans les muscles artificiels pneumatiques afin de réaliser ces couples requis.

Les paramètres dynamiques du modèle dynamique décrivant le softarm ne sont pas connus avec précision. Il y aura donc des fautes importantes sur les couples à exercer. De même sorte, les paramètres du comportement des muscles artificiels pneumatiques ne sont pas connus avec précision. Ceci donne une autre source d'erreurs dans les couples effectués par les actuateurs.

Le but de ce travail de fin détudes est de minimaliser les erreurs de contrôle, d'une part par une estimation des paramètres dynamiques du manipulateur et d'une autre part par une estimation des paramètres des muscles artificiels des actuateurs. Cette estimation se fait par des mesures de forces, pressions et déplacements angulaires des actuateurs, et par la mesure des forces, déplacements angulaires et vitesses angulaires du manipulateur.

Plusieurs méthodes itératives pour l'estimation des paramètres musculaires des actuateurs ont été testées: méthode unidimensionnelle, méthodes par gradients et méthode de Levenberg-Marquardt.
Pour le manipulateur, un estimateur off-line de moindres carrés a été testée aussi bien qu'un estimateur en temps réel par méthode récursive de moindres carrés.
De bons résultats ont été obtenus pour l'identification des paramtres musculaires. Des études complémentaires sont nécessaires pour l'optimisation des paramètres du manipulateur.

Dankwoord

Het maken van een eindwerk is geen eenvoudige opgave. Helemaal alleen, zonder steun of hulp zou het zelfs een onmogelijke opgave zijn. Graag zou ik dan ook iedereen willen bedanken die op een of andere wijze bijdroeg aan dit eindwerk.

Mijn ouders In de eerste plaats wil ik mijn ouders bedanken voor al de kansen die me geboden werden, de oneindige steun en toewijding die me gevormd hebben tot wie ik ben. Het geduld en begrip dat opgebracht werd bij het werken aan mijn thesis was uitermate groot.

Mijn broer en zus Zij hebben dit alles van op de eerste rij kunnen meemaken en hebben nooit geklaagd. Ze hebben me al heel hun leven moeten verduren wat waarschijnlijk niet altijd even gemakkelijk is. Daarom bedankt om nog steeds klaar te staan in nood.

Dirk Lefeber Als promotor van dit eindwerk moet het niet eenvoudig zijn om een heel aantal thesisstudenten te hebben. Toch is hij er in geslaagd om de eindwerken vlot te laten verlopen.

Michaël Van Damme Als begeleider van mijn eindwerk heeft hij echt zo hard mogelijk zijn best gedaan. Telkens als er een probleem opdook kon Michaël hier wel wat op vinden. Ook het werk gestoken in het nalezen van de thesis is een grote opgave die hij zonder klagen op zich nam. Zonder Michaël had dit eindwerk nooit geweest wat het nu is.

Christy Negrescu Thank you for the wondeful pictures you made for me, they look amazing! I also want to thank you for always being there for me. You kept me sane.

Ben, Bram, Innes, Geert De klasgenootjes waar ik altijd met een warm hart aan zal terugdenken bij het herinneren van de voorbije jaren. De vriendschap opgedaan met hen is onbetaalbaar. Ik hoop van ganser harte dat dit niet het einde is maar slechts een nieuw begin.

Jean-Paul Schepens JP zou ik graag willen bedanken voor al de technische hulp. Hij weet steeds precies te vinden wat ik nodig heb. Ook bedankt om de krachtsensor te repareren, je hebt me gered!

Bobonne die dikwijls voor het nodige krachtvoer zorgde tijdens de eenzame dagen werken aan de thesis. Ook wens ik haar te bedanken voor het tweede thuis dat ik bij haar vind.

Voor de rest nog al de klasgenootjes die ik niet vermeld heb en die zeker ook dank verdienen. Ook de buitenschoolse vrienden die ik niet vernoemd heb zou ik graag bedanken voor het begrip wanneer ik geen tijd had om “eens iets te gaan doen” wegens werken aan dit eindwerk.

Willem Deconinck

Inhoudsopgave

III Identificatie van manipulator

Lijst van figuren

Lijst van tabellen

Deel I

Inleiding

Hoofdstuk 1

Inleiding

Dit eindwerk is een verderzetting van vorige eindwerken [9], [2] die handelen over de *softarm*. De softarm is ontworpen als zijnde een pneumatische manipulator die gestuurd zal worden via direct contact met een operator. Dit wil zeggen dat de softarm een taak zal uitvoeren – bvb. een last dragen – terwijl de operator de last manueel stuurt naar een gewenste plaats. Het manueel sturen gebeurt dan niet via joysticks maar door de last zelf in de gewenste richting te duwen. Zo heeft de operator een zekere voeling met de last. De softarm vermijdt dat de operator rugklachten en andere gezondheidsproblemen kan krijgen die ontstaan door de last bij zulke repetitieve handelingen zelf volledig te dragen.

Als actuator wordt gebruik gemaakt van *Geplooide Pneumatische Artificiële Spieren* die worden beschreven in Hoofdstuk 2. Verder zal naar deze geplooide pneumatische artificiële spieren gerefereerd worden als *GPAS* of gewoon *spieren*. Deze worden via druk gestuurd. De voordelen van deze actuatoren zijn:

- Ze laten toe een lichte constructie te gebruiken voor de manipulator. Dit komt omdat de spieren een licht gewicht hebben.
- De materiaalkost is klein. Het maken van een GPAS is echter niet eenvoudig.
- Dure kracht- en koppelsensoren zijn niet strikt nodig als de spierparameters goed gekend zijn. Dan kunnen kracht en koppel geschat worden door drukmetingen in de spieren.
- Montage behoeft slechts enkele schroeven. Geen transmissies zijn nodig die de kosten opdrijven en meer onderhoud vergen.
- Intrinsieke veiligheid

Vooral intrinsieke veiligheid is een belangrijke factor. De natuur van een GPAS zorgt ervoor dat een impact op de manipulator resulteert in een meeverend effect met een stijfheid die controleerbaar is. Ook is er geen gevaar op elektrocutie, en door het lichte gewicht is het gevaar voor de operator bij problemen kleiner.

In voorgaande eindwerken is een schaalmodel van de softarm ontworpen omdat de oorspronkelijke versie niet handig is om onderzoek op te verrichten. Vervolgens zijn verschillende potentiële sturingen onderzocht.

De controle van de manipulator staat echter nog niet op punt. De parameters die voorkomen in wiskundige modellen die het te controleren systeem beschrijven zijn niet precies gekend. Het is dus mogelijk dat het wiskundig model het werkelijke systeem niet goed benadert. Dit is één van de redenen waarom de controle nog niet optimaal verloopt. Om de controle te verbeteren is het namelijk aan te raden dat het wiskundig model dat het systeem beschrijft zo goed mogelijk de werkelijkheid benadert.

Het doel van dit eindwerk is dan ook om het wiskundig model van deze manipulator en actuatoren zo goed mogelijk met de werkelijkheid te doen overeenstemmen. Dit gebeurt door de onbekende of niet precies gekende parameters die voorkomen in deze wiskundige modellen te identificeren.

Het is de wens dat de data die gebruikt wordt bij het identificatieproces niet gemeten wordt via een testopstelling, maar op de softarm. Daardoor kan de identificatie snel gebeuren zonder dat de spieren uit de softarm dienen gehaald te worden, en gebeurt de identificatie met data met dezelfde omstandigheden als in reële werking.

In Deel I wordt een kort overzicht gegeven van de softarm.

In Deel II wordt beschreven hoe de actuatoren geïdentificeerd worden. Er worden drie methoden getest: een ééndimensionale methode, de gradiëntmethode en de Levenberg-Marquardt methode. Ze zijn alledrie gebaseerd op een kleinste kwadratenschatting.

In Deel III wordt beschreven hoe de dynamische parameters van de manipulator worden geïdentificeerd. Hiervoor wordt ook het kleinste kwadratenprincipe gebruikt

Hoofdstuk 2

Overzicht van de softarm

In dit hoofdstuk wordt het werkingsprincipe en de opbouw van de softarm. Voor het wiskundig model dat de spier beschrijft wordt verwezen naar Hoofdstuk 3. Voor het dynamisch model dat de softarm beschrijft wordt verwezen naar Hoofdstuk 9.

2.1 Geplooide pneumatische artificiële spier

De geplooide pneumatische artificiële spier is ontwikkeld in de VUB in de vakgroep Toegepaste Mechanica.

Ze bestaat uit een geplooid membraan dat cilindrisch gemonteerd wordt in bevestigingsstukken. Dit vormt een vervormbare drukkamer. De plooien bevinden zich volgens de langsrichting van deze cilinder. In de plooien van het membraan worden niet-elastische kevlar-vezels over de langsrichting bevestigd die de spanningen opvangen. Eén van de bevestigingsstukken bevat een druktoevoerkanaal.

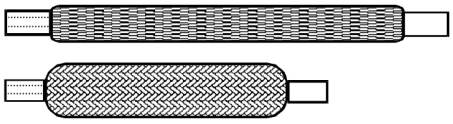
Door de spier op druk te brengen zullen de plooien ontvouwen. Er ontstaat een uitzetting van de diameter en een contractie in de lengterichting van de spier. Die contractie veroorzaakt een kracht volgens de lengterichting. De GPAS en zijn werkingsprincipe zijn getoond in Figuur 2.1.

Door een kracht op de spier uit te oefenen die de contractie tegenwerkt, zal er meer druk nodig zijn om dezelfde contractie teweeg te brengen dan in het geval zonder kracht. Er is dus duidelijk een verband tussen kracht, contractie en druk. Dit verband wordt nader bekeken in Hoofdstuk 3.

Er bestaan verscheidene designs voor artificiële spieren. De McKibben spier is de meest bekende (zie Figuur 2.2). Deze is opgebouwd uit een rubberen darm waarrond een net van vezels zit die de spanningen opvangen. Het gebruik van deze spier heeft echter enkele nadelen. Een eerste nadeel is dat er een grote drempeldruk nodig is voordat er enige contractie kan optreden. Bij de GPAS is er geen drempeldruk nodig om contractie te veroorzaken. Hier ontvouwen de plooien zich in vergelijking met het vervormen van de rubberen darm bij de McKibben spier. Een tweede nadeel is het optreden van een groot hysteresisverschijnsel. Waarom er hysteresis optreedt bij de McKibben spier is omdat er wrijving bestaat tussen het net en de rubberen darm tijdens vervormingen. Dit is een beperking die de GPAS minder heeft. Bij de GPAS is er ook een hysteresisverschijnsel maar het is veel minder groot. De hysteresis bij de

Figuur 2.1: De GPAS en zijn werkingsprincipe

GPAS is kleiner eveneens omdat het membraan zich ontvouwt (in tegenstelling tot het vervormen van het rubber bij de McKibben spier).



Figuur 2.2: De McKibben spier en zijn werkingsprincipe

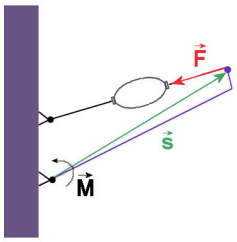
2.1.1 Rotatieve actuator

Een GPAS is een lineaire actuator: de druk veroorzaakt een contractie in de lijn van zijn aanhechtingspunten. Om over te gaan op een rotatieve actuator moet gebruik gemaakt worden van een hefboom. Hoe dit gerealiseerd wordt is te zien in Figuur 2.3.

Hierin zijn enkele symbolen gedefinieerd: $\vec{F}$, $\vec{s}$ en $\vec{M}$. Deze stellen respectievelijk voor: de kracht uitgeoefend door de spier, de vector die het aanhechtingspunt van de spier op de link beschrijft vanuit zijn scharnier, en het koppel dat de kracht $\vec{F}$ teweeg brengt in het aanhechtingspunt van de link. In dit aanhechtingspunt kan $\vec{M}$ geschreven worden als:

$$\vec{M} = \vec{s} \times \vec{F} \tag{2.1}$$

In het gravitatieveld is er tevens een lastkoppel dat het koppel $\vec{M}$ tegenwerkt. Zo onstaat er een evenwichtssituatie. Bij een andere druk is er een andere evenwichtssituatie. $\vec{s}$ heeft een vaste norm en hangt enkel af van de hoek die de link maakt ten

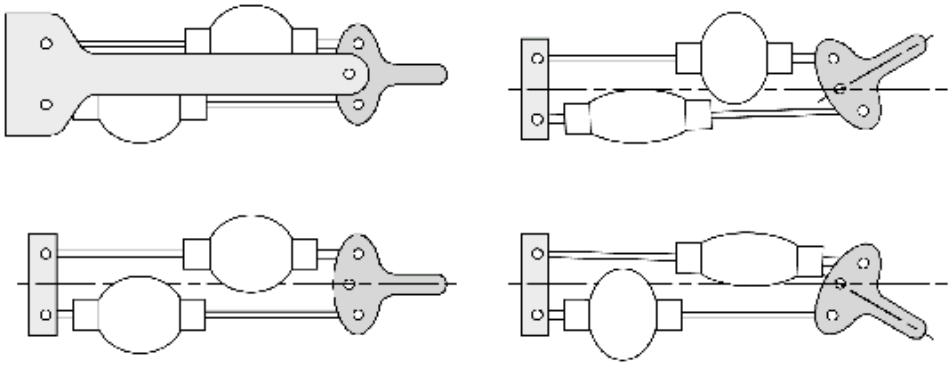


Figuur 2.3: Het gebruik van een GPAS als rotatieve actuator

opzichte van de buitenwereld. Meerdere links zijn mogelijk door de aanhechtingspunten met de buitenwereld in Figuur 2.3 te vervangen door aanhechtingspunten met een vorige link.

2.2 Antagonistisch werkingsprincipe

Indien er geen koppel aanwezig is dat $\vec{m}$ tegenwerkt, of als het tegenwerkende koppel van richting kan veranderen, is gebruik van een opstelling als in paragraaf 2.1.1 af te raden. Het tegenwerkende koppel dat voor een evenwichtssituatie zorgt kan wel gerealiseerd worden door een tegenwerkende spier zoals geschetst is in Figuur 2.4.



Figuur 2.4: Het antagonistisch werkingsprincipe

Als de ene spier een contractie ondergaat, ondergaat de andere een uitrekking. In het menselijk lichaam vindt men hetzelfde principe terug: hier zijn o.a. biceps en triceps antagonistisch opgesteld.

De sturing van een antagonistisch paar kan gebeuren met een gemiddelde druk p_m en een drukverschil tussen met de gemiddelde druk Δp. Als p_1 overeen stemt met de druk in de ene spier en p_2 de druk in de andere spier kunnen deze geschreven worden als:

$$p_1 = p_m + \Delta p \tag{2.2}$$
$$p_2 = p_m - \Delta p \tag{2.3}$$

Door Δp te veranderen zal de de kracht waarmee de ene spier trekt vergroten en de kracht waarmee de andere spier trekt verminderen.

Een tweede reden waarom deze opstelling voordelen biedt is dat de stijfheid van de rotatieve actuator kan gecontroleerd worden. Indien de gemiddelde druk p_m verhoogd wordt in beide spieren zullen de krachten waarmee ze beide trekken verhoogd worden. Dit wil zeggen dat een impact op het gelid een minder meeverend effect zal veroorzaken. Het systeem is dan stijver.

In een gravitatieveld zal de positie echter wel nog variëren bij een verandering van gemiddelde druk. Het lastkoppel (gewicht van het gelid en/of extra gewicht door een last) dat samenwerkt met het koppel veroorzaakt door de onderste spier zal bij lage gemiddelde druk relatief meer bijdragen tot het neerwaartse koppel dan bij hoge gemiddelde druk.

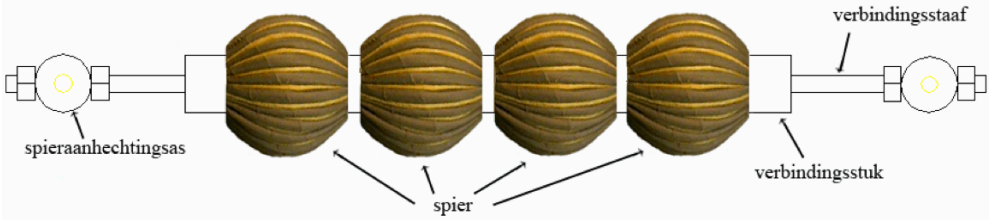
2.3 Ontwerp van de softarm

De softarm is ontworpen als een slanke manipulator met een licht gewicht. Het doel is om te bewegen in het vertikale vlak. Als ontwerp is gekozen voor een omgekeerde elleboog. Deze laat maximum vrijheid van bewegen toe voor de operator (geen obstructie) en biedt bovendien een groot werkbereik. De oorspronkelijk ontworpen lengte voor een gelid bedraagt 1m maar om practische redenen zoals veiligheid en testbaarheid is een schaalmodel ontworpen met schaal 3:10. De ontwerpparameters van de het oorspronkelijk model en schaalmodel zijn getoond in Bijlage A.

2.3.1 Keuze van spieren

Om een zo groot mogelijk werkbereik te hebben zijn grote contracties vereist. Een lange spier kan dit realiseren. Bij grote contractie is dan echter de opbolling zo groot dat het design van de softarm zelf deze uitzetting in de breedte moet kunnen toelaten. De bedoeling was echter een slanke arm te hebben.

De oplossing ligt in het gebruik van kleinere spieren in serie. De totale contractie is dan gelijk aan de som van de individuele contracties. De kracht is echter die van een individuele spier. Die kracht is wel kleiner dan die van een enkele spier die even lang is als de serieschakeling – zie vergelijking (3.2) verderop voor de formule van de spierkracht. Een schets van deze opstelling is gegeven in Figuur 2.5.

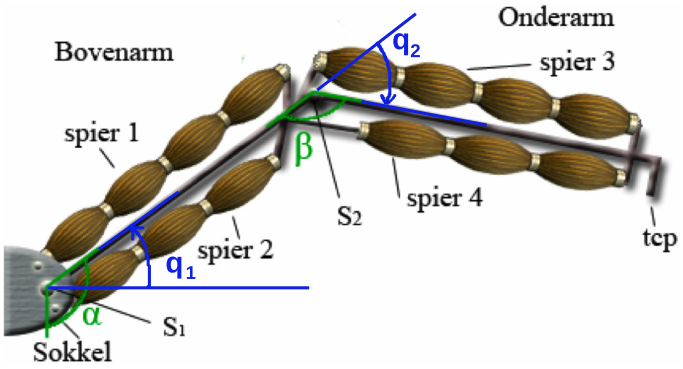


Figuur 2.5: Seriële spieren voor gebruik in de softarm

Een individuele spier heeft een ontwerplengte van 6cm en een ontwerpstraal van 1cm bij 0% contractie. Voortaan zal naar een serieschakeling van individuele spieren gerefereerd worden als *spier*.

2.3.2 Totale softarm

Een schets van welke spieren gebruikt worden voor de softarm is te zien in Figuur 2.6. Op deze figuur zijn hoeken aangeduid die gebruikt worden om de beweging van de softarm te beschrijven.



Figuur 2.6: Hoeken aangeduid op de softarm

De hoeken α en β zijn de hoeken die eenvoudig te hanteren zijn bij het bestuderen van de beweging. In dit eindwerk echter wordt steeds gewerkt met relatieve hoeken, die de hoek beschrijven tussen het verlengde van de vorige link en de langsrichting van de link waarvan men de hoek wenst te kennen. q_1 is de relatieve hoek voor link 1. Aangezien deze de eerste link is, is dit een absolute hoek. q_2 is de relatieve hoek voor link 2. Deze wordt gemeten ten opzichte van de eerste link.

In Figuur 2.6 is ook de nummering van de spieren getoond. Voortaan zal telkens volgens het spiernummer gegeven in deze figuur naar de overeenkomstige spier gerefereerd worden. Voor spieren 2 en 4 is slechts een serieschakeling van 3 individuele spieren genomen in plaats van 4. Deze spieren verzorgen immers bijna over het hele werkbereik geen draagvermogen. Ze worden dan enkel gebruikt om de stijfheid te controleren – zie paragraaf 2.2. Daarom worden ze toegelaten om een grotere contractie te ondergaan.

Spier 1 en spier 3 hebben 25 kevlarvezels, spier 2 en spier 4 hebben 40 kevlarvezels.

De softarm heeft als gewenst werkbereik:

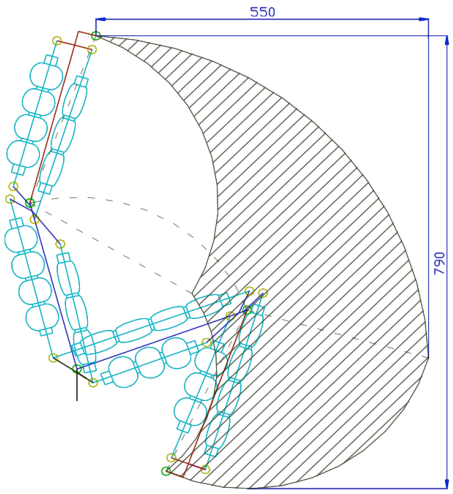
$$\begin{aligned} 110^\circ &\leqslant \alpha \leqslant 195^\circ \\ 50^\circ &\leqslant \beta \leqslant 150^\circ \end{aligned}$$

In relatieve hoeken is dit:

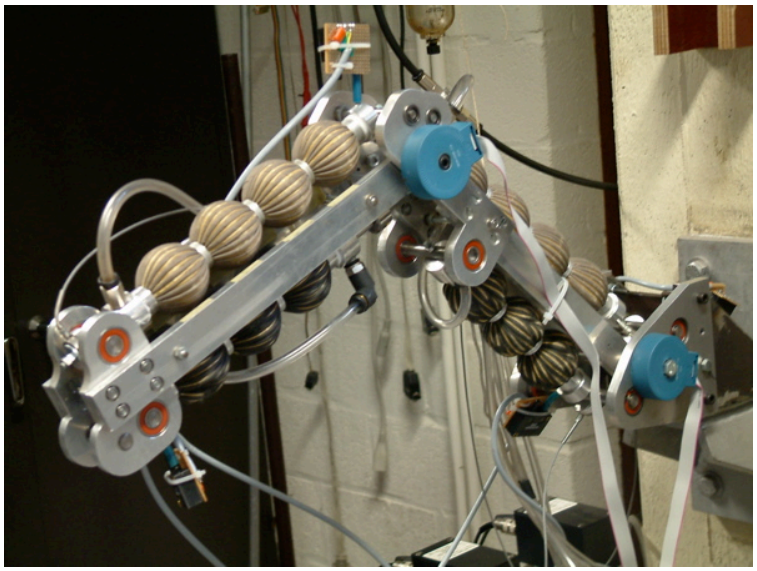
$$\begin{aligned} 20^\circ &\leqslant q_1 \leqslant 105^\circ \\ -130^\circ &\leqslant q_2 \leqslant -30^\circ \end{aligned}$$

Dit werkbereik is voorgesteld in Figuur 2.7.

Het werkelijk gebouwde schaalmodel voor de softarm is weergegeven in Figuur 2.8



Figuur 2.7: Het gekozen werkbereik van de softarm – afstanden in mm



Figuur 2.8: De gebouwde softarm

2.4 Conclusie

In dit deel is een overzicht gegeven over de stand van zaken vòòr de start van dit afstudeerwerk. De problematiek is kort geschetst. Er is besproken wat het concept softarm is: welk doel, welk ontwerp, het werkbereik, en dergelijke meer. De actuatoren die gebruikt worden zijn besproken met motivatie voor hun keuze.

Volgende delen gaan dieper in op de actuatoren en hun eigenschappen, en de manipulator en zijn karakteristieken.

Deel II

Identificatie van actuatoren

Hoofdstuk 3

Inleiding

In dit Deel wordt de identificatie van de actuatoren beschreven. Zoals zal blijken is dit geen eenvoudige opgave. Er zullen 3 methoden bekeken worden die de identificatie kunnen volbrengen: een ééndimensionale methode, de gradiëntmethode en de Levenberg-Marquardt methode.

De bedoeling van dit eindwerk is niet om een identificatie van de spieren uit te voeren in een testopstelling, maar om data te gebruiken die rechstreeks gemeten is op de softarm. De voordelen hiervan zijn tweeledig: enerzijds dient de softarm niet ontmanteld te worden om zijn spieren te identificeren. En anderzijds is de gebruikte data in overeenstemming met de reële werkomstandigheden.

Na de identificatiemethoden te hebben besproken zal beschreven worden hoe de data opgemeten wordt en omgezet wordt in "voor de identificatie bruikbare data".

Uiteindelijk zal een hoofdstuk gewijd worden aan de resultaten van de identificatie op de softarmdata.

Dit hoofdstuk zal de nodige wiskundige basis leggen die door de identificatiemethoden gebruikt worden.

3.1 Wiskundig model

Het wiskundig model van een GPAS geeft de opgewekte kracht F_t en wordt gekarakteriseerd door verschillende parameters:

- p : de relatieve druk in de spier, ten opzichte van de atmosferische luchtdruk
- L : de maximale lengte van de spier
- R : de straal van de spier bij maximale lengte
- n : het aantal kevlarvezels bevestigd op de spier (zie paragraaf 2.1)
- ε: de contractie van de spier

Volgende formule opgesteld in [3] en verfijnd in [10] geeft het verband tussen deze parameters en F_t . De voorkomende parameters m en φ_r zijn functie van L, R en ε.

$$F_t = \frac{n}{2\pi} \sin\left(\frac{2\pi}{n}\right) pR^2 \frac{1-2m}{2m\cos^2\varphi_r} \tag{3.1}$$

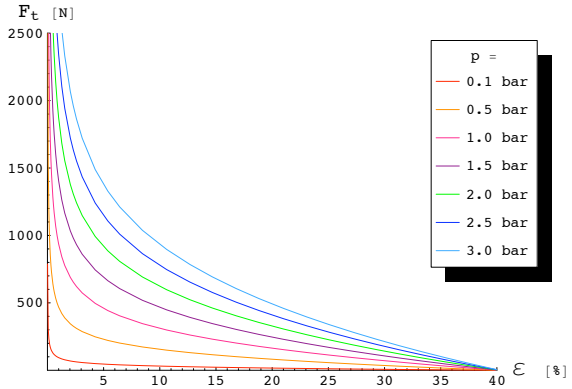
$$\triangleq pL^2 f_{t0}(\varepsilon, L/R, n) \tag{3.2}$$

f_{t0} wordt de dimensieloze spierfunctie genoemd en is functie van ε – de contractie, en L/R – de slankheid van de spier.

$$f_{t0}(\varepsilon, L/R, n) = \frac{n}{2\pi} \sin\left(\frac{2\pi}{n}\right) \frac{R^2}{L^2} \frac{1-2m}{2m\cos^2\varphi_r} \tag{3.3}$$

Dit is de niet-elastische benadering. Bij hoge krachten zullen de kevlarvezels (zie paragraaf 2.1) elastisch gedrag vertonen. Vergelijking (3.2) houdt geen rekening met dit gedrag.

Figuur 3.1 toont vergelijking (3.2) grafisch in functie van ε, voor verschillende waarden van p.



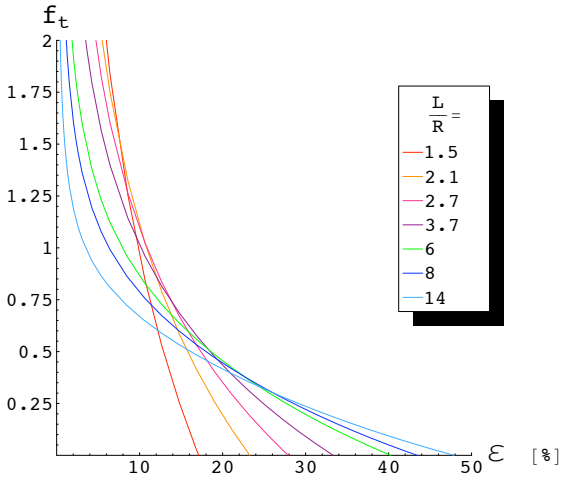
Figuur 3.1: F_t i.f.v. ε voor verschillende waarden van p – met $L = 6$cm en $R = 1$cm.

De dimensieloze spiervergelijking is getoond in Figuur 3.2 voor verschillende slankheden $\frac{L}{R}$. Het verband tussen f_{t0} en ε is niet meteen zichtbaar in vergelijking (3.3). De parameter ε wordt echter gebruikt om de parameters φ_R en m te berekenen. De parameters φ_R en m worden numeriek berekend door het oplossen van stelsel vergelijkingen (3.4) [3] via ingegeven numerieke waarden van L,R en ε. In dit stelsel komen ellitpische integralen voor: $\mathrm{E}(\varphi_R \backslash m)$ en $\mathrm{F}(\varphi_R \backslash m)$ [1], waarvan zowel de gezochte parameter φ_R als de gezochte parameter m de argumenten zijn Het is onmogelijk om dit stelsel analytisch op te lossen.

$$\begin{cases} \dfrac{\mathrm{E}(\varphi_R \backslash m)}{\sqrt{m}\cos\varphi_R} = \dfrac{L}{R}\left(1 - \dfrac{\varepsilon}{2}\right) \\ \dfrac{\mathrm{F}(\varphi_R \backslash m)}{\sqrt{m}\cos\varphi_R} = \dfrac{L}{R} \end{cases} \tag{3.4}$$

Bij gebruik van de spier zijn de variabele parameters de contractie ε en de druk p. De parameters L en R zijn karakteristieken van de spier en zijn dus ten alle tijde constant. Het zijn de waarden van deze constanten die in dit eindwerk zullen geschat

[1] Voor meer informatie over elliptische integralen zie [3]



Figuur 3.2: f_{t0} i.f.v. ε voor verschillende waarden van de slankheid $\frac{L}{R}$

worden. Men kan deze waarden ook meten, maar het probleem is dat een GPAS een geplooid membraan bevat. Wat de precieze straal van zo een cilinder is is onzeker. Wat de optimale waarde voor L is die het wiskundig model gebruikt om de realiteit het best te beschrijven is ook niet helemaal zeker. Zoals zal blijken heeft deze parameter een grote invloed.

3.2 De identificatie

Om een systeemidentificatie te volbrengen is het noodzakelijk om over opgemeten data te beschikken. De eerste vraag die men zich moet stellen is welke parameters dienen te worden opgemeten. De enige variabele parameters in het wiskundig model van de kracht F_t – vergelijking (3.2) – zijn de druk p en de contractie ε. Het zijn dus deze parameters, samen met de output F_t die zullen moeten opgemeten worden.

Figuur 3.3: Het systeem schematisch voorgesteld

De parameters die dienen geoptimaliseerd te worden opdat het wiskundig model zo minimaal mogelijk afwijkt van de realiteit (de opgemeten data) zijn L en R. Voor

verdere notatie zullen de parameters L en R als een parametervector θ voorgesteld worden:

$$\theta = \begin{pmatrix} L \\ R \end{pmatrix}$$

Er zullen een beginwaarden voor L en R gekozen worden die dan gebruikt worden in het wiskundig model (vergelijking 3.2)). In dit wiskundig model zullen de opgemeten data van ε en p gebruikt worden. Volgens de methode van paragraaf 3.1 zal een theoretische numerieke waarde van de kracht F_t berekend worden.

Het verschil tussen deze theoretische en de werkelijke waarde is gegeven door volgende formule.

$$e(k,\theta) = F_t(\theta)\Big|_{\substack{\varepsilon(k)\\p(k)}} - F_{\text{data}}(k) \tag{3.5}$$

met k de index van de gemeten data. Een kostfunctie zal een scalair getal geven dat de mate van het verschil voor alle k weergeeft. Volgende kostfunctie wordt gebruikt, waarbij de fout gekwadrateerd wordt. Dit is een kleinste kwadraten kostfunctie.

$$V = \frac{1}{2}\sum_{k=1}^{N} e^2(k,\theta) \tag{3.6}$$

$$= \frac{1}{2}\sum_{k=1}^{N}\left(F_t(\theta)\Big|_{\substack{\varepsilon(k)\\p(k)}} - F_{\text{data}}(k)\right)^2 \tag{3.7}$$

Minimalisatie van deze kostfunctie geeft aanleiding tot het principe van de kleinste kwadraten [7]. Minimalisatie betekent de afgeleide naar beide parameters van de kostfunctie gelijk aan nul stellen[2].

Volgende vergelijkingen tonen respectievelijk de afgeleide van de kostfunctie V naar L en naar R.

$$\frac{\partial V}{\partial L} = \sum_{k=1}^{N}\left[\left(F_t(\theta)\Big|_{\substack{\varepsilon(k)\\p(k)}} - F_{\text{data}}(k)\right)\frac{\partial F_t(\theta)}{\partial L}\Big|_{\substack{\varepsilon(k)\\p(k)}}\right] \tag{3.8}$$

$$\frac{\partial V}{\partial R} = \sum_{k=1}^{N}\left[\left(F_t(\theta)\Big|_{\substack{\varepsilon(k)\\p(k)}} - F_{\text{data}}(k)\right)\frac{\partial F_t(\theta)}{\partial R}\Big|_{\substack{\varepsilon(k)\\p(k)}}\right] \tag{3.9}$$

Hierin kunnen $\frac{\partial F_t(\theta)}{\partial L}$ en $\frac{\partial F_t(\theta)}{\partial R}$ verder bekeken worden:

$$\frac{\partial F_t(\theta)}{\partial L} = 2pLf_{t0}(\varepsilon,\theta) + pL^2\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L} \tag{3.10}$$

$$\frac{\partial F_t(\theta)}{\partial R} = pL^2\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R} \tag{3.11}$$

Hier rijzen de eerste problemen. F_t is namelijk geen lineaire functie in L en in R. Zo zal bvb. in vergelijking (3.9) de parameter L voorkomen via vergelijking (3.11). Het minimum van deze kostfunctie voor de parameter R zal dus afhangen van L. Dit geldt ook omgekeerd voor vergelijking (3.8). Doordat het wiskundig model niet lineair

[2]Dit geldt ook voor maximalisatie, de tweede afgeleide berekenen levert uitsluitsel of het om een minimum of om een maximum gaat dat berekend wordt.

is in de parameters, zijn de parameters aldus gekoppeld.

De volgende moeilijkheid bevindt zich nu in het berekenen van $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L}$ en $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R}$: $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L}$ zal niet enkel meer functie zijn van m en φ_R, maar ook van $\frac{\partial m}{\partial L}$ en $\frac{\partial \varphi_R}{\partial L}$.
Dit is getoond in volgende vergelijking:

$$\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L} = \\ -\frac{n}{2\pi}\sin\left(\frac{2\pi}{n}\right)\frac{R^2\sec^2(\varphi_R)}{2L^3m^2}\left(L\frac{\partial m}{\partial L}+2m\left(-1+2m\right)\left(-1+L\frac{\partial \varphi_R}{\partial L}\tan(\varphi_R)\right)\right) \tag{3.12}$$

Om $\frac{\partial m}{\partial L}$ en $\frac{\partial \varphi_R}{\partial L}$ te berekenen wordt gebruik gemaakt van het stelsel vergelijkingen (3.4).
Afleiden van dit stelsel naar L geeft:

$$\begin{cases} \dfrac{\partial}{\partial L}\left[\dfrac{\mathrm{E}(\varphi_R\backslash m)}{\sqrt{m}\cos\varphi_R}\right] = \dfrac{1}{R}\left(1-\dfrac{\varepsilon}{2}\right) \\ \dfrac{\partial}{\partial L}\left[\dfrac{\mathrm{F}(\varphi_R\backslash m)}{\sqrt{m}\cos\varphi_R}\right] = \dfrac{1}{R} \end{cases} \tag{3.13}$$

De onbekenden van dit stelsel zijn $\frac{\partial m}{\partial L}$ en $\frac{\partial \varphi_R}{\partial L}$ in functie van m, φ, ε, L en R. Deze worden gesubstitueerd in vergelijking (3.12) [3] met als resultaat:

$$\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L} = \tag{3.14}$$

```
(n/(2*pi)*sin((2*pi)/n)*R^2*Sec(phi)^2*((-4*L*(-1 + m)*m*cos(phi)*(eps - m +
m*cos(2*phi)))/(-2*eps*L*(-1 + m)*cos(phi) + E(phi,
m)*(-4*sqrt(m)*R*cos(phi)^2 + L*(eps - 2*m)*sqrt(4 - 2*m +
2*m*cos(2*phi))*sin(phi)) + sin(phi)*((-1 + m)*F(phi, m)*
(eps*L*sqrt(4 - 2*m + 2*m*cos(2*phi)) + 4*sqrt(m)*R*sin(phi))
+ 4*sqrt(m)*R*cos(phi)*sqrt(1 - m*sin(phi)^2) +
L*m*(-eps + 2*m)*sin(2*phi))) + 2*m*(-1 + 2*m)*(1 -
(L*sin(phi)*((eps - 2*m)*sqrt(4 - 2*m + 2*m*cos(2*phi))*
E(phi, m) + eps*(-1 + m)*sqrt(4 - 2*m + 2*m*cos(2*phi))*
F(phi, m) - (-2 + eps)*m*sin(2*phi)))/(-2*eps*L*(-1 + m)
*cos(phi) + E(phi, m)*(-4*sqrt(m)*R*cos(phi)^2 + L*(eps - 2*m)*
sqrt(4 - 2*m + 2*m*cos(2*phi))*sin(phi)) + sin(phi)*((-1 + m)
*F(phi, m)*(eps*L*sqrt(4 - 2*m + 2*m*cos(2*phi)) +
 4*sqrt(m)*R*sin(phi)) + 4*sqrt(m)*R*cos(phi)*sqrt(1 -
m*sin(phi)^2) + L*m*(-eps + 2*m)*sin(2*phi))))))/(2*L^3*m^2)
```

$\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R}$ zal eveneens niet enkel meer functie zijn van m en φ_R, maar ook van $\frac{\partial m}{\partial R}$ en $\frac{\partial \varphi_R}{\partial R}$. Dit is in volgende vergelijking te zien:

[3] Als notatie voor φ_R is hier voor de eenvoud "phi" genomen.

$$\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R} = \\ -\frac{n}{2\pi}\sin\left(\frac{2\pi}{n}\right)\frac{R\sec^2(\varphi_R)}{2L^2m^2}\left(R\frac{\partial m}{\partial R}+2m\left(-1+2m\right)\left(1+R\frac{\partial \varphi_R}{\partial R}\tan(\varphi_R)\right)\right) \quad (3.15)$$

$\frac{\partial m}{\partial R}$ en $\frac{\partial \varphi_R}{\partial R}$ zullen analoog berekend worden als de partieel afgeleiden naar L. Afleiden van stelsel vergelijkingen (3.4) naar R geeft:

$$\begin{cases} \dfrac{\partial}{\partial R}\left[\dfrac{\mathrm{E}(\varphi_R\backslash m)}{\sqrt{m}\cos\varphi_R}\right] = -\dfrac{L}{R^2}\left(1-\dfrac{\varepsilon}{2}\right) \\ \dfrac{\partial}{\partial R}\left[\dfrac{\mathrm{F}(\varphi_R\backslash m)}{\sqrt{m}\cos\varphi_R}\right] = -\dfrac{L}{R^2} \end{cases} \quad (3.16)$$

Substitutie van de gevonden $\frac{\partial m}{\partial R}$ en $\frac{\partial \varphi_R}{\partial R}$ in vergelijking (3.15) [4] leidt tot:

$$\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R} = \quad (3.17)$$

```
-(n/(2*pi)*sin((2*pi)/n)*R*Sec(phi)^2*((-4*L*(-1 + m)*m*cos(phi)*(eps - m +
m*cos(2*phi)))/(-2*eps*L*(-1 + m)*cos(phi) + E(phi, m)
*(-4*sqrt(m)*R*cos(phi)^2 + L*(eps - 2*m)*sqrt(4 - 2*m +
2*m*cos(2*phi))*sin(phi)) + sin(phi)*((-1 + m)*F(phi, m)*
(eps*L*sqrt(4 - 2*m + 2*m*cos(2*phi)) + 4*sqrt(m)*R*sin(phi))
+ 4*sqrt(m)*R*cos(phi)*sqrt(1 - m*sin(phi)^2)
+ L*m*(-eps + 2*m)*sin(2*phi))) + 2*m*(-1 + 2*m)*
(1 + (L*sin(phi)*((-eps + 2*m)*sqrt(4 - 2*m +
2*m*cos(2*phi))*E(phi, m) - eps*(-1 + m)*
sqrt(4 - 2*m + 2*m*cos(2*phi))*F(phi, m) +
(-2 + eps)*m*sin(2*phi)))/(-2*eps*L*(-1 + m)*cos(phi) +
E(phi, m)*(-4*sqrt(m)*R*cos(phi)^2 + L*(eps - 2*m)*
sqrt(4 - 2*m + 2*m*cos(2*phi))*sin(phi)) +
sin(phi)*((-1 + m)*F(phi, m)*(eps*L*sqrt(4 - 2*m +
2*m*cos(2*phi)) + 4*sqrt(m)*R*sin(phi)) +
 4*sqrt(m)*R*cos(phi)*sqrt(1 - m*sin(phi)^2) +
 L*m*(-eps + 2*m)*sin(2*phi))))))/(2*L^2*m^2)
```

Invullen van m en φ_R in vergelijkingen (3.14) en (3.17) – gevonden via het oplossen van het stelsel elliptische integralen (vergelijking 3.4) – levert dan de uitdrukking voor $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L}$ en $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R}$ in functie van enkel L, R en ε.

Zo zijn de parameters m en φ_R volledig geëlimineerd, en kan de afgeleide van de kostfunctie V berekend worden in functie van L en R.

Op te merken valt dat door het gebruik van kleinste kwadraten de oplossing een bias kan bezitten [8]. Dit is het geval als er veel ruis zit op de output van het systeem. In dit geval is dit $F_{\mathrm{data}}(k)$. Er word hier dus ondersteld dat de ruis op $F_{\mathrm{data}}(k)$ klein is.

[4] Als notatie voor φ_R is hier voor de eenvoud "phi" genomen.

3.3 Conclusie

Het wiskundig model is geanalyseerd en een kostfunctie volgens de kleinste kwadratenmethode is opgesteld. Ook zijn de berekeningen gebeurd om de afgeleide van de kostfunctie naar L en naar R semi-analytisch te verkrijgen. De parameters m en φ_R moeten telkens voor elke ε berekend worden via een numerieke methode uit het stelsel elliptische integralen – vergelijking (3.4). Hun afgeleiden echter zijn analytisch te berekenen met de kennis van m en φ_R.

De *bottleneck* in de berekening van de afgeleide naar L en naar R van V bevindt zich dus in de numerieke solver.

Met de in dit hoofdstuk opgestelde formules is het nu mogelijk diverse identificatiemethodes toe te passen, gebaseerd op kleinste kwadraten.

Hoofdstuk 4

Iteratieve ééndimensionale methode

4.1 Inleiding

Dit hoofdstuk beschrijft een eerst geprobeerde methode. Zoals later zal aangetoond worden is dit geen optimale methode. Het is echter een zeer intuïtieve methode. Hij is dan ook gebruikt om de eerste stappen te zetten in het identificatieproces.

4.2 Methode

De identificatie van een systeemparameter wil zeggen dat er gezocht wordt naar die waarde van de parameter waarbij de kostfunctie een globaal minimum heeft. Bij het zoeken naar de optimale L wordt dus gebruik gemaakt van vergelijking (3.8). Zoals te zien is in vergelijking (3.14) is er in $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial L}$ nog afhankelijkheid van de parameter R. Deze is echter ook onbekend. De kostfunctie V afgeleid naar L (vergelijking (3.8)) – die gebruik maakt van vergelijking (3.14) – zal dus niet eenduidig bepaald zijn. Een minimale waarde van V – ofwel een optimale waarde van L – kan echter wel gevonden worden bij een opgegeven beginwaarde van R.

In vergelijking (3.17) is ook te zien dat er nog afhankelijkheid is van L in $\frac{\partial f_{t0}(\varepsilon,\theta)}{\partial R}$, terwijl het de optimale R is die gezocht wordt. In plaats van een willekeurige waarde voor L te nemen is het nu best om de waarde van L te gebruiken die net gevonden is in de vorige stap.

Herhaling van vorige stappen tot aan een convergentiecriterium voldaan is kan leiden tot een optimale waarde van L en R. Het is dus een iteratief proces. Dit komt door de niet-lineariteit in de parameters L en R van het wiskundig model.

De toegepaste methode om het minimum te vinden van de kostfunctie voor een bepaalde parameter was in een eerste poging de afgeleide van de kostfunctie te berekenen voor een vijftiental waarden van de gezochte parameter, rond het verwachtte nulpunt van de afgeleide van de kostfunctie. Vervolgens werd een polynoom hierop gefit, waarvan het nulpunt dan numeriek berekend kon worden. Dit komt overeen met een parameter die beschouwd kan worden als de optimale parameter bij een bepaalde waarde van de andere parameter.

Deze methode is wat omslachtig aangezien de berekening voor elk van die vijftien punten moet gebeuren, waarbij telkens voor alle waarden van ε opnieuw het stelsel vergelijkingen (3.4) moet worden opgelost naar m en φ_R.

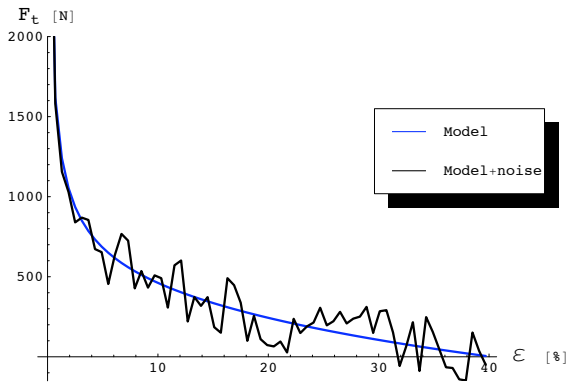
De methode bestaat er dus in om telkens een nieuw minimum te bereken voor 1 parameter, vanwaaruit een volgend minimum berekend wordt voor de andere parameter. Dit lijkt intuïtief en is de eerst geprobeerde methode.

4.3 Toepassen op theoretisch model met ruis

4.3.1 De ruis

Om deze methode te testen werd een theoretisch model gebruikt via formule (3.2) met volgende gekozen parameters: $L = 6$ cm, $R = 1$ cm, $p = 1.5$ bar. ε bestrijkt een hele range. Op dit model werd een uniforme ruis gesuperponeerd. Dit is dus de testdata waarmee het identificatiesproces moet gebeuren. Als het goed werkt moeten hier terug de waarden $L = 6$ cm en $R = 1$ cm tevoorschijn komen. Figuur 4.1 toont de data waarmee de identificatie gebeurt. Werkelijke data zal nooit zo grillig zijn, en de meetpunten zullen nooit zo ver afwijken van elkaar. Ruis is over het algemeen ook bijna nooit uniform verdeeld maar Gaussiaans. Door uniforme ruis te gebruiken wordt de methode getest op een *worst case scenario*. Indien de methode dus goede resultaten oplevert met deze uniforme ruis, kan men stellen dat Gaussiaanse ruis ook goede resultaten zal opleveren.

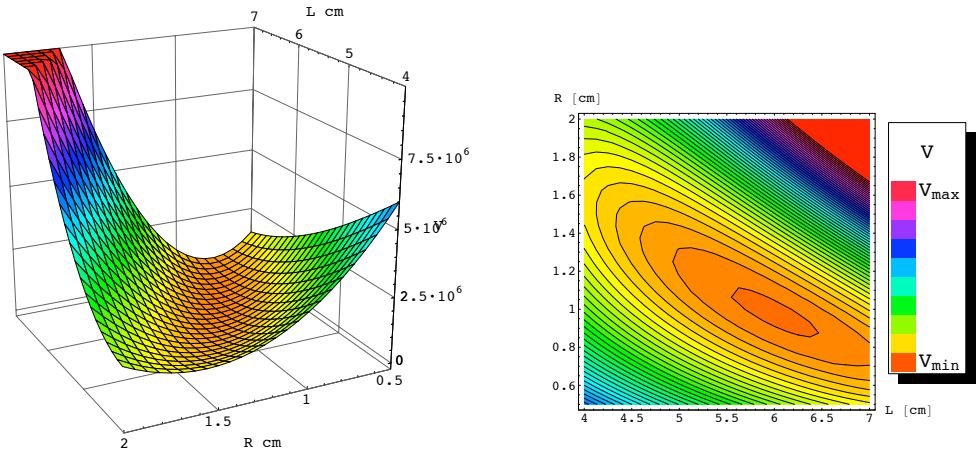
Het aantal punten dat gebruikt is, is slechts 67. Meer punten zou de rekentijd enorm verlengen. Voor testdoeleinden is dit echter een voldoende groot aantal.



Figuur 4.1: Ruis op theoretisch model van F_t (vergelijking 3.2) i.f.v. ε voor $L = 6$ cm, $R = 1$ cm en $p = 1.5$ bar.

4.3.2 De identificatie

Een plot van de kostfunctie V in functie van L en R geeft een oppervlak waarvan het minimum bij $L = 6$ cm en $R = 1$ cm ligt – zie Figuur 4.2.

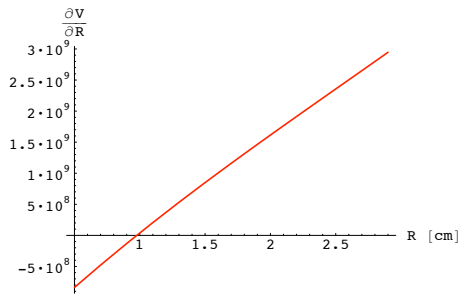


(a) 3D plot

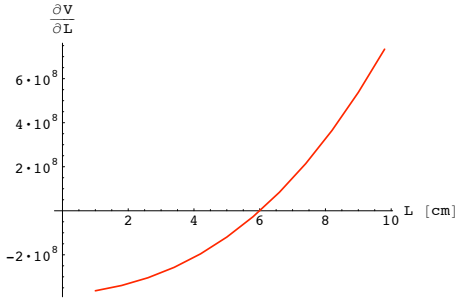
(b) Contourplot

Figuur 4.2: Kostfunctie V i.f.v. L en R voor ruis op het theoretisch model van F_t

Deze identificatiesmethode zal op dit oppervlak telkens 1 dimensie bekijken. Voor een vaste waarde van L zal eerst de optimale R berekend worden via vergelijking (3.9). Deze functie wordt berekend en het nulpunt wordt ervan gezocht. De gevonden R zal als vaste waarde gebruikt worden om L te vinden, via vergelijking (3.8). Figuren 4.3 en 4.4 tonen deze functies in functie van de gezochte parameter.



Figuur 4.3: Afgeleide van kostfunctie V naar R met $L = 6.00516$ cm. (38ste iteratie)



Figuur 4.4: Afgeleide van kostfunctie V naar L met $R = 0.97679$ cm. (38ste iteratie)

Tabel 4.1 geeft de waarden van L, R en V na elke iteratie. Na een 20-tal iteraties verandert er niets significants meer aan de waarden van R, L en V. Men kan stellen dat het verschil tussen de opeenvolgende waarden na 20 iteraties voldoende klein is om te spreken van convergentie (het verschil kleiner dan 0.001 cm).

iteratie	L [cm]	R [cm]	V
0	7.00000	1.50000	6735771.9
2	6.68897	0.76463	588338.2
4	6.48313	0.82605	522215.4
6	6.34261	0.86905	491464.8
8	6.24486	0.89956	476557.6
10	6.17603	0.92137	469147.7
12	6.12719	0.93701	465406.3
14	6.09235	0.94827	463498.0
16	6.06740	0.95638	462518.2
18	6.04950	0.96222	462012.7
20	6.03662	0.96644	461751.2
22	6.02736	0.96948	461615.5
24	6.02068	0.97167	461545.1
26	6.01587	0.97326	461508.5
28	6.01239	0.97440	461489.4
30	6.00989	0.97523	461479.5
32	6.00808	0.97583	461474.3
34	6.00678	0.97626	461471.7
36	6.00584	0.97657	461470.3
38	6.00516	0.97679	461469.5

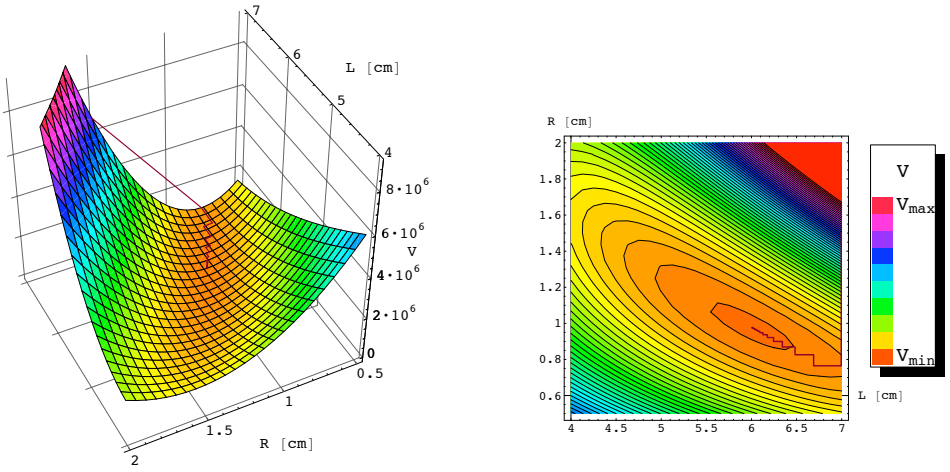
Tabel 4.1: V, L en R na elke iteratie

Het uiteindelijke resultaat na 38 iteraties is gegeven door:

$$\begin{cases} L = 6.00516 \text{ cm} \\ R = 0.97679 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 6.14785$$

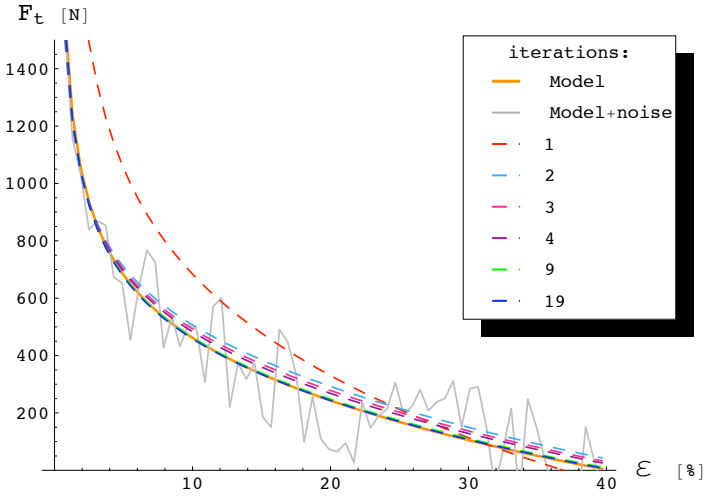
Tabel 4.1 kan ook grafisch voorgesteld worden waarbij 1 iteratie bestaat uit 2 ééndimensionale oplossingen. Figuur 4.5 toont dit.

Om nu uiteindelijk te zien of het minimum van de kostfunctie V overeen komt met een wiskundig model dat perfect ligt op het oorspronkelijke theoretische model waarop de ruis gesuperponeerd is, worden deze nog met elkaar vergeleken in Figuur 4.6. De evolutie van het wiskundig model naar het oorspronkelijk theoretisch model is ook geplot voor enkele iteraties.



(a) 3D plot (b) Contourplot

Figuur 4.5: Grafische weergave van het verloop van de gevonden waarden van L, R en V voor de opeenvolgende iteraties



Figuur 4.6: Evolutie van F_t naar het theoretische model waarop ruis gesuperponeerd is

4.3.3 Conclusie

Deze methode werkt, maar het nodige aantal iteraties ligt vrij hoog. De gekozen beginwaarde voor L speelt een grote rol. Als deze ver verwijderd is van de oplossing zal het erg lang duren vooraleer er convergentie begint op te treden. Ook is er heel veel rekentijd per iteratie nodig. Dit komt omdat er telkens een ééndimensionale optimalisatie gebeurt. De penalisatie voor een groot aantal meetpunten is dan ook erg hoog: het aantal keer dat het stelsel vergelijkingen (3.4) moet opgelost worden per iteratie is het aantal meetpunten maal het aantal punten gebruikt om het minimum te vinden van de kostfunctie, maal 2 want voor elke iteratie zijn er 2 ééndimensionale delen.
Het model is klaar om getest te worden op meer reële data.

4.4 Toepassen op trekbankdata

4.4.1 Trekbankdata

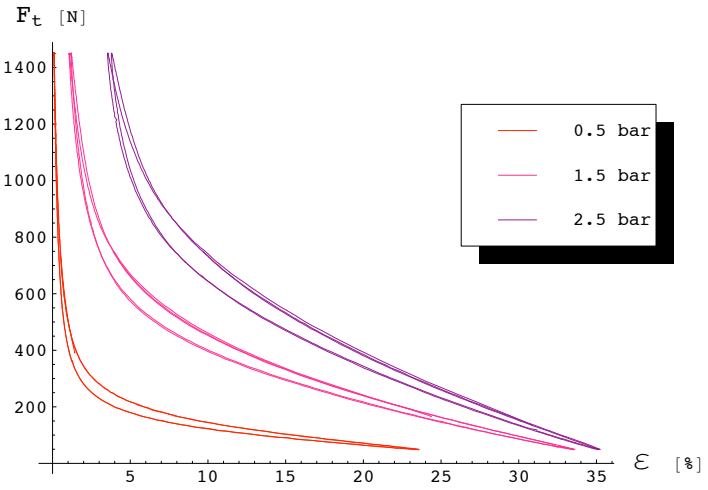
Op een trekbank is een spier bij 3 verschillende drukken bemeten: 0.5 bar, 1.5 bar en 2.5 bar. De kracht werd stapsgewijs verhoogd en de lengteverandering opgemeten. De opgemeten data bij de 3 drukken ziet eruit zoals in Figuur 4.7. Zoals te zien is, zijn er meerdere cycli opgelegd van op- en afbouw van kracht. Opvallend is het verschijnen van hysteresis. Bij het op- en afbouwen van kracht wordt er een verschillend traject gevolgd.

Dit is een probleem vermits het met het wiskundige model en de gebruikte identificatiesmethode niet mogelijk is een hysteresis te volgen. Het valt echter op te merken dat dit geen realistische data is zoals zal komen uit de softarm-opstelling. Dit is data waarvan het bereik van de contractie ε ver buiten het te gebruiken bereik ligt ($5\% \leqslant \varepsilon \leqslant 30\%$). In werkelijkheid zal de druk niet constant zijn bij een beweging. Het is net de druk die de beweging zal sturen. Ook zal het maximum verschil door hysteresis nooit een zo grote waarde kunnen aannemen als de data op de testbank. De softarm zal namelijk nooit zo een bereik bestrijken en zal variëren in zijn beweging zodat de hysteresislus desgevallend kleiner is.

In de volgende paragrafen zal getracht worden de parameters L en R te zoeken zodanig dat het zo bekomen wiskundig model het gemiddelde van de hysteresislus vormt.

Allereerst wordt elke set data bij een constante druk afzonderlijk bestudeerd. De gevonden L en R wordt dan gebruikt om ook het wiskundig model te maken voor de 2 andere sets bij een andere constante druk. Hopelijk komen deze dan overeen met de overeenkomstige data van de trekbank. Daarna wordt de combinatie gemaakt en wordt zo een soort gemiddelde gemaakt van alledrie de sets tesamen. De combinatie van de 3 datasets kan meer beschouwd worden als data afkomstig van de softarm, waar ook verschillende drukken tesamen voorkomen. Als de identificatie dus op deze data bevredigende resultaten geeft, kan data van de softarm gebruikt worden.

Elke dataset bevat meer dan 1000 punten. De identificatie op deze hoeveelheid punten uitvoeren zou veel te lang duren: ongeveer 15 minuten per iteratie. Uit deze data is dan ook slechts een beperkt aantal punten genomen waarmee er verder gerekend wordt. Ook is er slechts 1 hysteresislus uitgenomen. Het aantal punten is zo gereduceerd tot een vergelijkbaar aantal als bij het ruismodel gebruikt is (67).



Figuur 4.7: Trekbankdata

4.4.2 De identificatie

Data bij 1.5 bar

De gebruikte data voor deze identificatie is te zien in Figuur 4.8. Het aantal punten is beperkt tot 71. Dit leidt tot een kleine 5 minuten rekentijd per iteratie. Dit is al enorm veel. Het stopcriterium voor het itereren is zo gekozen dat het verschil tussen de waarde van de nieuwe kostfunctie en de kostfunctie van de vorige iteratie kleiner is dan 20.

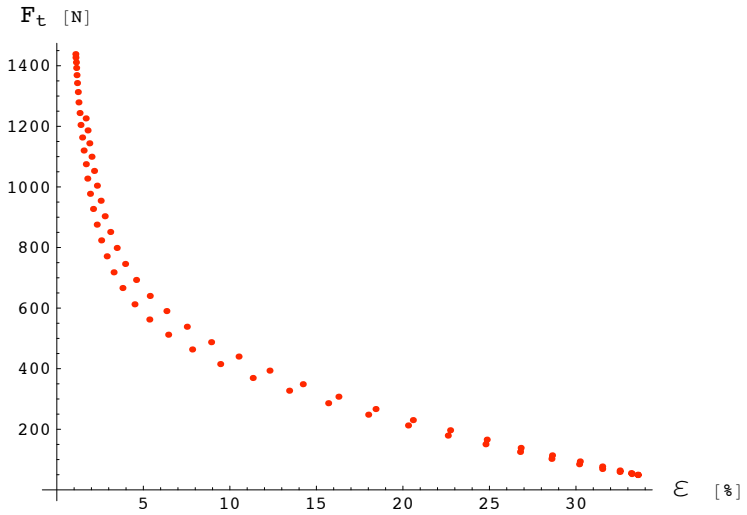
De kostfunctie samen met het gevolgde pad van L en R naar de optimale waarden zijn getoond in Figuur 4.9. Numerieke gegevens zijn te vinden in Tabel 4.2. Het is te zien dat de convergentie zeer traag verloopt. Veel trager dan het geval was bij het geval van zelf gemaakte ruis op een theoretisch model. Een totaal van 76 iteraties was nodig om aan het convergentiecriterium te voldoen. Dit toont weer dat de methode erg langzaam is en dus de berekening wel enkele uren kan duren, zeker met grote sets data.

De evolutie van het wiskundig model naar de optimale fit voor opeenvolgende iteraties toont zich in Figuur 4.10.

Zoals te zien is, is het niet mogelijk om het wiskundig model volledig te doen overeenstemmen met de data. Dit komt vooral door het elastische gedrag bij hoge kracht. Men kan dus nadenken over het feit of een fit met een veelterm geen beter idee zou zijn dan de identificatie van deze parameters L en R.

Vooraleer de boeg zo om te gooien zal eerst geprobeerd worden of het niet beter is de data die buiten het werkbereik van de softarm ligt gewoon weg te laten. De gebruikte data voor de identificatie is getoond in Figuur 4.11.

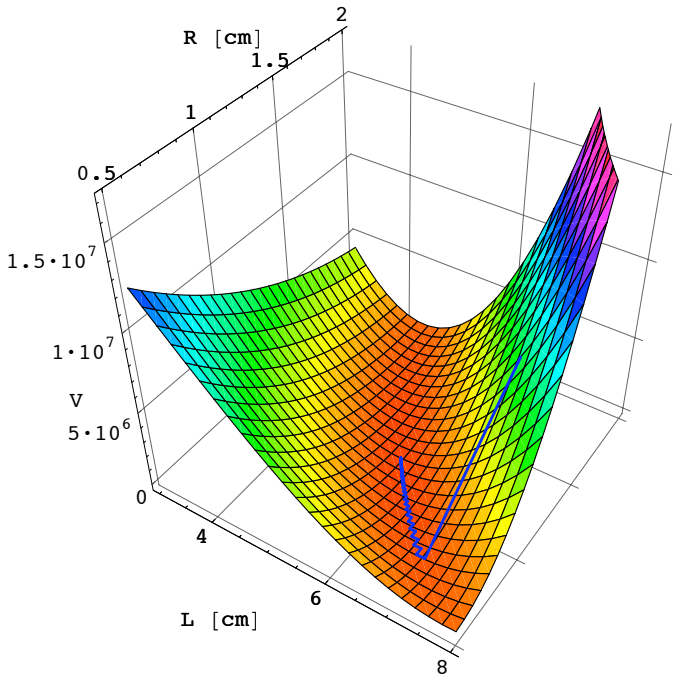
Dit blijkt inderdaad tot een goed resultaat te leiden binnen het werkgebied van de softarm. Erbuiten echter – vooral langs de kant waar $\varepsilon < 5\%$ – is de voorspelling



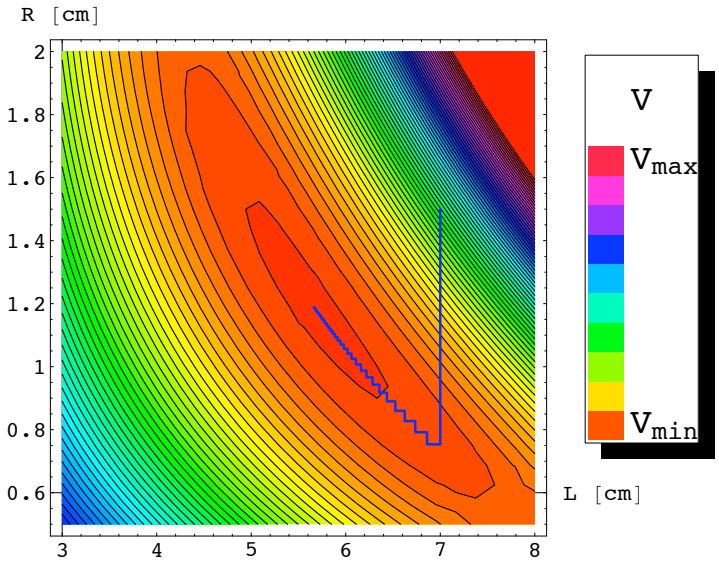
Figuur 4.8: Gebruikte data voor de identificatie; $p = 1.5$ bar

iteratie	L [cm]	R [cm]	V	L/R
0	7	1.5	6576505.7	4.667
4	6.73540	0.79204	219222.1	8.504
8	6.52472	0.85998	171463.1	7.587
12	6.35551	0.91751	139773.6	6.927
16	6.21887	0.96614	118577.3	6.437
20	6.10822	1.00707	104356.8	6.065
24	6.01848	1.04139	94816.6	5.779
28	5.94567	1.07002	88431.1	5.557
32	5.88663	1.09378	84172.0	5.382
36	5.83881	1.11340	81342.5	5.244
40	5.80010	1.12954	79470.3	5.135
44	5.76880	1.14276	78236.3	5.048
48	5.74352	1.15356	77425.8	4.979
52	5.72313	1.16234	76895.1	4.924
56	5.70669	1.16948	76548.7	4.880
60	5.69344	1.17526	76322.9	4.844
64	5.68277	1.17994	76176.2	4.816
68	5.67419	1.18372	76081.0	4.794
72	5.66729	1.18677	76019.2	4.775
76	5.66174	1.18923	75979.3	4.761

Tabel 4.2: L/R, V, L en R na elke iteratie voor trekbankdata bij 1.5 bar

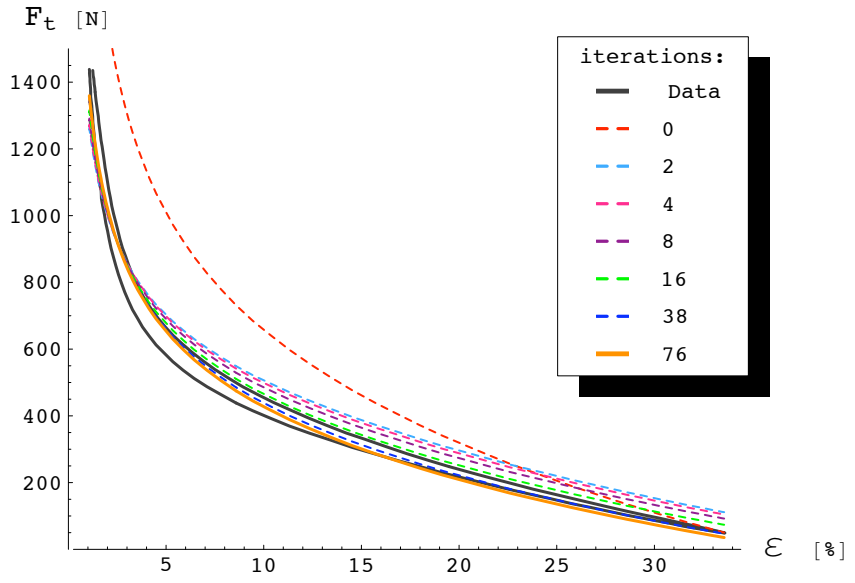


(a) 3D plot

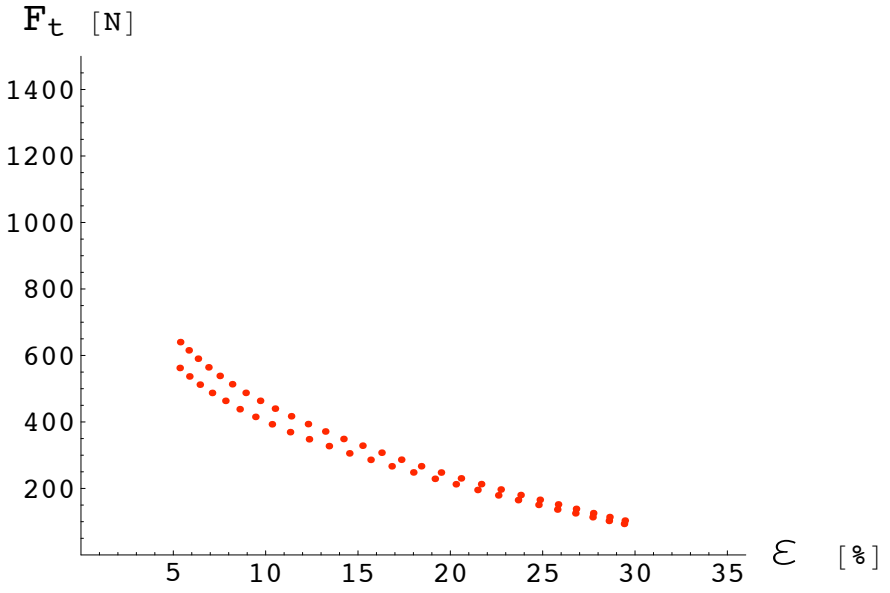


(b) Contourplot

Figuur 4.9: Kostfunctie V i.f.v. L en R voor trekbankdata bij $p = 1.5$ bar

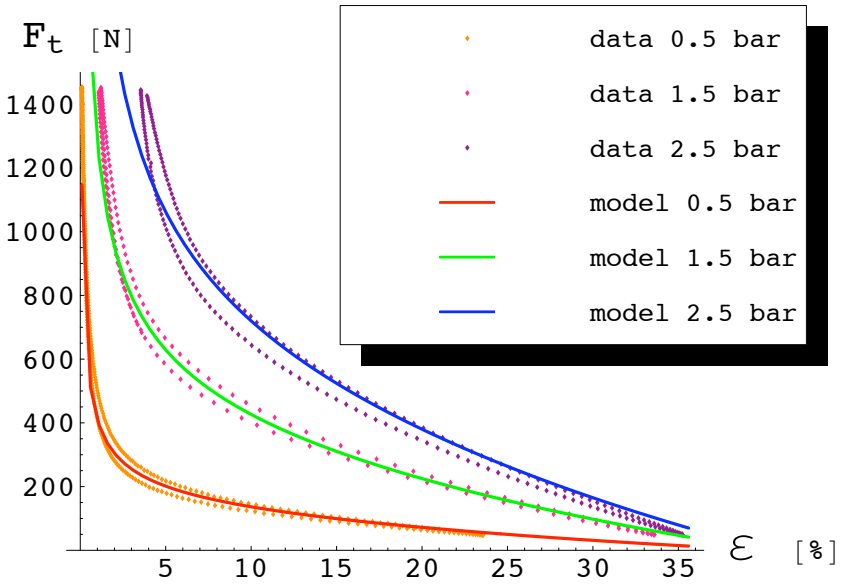


Figuur 4.10: Evolutie van initieel wiskundig model naar optimaal wiskundig model; trekbankdata met $p = 1.5$ bar



Figuur 4.11: Gebruikte data voor de identificatie voor $5\% \leqslant \varepsilon \leqslant 30\%$; $p = 1.5$ bar

niet volledig correct. Dit toont zich in Figuur 4.12. Of deze parameters dan ook werken voor de overige datasets binnen het werkbereik levert bevredigende resultaten op. Dit is te zien in dezelfde figuur. Enkel voor de data bij 2.5 bar is er nog een kleine afwijking. In ieder geval is het huidige wiskundig model een goede benadering van de werkelijkheid.



Figuur 4.12: Het model op de trekbankdata; gebruik van data uit Figuur 4.11 met $p = 1.5$ bar

Het resultaat van de identificatie voor L en R, dat aanleiding geeft tot het optimale model bestaat uit:

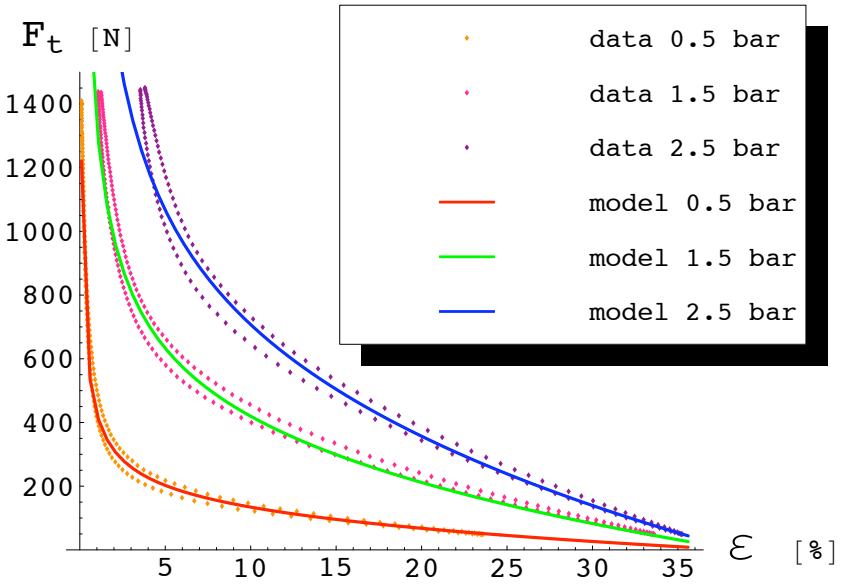
$$\begin{cases} L = 5.86171 \text{ cm} \\ R = 0.97253 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 6.02729$$

Data bij 2.5 bar

Bij data van 2.5 bar kan eveneens dezelfde werkwijze toegepast worden. Er wordt enkel data gebruikt waarbij ε tussen 5% en 30% ligt. Dit leidt uiteindelijk tot volgende waarden voor L , R en de slankheid L/R.

$$\begin{cases} L = 5.67361 \text{ cm} \\ R = 1.08315 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.23809$$

Figuur 4.13 toont het met deze waarden gevonden wiskundig model. Binnen het werkgebied past het model nu beter op de data van 2.5 bar. Ook op de data bij 0.5 bar is het ook nog steeds een goede identificatie. Voor 1.5 bar is het model nu iets minder goed.



Figuur 4.13: Het model op de trekbankdata; gebruik van data waarbij $5\% \leqslant \varepsilon \leqslant 30\%$ met $p = 2.5$ bar

Data bij 0.5 bar

Ook bij de data van 0.5 bar kan eveneens dezelfde werkwijze toegepast worden. Er wordt enkel data gebruikt waarbij ε tussen 5% en 30% ligt. Dit leidt uiteindelijk tot volgende waarden voor L , R en de slankheid L/R.

$$\begin{cases} L = 5.62943 \text{ cm} \\ R = 1.10895 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.07636$$

Figuur 4.14 toont het met deze waarden gevonden wiskundig model. Voor de data van 0.5 bar is het model quasi perfect over de hele range. Ook voor 1.5 bar en 2.5 bar is het model zeker nog aanvaardbaar te noemen.

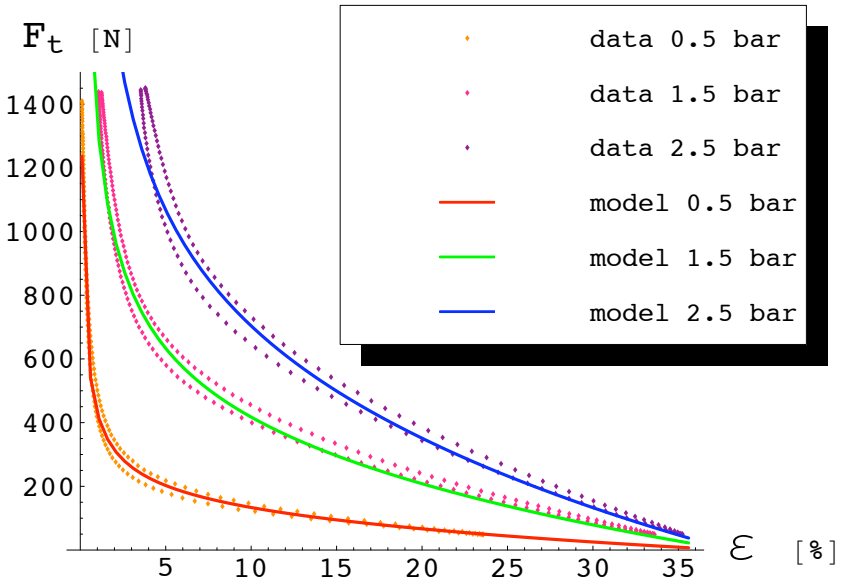
Alle data samen

De gebruikte data voor deze identificatie is een combinatie van de drie datasets. Dit is een goede test om het model te testen bij data waar verschillende drukken aanwezig zijn. Dit zal tevens het geval zijn bij de metingen gedaan op de soft arm in een later stadium.

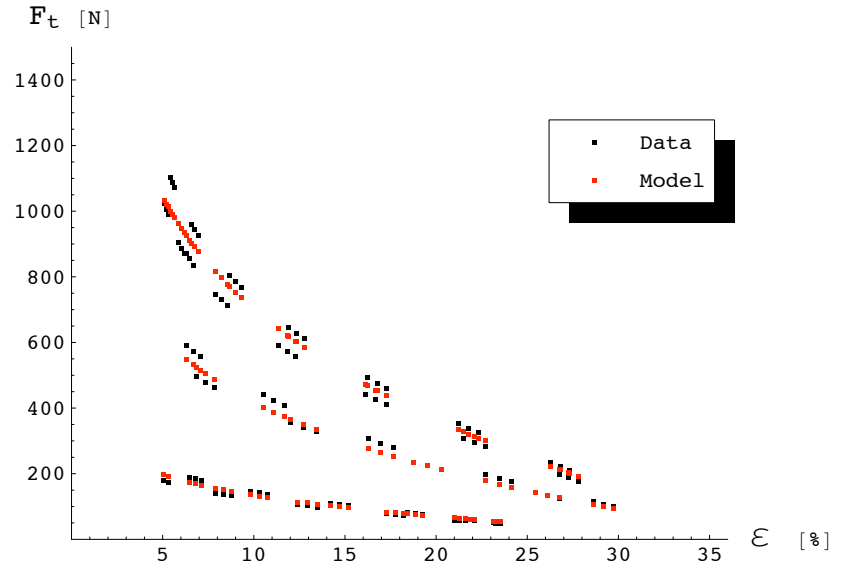
De gebruikte data met het overeenkomstige wiskundig berekende model is te bekijken in Figuur 4.15

De bekomen waarden voor L, R en de slankheid L/R zijn gegeven door:

$$\begin{cases} L = 5.74668 \text{ cm} \\ R = 1.01689 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.34019$$



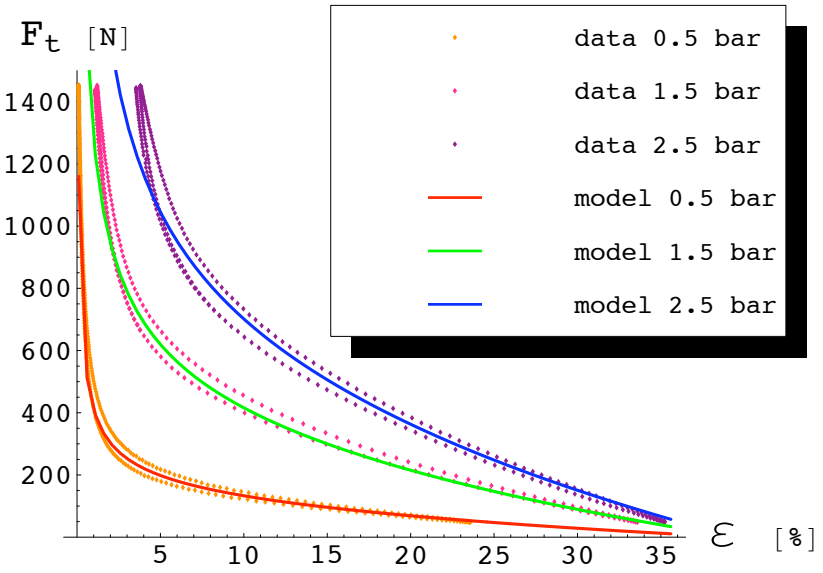
Figuur 4.14: Het model op de trekbankdata; gebruik van data waarbij $5\% \leqslant \varepsilon \leqslant 30\%$ met $p = 0.5$ bar



Figuur 4.15: Gebruikte data en wiskundige benadering; combinatie van data van zowel 0.5 bar, 1.5 bar en 2.5 bar; $5\% \leqslant \varepsilon \leqslant 30\%$

Hiervoor waren 13 iteraties nodig.

In Figuur 4.16 kan bekeken worden hoe het wiskundig model overeenstemt met de hele range van de data. Het is duidelijk een goed resultaat. Het model komt vrij goed overeen met alle data. Dit duidt erop dat de methode geschikt is om te gebruiken bij data van de soft arm.



Figuur 4.16: Het model op de trekbankdata; gebruik van alle data waarbij $5\% \leqslant \varepsilon \leqslant 30\%$

4.4.3 Conclusie

De berekening van L en R is gebeurd met data van een echte spier, opgenomen met een trekbank. De metingen zijn gebeurd bij 0.5 bar, 1.5 bar en 2.5 bar. Berekeningen zijn afzonderlijk gebeurd voor elk van deze drukken, en daarna een combinatie van de drie drukken samen. De conclusie is dat het wiskundig model en de gevolgde methode voor de identificatie te gebruiken is binnen een werkgebied van ε tussen 5% en 30 %. Voor kleinere ε zijn krachten te hoog. Dan is het model niet meer geldig omwille van het elastische gedrag van de materialen. Daar kan het model het systeem niet meer correct beschrijven.

Het komen tot een voldoende nauwkeurig resultaat voor L en R duurt echter lang. Het is dus niet mogelijk veel meetpunten te gebruiken. Het is dan ook nodig een nieuwe snellere methode te ontwikkelen.

Hoofdstuk 5

Iteratieve gradiëntmethode

De vorige getrachte methode blijkt veel te traag te zijn en is dus onaanvaardbaar. Een veel gebruikte methode is dan ook de gradiëntmethode [8]. In plaats van telkens het minimum te zoeken in 1 dimensie zal met deze methode een richtingsvector gezocht worden die wijst in de richting van de sterkste daling van de kostfunctie. Een vaste stap zal dan genomen worden in deze richting. Dit proces wordt dan iteratief herhaald.

Indien de de stap tegroot blijkt – dit wil zeggen als de kostfunctie niet gedaald is en dus de stap voorbij het minimum komt – zal deze verkleind worden. Zo kan ook het minimum van de kostfunctie gevonden worden.

Zoals zal aangetoond worden duurt een iteratie met de gradiëntmethode veel korter dan bij de vorige methode. Wel gebeurt de convergentie met deze methode redelijk traag. Er zijn met andere woorden zeer veel iteraties nodig.

5.1 Algoritme

Deze methode gebruikt de gradiënt van de kostfunctie V. De kostfunctie V was gedefinieerd als:

$$V = \frac{1}{2}\sum_{k=1}^{N} e^2(k,\theta) \tag{5.1}$$

$$= \frac{1}{2}\sum_{k=1}^{N}\left(F_t(\theta)\Big|_{\substack{\varepsilon(k)\\p(k)}} - F_{\text{data}}(k)\right)^2 \tag{5.2}$$

waarbij

$$e(k,\theta) = F_t(\theta)\Big|_{\substack{\varepsilon(k)\\p(k)}} - F_{\text{data}}(k) \tag{5.3}$$

De gradiënt van V wordt dus gegeven door:

$$\nabla V = \left(\frac{\partial V}{\partial L}\,,\ \frac{\partial V}{\partial R}\right)^T \tag{5.4}$$

Invullen van vergelijking (5.2) en uitwerken geeft:

$$\nabla V = \left(\sum_{k=1}^{N} \left[\left(F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} - F_{\text{data}}(k) \right) \frac{\partial F_t(\theta)}{\partial L} \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right] , \right.$$
$$\left. \sum_{k=1}^{N} \left[\left(F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} - F_{\text{data}}(k) \right) \frac{\partial F_t(\theta)}{\partial R} \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right] \right)^T \quad (5.5)$$

Gebruik makend van vergelijking (5.3) kan dit korter geschreven worden.

$$\nabla V = \left(\sum_{k=1}^{N} \left[e(k,\theta) \frac{\partial F_t(\theta)}{\partial L} \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right] , \sum_{k=1}^{N} \left[e(k,\theta) \frac{\partial F_t(\theta)}{\partial R} \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right] \right)^T \quad (5.6)$$

$$= \sum_{k=1}^{N} \left[\left(\frac{\partial F_t(\theta)}{\partial L} \Big|_{\substack{\varepsilon(k) \\ p(k)}} , \frac{\partial F_t(\theta)}{\partial R} \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right)^T e(k,\theta) \right] \quad (5.7)$$

Dit kan onder vectornotatie geschreven worden:

$$\nabla V = J(\theta)^T \cdot e(\theta) \quad (5.8)$$

met $J(\theta)$ de Jacobiaan van de spierfunctie F_t .

De gradiënt van de kostfunctie berekenen komt er dus op neer de jacobiaan van F_t te berekenen. Deze kan analoog berekend worden als in Hoofdstuk 3. J is uiteraard afhankelijk van L en R (en dus θ). Bij elke nieuwe iteratie zal dus zowel $e(\theta)$ als $J(\theta)$ opnieuw berekend moeten worden.

De nieuwe waarde van $\theta = \begin{pmatrix} L \\ R \end{pmatrix}$ wordt gevonden door een lineaire stap te nemen gebruikmakend van "∇V" – de gradiënt van V. De grootte van de stap wordt bepaald door de variabele δ.

$$\theta_{i+1} = \theta_i - \delta \frac{\nabla V}{\|\nabla V\|} \quad (5.9)$$

Indien de kostfunctie V_{i+1}, gevonden met de nieuwe waarde θ_{i+1} groter is dan de kostfunctie overeenstemmend met de oude θ_i, is de stap δ te groot gekozen, en zal een nieuwe θ_{i+1} berekend worden met een kleinere δ. Dit kan zoveel herhaald worden tot $V_{i+1} < V_i$.

Op deze manier is dan een nieuwe θ bekomen, waarmee opnieuw de gradiënt van V kan berekend worden in een volgende iteratie.

Er hoeft dus slechts 1 maal een gradiënt berekend worden per iteratie. Dit is het gedeelte van de iteratie dat het langste duurt. In de vorige methode werd deze ongeveer 15 maal berekend. Elke iteratie duurt dus gemiddeld 15 maal minder lang. Dit zou een hele verbetering moeten geven.

5.1.1 Convergentiecriterium

Als convergentiecriterium is nu een iets intelligentere methode gebruikt. In plaats van te kijken of de kostfunctie nog veel verandert wordt nu gekeken of de parameters nog

veel veranderen. De conditie waarbij de iteratie stopt wordt gegeven via de volgende vergelijking.

$$\left|\frac{\theta_i - \theta_{i-1}}{\theta_{i-1}}\right| < 0.0001 \tag{5.10}$$

Dit wordt uitgerekend voor L en voor R. Pas als voor beide parameters aan dit criterium voldaan is stoppen de iteraties.

5.2 Toepassen op theoretisch model met ruis

Zoals voor de eerst gebruikte methode is het best om te controleren of de gradiëntmethode op een perfect model met ruis ook werkt. Als dit werkt, zal het gebruikt worden op trekbankdata.

De gebruikte ruis is precies dezelfde als die gebruikt voor de eerste methode (zie Figuur 4.1). De druk is 1.5 bar, L = 6 cm, en R = 1 cm. De oplossing zou dus dezelfde moeten zijn als voor de eerste methode.

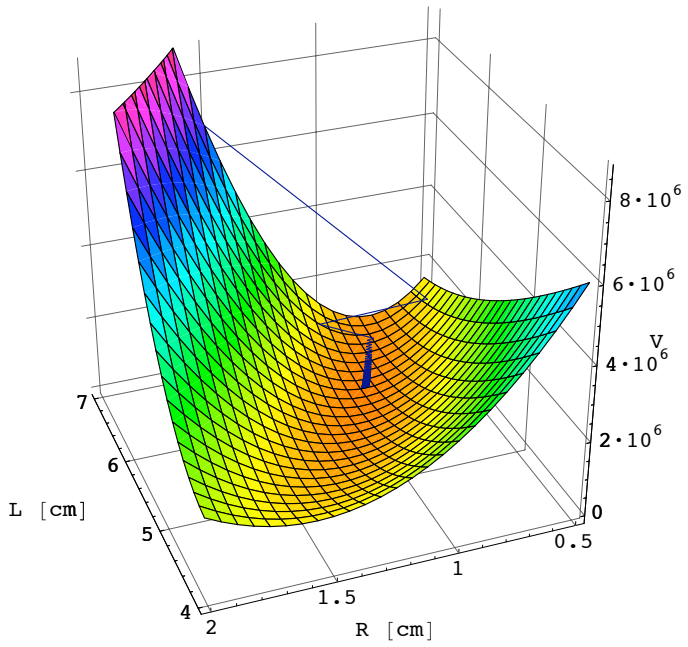
5.2.1 Identificatie

Aangezien de gebruikte data dezelfde is als de vorige methode, zal ook de kostfunctie dezelfde zijn (zie Figuur 4.2). Wanneer het algoritme toegepast wordt op deze functie kan de convergentie gevolgd worden. Resultaten worden getoond in Tabel 5.1. Als beginwaarde voor de stap δ is 1cm gekozen.

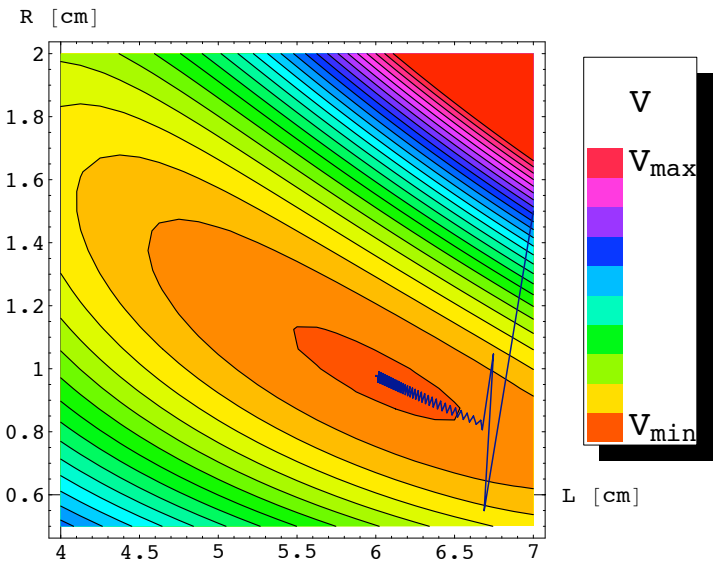
iteratie	L [cm]	R [cm]	V	L/R
0	7	1.5	6735771.9	4.667
6	6.61990	0.84975	532467.9	7.790
12	6.50637	0.87535	509307.9	7.433
24	6.33340	0.91497	483188.0	6.922
36	6.21648	0.94232	471738.8	6.597
42	6.17386	0.95242	468802.3	6.482
60	6.08977	0.97253	464941.3	6.262
78	6.04702	0.98285	463961.4	6.153
96	6.02572	0.98801	463720.5	6.099
102	6.02137	0.98907	463691.6	6.088
108	6.00113	0.97795	461469.7	6.136
114	6.00173	0.97790	461469.4	6.137
120	6.00209	0.97781	461469.2	6.138
126	6.00238	0.97774	461469.1	6.139
132	6.00260	0.97769	461469.0	6.140
144	6.00292	0.97761	461468.9	6.140
150	6.00303	0.97759	461468.9	6.141
155	6.00317	0.97742	461468.7	6.142

Tabel 5.1: slankheid L/R, V, L en R na enkele iteraties via de gradiëntmethode toegepast op ruismodel

Deze resultaten zijn grafisch weergegeven in Figuur 5.1. Het uiteindelijke resultaat



(a) 3D plot



(b) Contourplot

Figuur 5.1: Grafische weergave van het verloop van de gevonden waarden van L, R en V voor de opeenvolgende iteraties

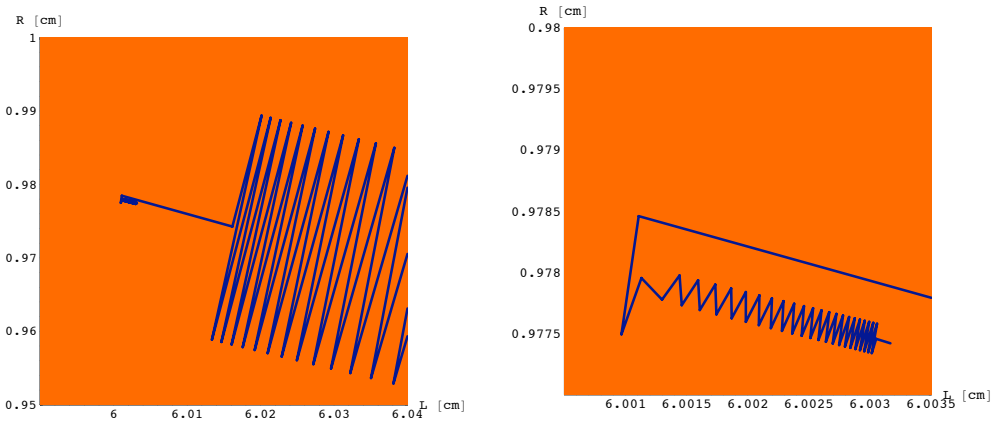
is nu na 155 iteraties:

$$\begin{cases} L = 6.00317 \text{ cm} \\ R = 0.97742 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 6.14185$$

Dit resultaat is bijna hetzelfde als de eerste methode. Opmerkelijk is dat hier 155 iteraties nodig waren om te convergeren. Dit komt omdat de kostfunctie een heel kleine gradiënt bezit over een groot gebied. Dit is ook in Figuur 5.1 te zien aan de zeer lange bijna vlakke vallei. De methode blijft vastzitten in deze vallei door telkens de sterkste daling te volgen en een stap in die richting te nemen. Deze komt niet overeen met de richting waarin de vallei verloopt. Het resultaat is daarom een grillig heen en weer stappen in plaats van recht naar het doel.

155 iteraties is een enorm aantal. In de eerste methode waren slechts 38 iteraties nodig. De eerste methode gaat sneller op zijn doel af. Hij zal telkens in het midden van de vallei staan bij een vaste waarde van een parameter en proberen in 1 dimensie naar een volgend minimum te gaan. Daar was de reden dat het lang duurde om aan het convergentiecriterium te voldoen omdat de vallei niet volgens 1 van de richtingen van L of R was gericht maar ongeveer 45° hierop.

Bij de gradiëntmethode speelt de hoek ten opzichte van de dimensies van L en R geen rol. Als men zich hier bevindt in de bedding van de vallei, zal de gradiënt wijzen in de richting van het verdere verloop van de vallei. Of er ongeveer hier beland wordt hangt enkel af van geluk. Als de stap δ te groot is zal de stap telkens over de bedding heen stappen. Bij de laatste iteraties heeft de methode de bedding van de vallei ongeveer gevonden en gaat hij plots snel op zijn doel af. Dit is niet erg zichtbaar op Figuur 5.1, enkel als men wat inzoomt. Dit is te zien in Figuur 5.2.

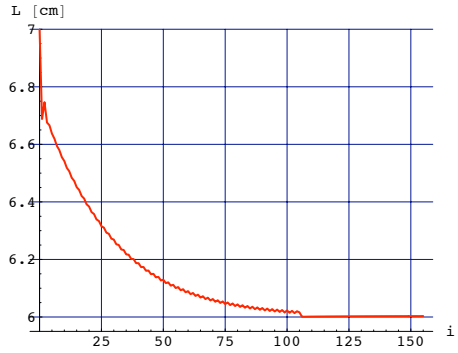


(a) zoom 1 (b) zoom 2

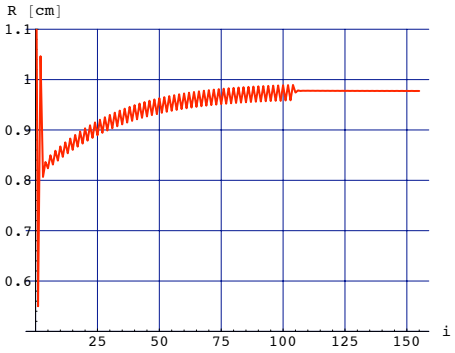
Figuur 5.2: Uitvergroting voor laatste iteraties uit Figuur 5.1

5.2.2 Convergentie

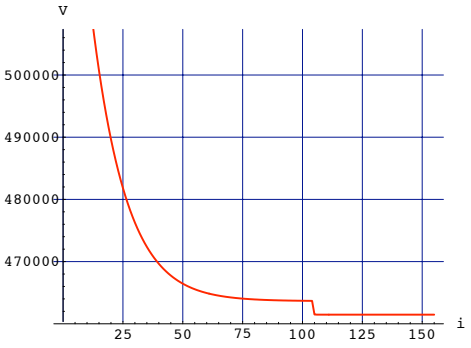
In de convergentie vertaalt zich dat in Figuur 5.3. Hier zien we dat we eigenlijk al van convergentie kunnen spreken na ongeveer 110 iteraties. De nauwkeurigheid hierna speelt zich af op duizendsten van centimeters.



(a) L



(b) R



(c) V

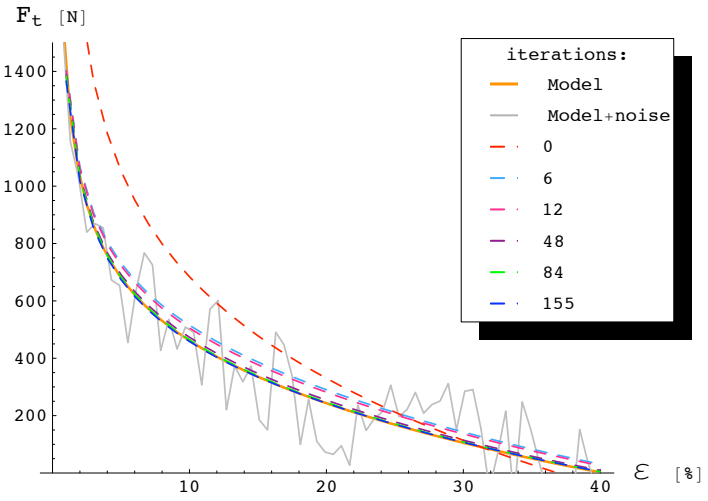
Figuur 5.3: L, R en V in functie van het aantal iteraties, voor het ruismodel met gradiëntmethode

In Tabel 5.1 is ook te zien dat na ongeveer 110 iteraties de verschillen voor L en R zich afspelen op duizendsten van centimeters. Naargelang de gewenste nauwkeurigheid kan men hier dus al beslissen of men al dan niet kan spreken over een geconvergeerde waarde.

5.2.3 Controle

Om nu uiteindelijk te zien of het minimum van de kostfunctie V overeen komt met een wiskundig model dat perfect ligt op het oorspronkelijke theoretische model waarop de ruis gesuperponeerd is, worden deze nog met elkaar vergeleken in Figuur 5.4. De evolutie van het wiskundig model naar het oorspronkelijk theoretisch model is ook weergegeven voor enkele iteraties.

In rekentijd is er een groot verschil vastgesteld tussen de gradiëntmethode en de voorgaande methode, ondanks het feit dat er veel meer iteraties nodig waren voor



Figuur 5.4: Evolutie van F_t naar het theoretische model waarop ruis gesuperponeerd is

de gradiëntmethode. De tijd nodig voor de eerste methode was nog ongeveer 4 maal langer. Dit komt doordat de tijd voor een iteratie in de voorgaande methode ongeveer 15 maal langer is.

5.2.4 Conclusie

Bij toepassing van de gradiëntmethode op ruis werkt de methode eveneens, maar zijn er toch veel iteraties nodig. De winst in tijd is in dit geval slechts een factor 4. Dit komt door de vorm van de kostfunctie. We kunnen besluiten dat de convergentie met de gradiëntmethode erg traag gebeurt, maar in ieder geval sneller (in rekentijd) dan de eerste methode.

5.3 Toepassen op trekbankdata

Om de methode nu aan een reëlere test te onderwerpen wordt de trekbankdata gebruikt – zie Figuur 4.7. Er wordt onmiddellijk gebruik gemaakt van de data $5\% \leqslant \varepsilon \leqslant 30\%$. Deze zorgde bij de vorige methode voor de beste benadering. Dit zou dus ook hier het geval moeten zijn.

Eerst zal de test uitgevoerd worden bij data van 1.5 bar. Daarna met een combinatie van de drie drukken (0.5 bar, 1.5 bar en 2.5 bar) waarbij de test gebeurd is.

5.3.1 Data bij 1.5 bar

De gebruikte data voor de identificatie is dezelfde als deze die gebruikt was bij de eerste methode (zie Figuur 4.11).

Toepassing van het algoritme op deze data levert resultaten die weergegeven worden in Tabel 5.2 evenals in Figuur 5.5 en Figuur 5.6.

iteratie	L [cm]	R [cm]	V	L/R
0	7	1.5	1587396.1	4.667
1	6.29728	0.78854	40788.2	7.986
2	6.19857	0.71184	37901.9	8.708
3	6.22108	0.83480	35512.4	7.452
4	6.12495	0.75490	33379.2	8.114
6	6.06204	0.79278	30215.3	7.647
8	6.01001	0.82496	28130.0	7.285
16	5.89336	0.90054	25409.0	6.544
30	5.89293	0.97573	20784.1	6.040
36	5.89349	0.97563	20784.0	6.041
37	5.89370	0.97552	20784.0	6.042
38	5.89369	0.97549	20784.0	6.042

Tabel 5.2: slankheid L/R, V, L en R na enkele iteraties via de gradiëntmethode toegepast op trekbankdata bij 1.5 bar

Het uiteindelijke resultaat is nu na 38 iteraties:

$$\begin{cases} L = 5.89369 \text{ cm} \\ R = 0.975487 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 6.04179$$

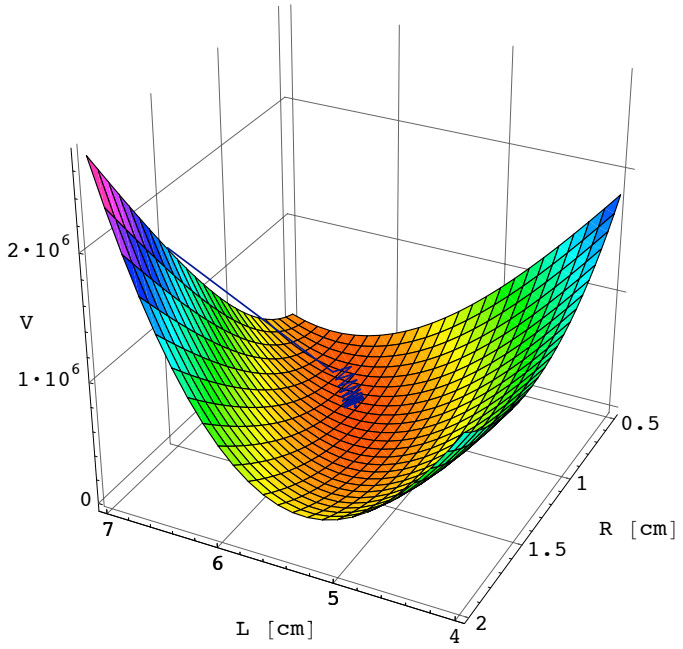
Dit is een vergelijkbaar resultaat als gevonden met de eerste methode. Deze oplossing is echter nog nauwkeuriger omdat het toegepaste convergentiecriterium hier tot een grotere nauwkeurigheid gebiedt.

Convergentie

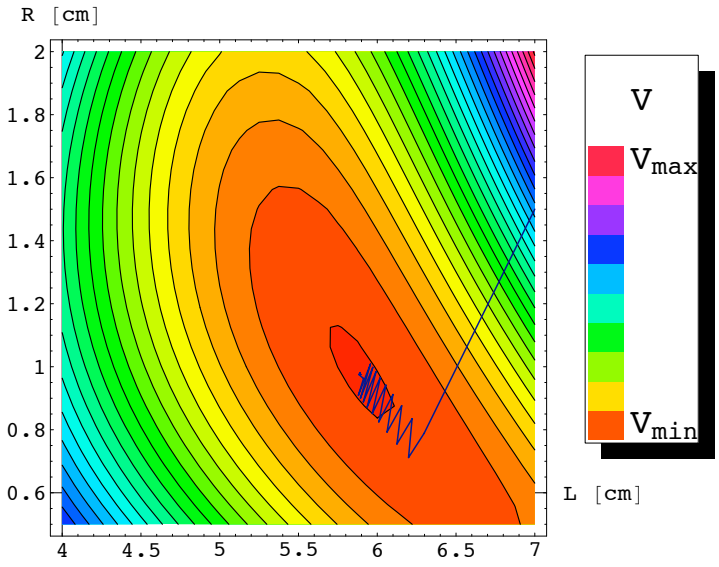
Het algoritme toepassen op deze data voldeed aan het convergentiecriterium na 38 iteraties. Dit is de helft van het aantal iteraties bij ruis. Het is opmerkelijk dat er veel minder iteraties nodig waren dan voor de toepassing op het theoretisch model met ruis (155 iteraties). Bij de vorige methode was dit net andersom: er waren meer iteraties nodig voor trekbankdata (76 iteraties) dan voor het ruismodel (38 iteraties). Dit toont aan dat er nu zeer snel in de bedding van de vallei terecht gekomen is (zie Figuur 5.5). Dit heeft veel te maken met geluk enerzijds en de vorm van de kostfunctie anderzijds. Voor deze trekbank-data is de vallei vrij uitgesmeerd en zal de gradiënt beter naar het globale minimum wijzen. Bij een sferische vorm zou de gradiënt steeds het minimum aanwijzen en zou de convergentie het snelst gaan.

De rekentijd valt hier heel goed mee. Op een vijftal minuten was de identificatie gebeurd. Dit in vergelijking met tweetal uur bij de vorige methode. Daarenboven is hier nog veel nauwkeuriger gegaan, wat de rekentijd hier nog opdreef ten opzichte van de vorige methode, waar dit niet gebeurde.

Aan Figuur 5.6 is te zien dat men al van convergentie kan spreken na een 20tal iteraties. Al de hierop volgende iteraties zijn voor extra nauwkeurigheid moest dit gewenst zijn.

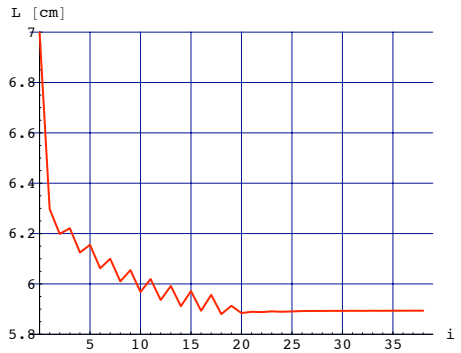


(a) 3D plot

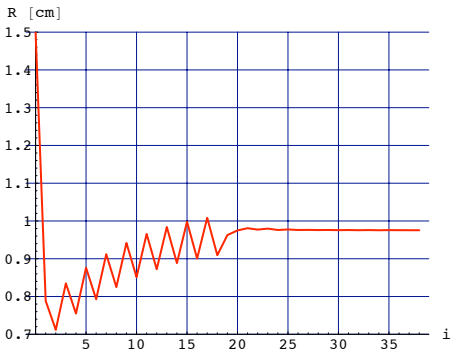


(b) Contourplot

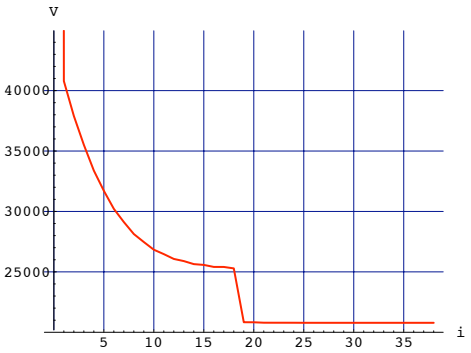
Figuur 5.5: Grafische weergave van het verloop van de gevonden waarden van L, R en V voor de opeenvolgende iteraties bij gradiëntmethode voor trekbankdata bij 1.5 bar



(a) L



(b) R



(c) V

Figuur 5.6: L, R en V in functie van het aantal iteraties, voor trekbankdata bij 1.5 bar met gradiëntmethode

Controle

Het valt te verwachten dat het gevonden wiskundige model de trekbankdata bij 1.5 bar goed zal volgen in het werkingsgebied van de softarm. Hoe goed deze elkaar benaderen is aangetoond in Figuur 5.7.

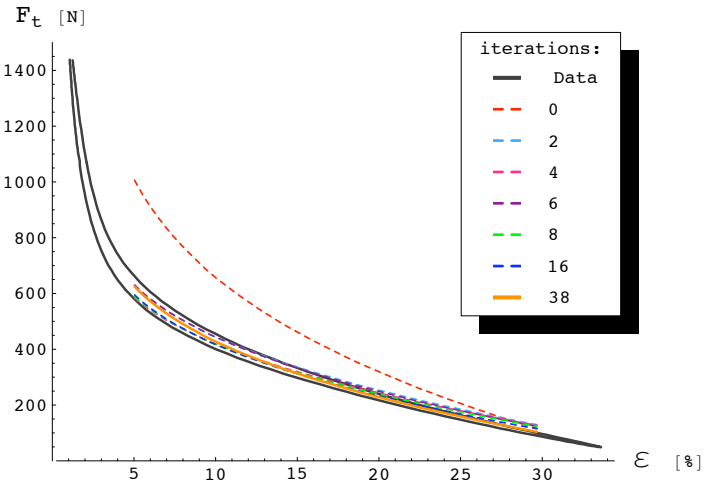
De convergentie gebeurt in dit geval met deze methode redelijk snel, en levert tevens goede resultaten. Het wiskundig model F_t ligt binnen dit werkbereik quasi perfect tussen de hysteresislus.

Het gebruik voor dit geval is dus een hele verbetering ten opzichte van de vorige methode.

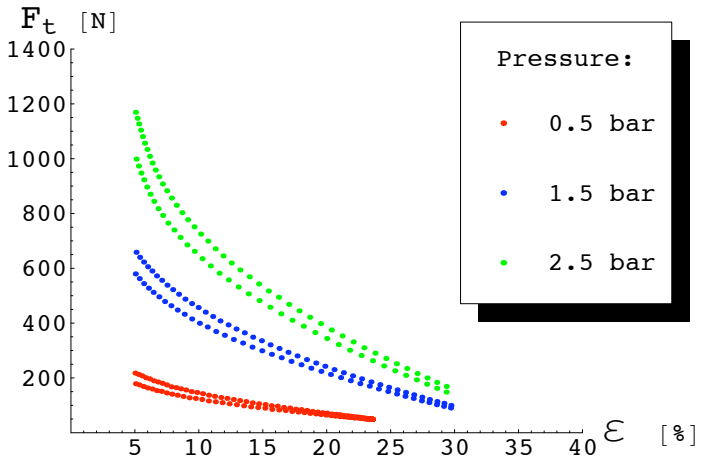
5.3.2 Data bij alle drukken samen

In deze paragraaf wordt getest of de methode enerzijds werkt met meerdere drukken samen, en anderzijds met een groter aantal punten, en wat het effect is op de rekentijd.

Er zijn nu 271 punten gebruikt, verdeeld over 3 drukken. Deze zijn te zien in Figuur 5.8.



Figuur 5.7: Evolutie van F_t naar de best benaderende oplossing via de gradiëntmethode, voor trekbankdata bij 1.5 bar



Figuur 5.8: De gebruikte datapunten van de trekbank voor de identificatie met de gradiëntmethode

Identificatie

De identificatie verloopt verder analoog als bij 1.5 bar. Voor de duidelijkheid van de studie volgen hier nog een tabel en figuren met het uiteindelijke resultaat (Tabel 5.3, Figuur 5.9 en Figuur 5.10). Het resultaat wordt bekomen na 39 iteraties. De waarden voor L, R en de slankheid L/R zijn dan gegeven door:

$$\begin{cases} L = 5.76029 \text{ cm} \\ R = 1.04441 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.51535$$

Dit is wederom quasi hetzelfde resultaat als bekomen met de vorige methode. Het aantal gebruikte punten ligt hier veel hoger en dus zal deze uitkomst beter overeenstemmen met de werkelijkheid.

iteratie	L [cm]	R [cm]	V	L/R
0	7	1.5	7680553.8	4.667
1	6.31733	0.76927	246262.2	8.212
2	6.22038	0.69037	227881.0	9.010
3	6.23511	0.81450	216333.6	7.655
4	6.13863	0.73502	201980.5	8.352
6	6.06434	0.77735	181105.3	7.801
8	5.99812	0.81644	165017.8	7.347
16	5.81814	0.92972	135811.2	6.258
30	5.75752	1.04517	111620.7	5.509
37	5.76029	1.04451	111613.1	5.515
38	5.76023	1.04440	111613.1	5.515
39	5.76029	1.04441	111613.1	5.515

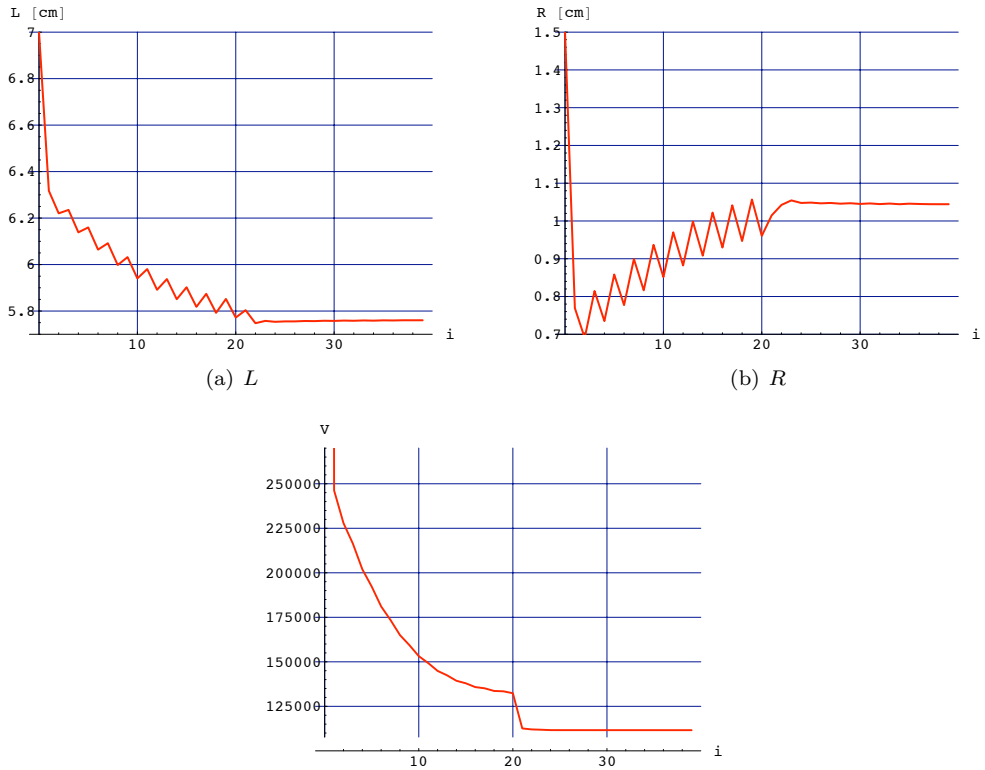
Tabel 5.3: slankheid L/R, V, L en R na enkele iteraties via de gradiëntmethode toegepast op trekbankdata bij 3 verschillende drukken

Convergentie

Bij het kijken naar Figuur 5.9 is duidelijk dat reeds na ongeveer 25 iteraties de waarden van L, R en V nog nauwelijks wijzigen. Er zijn dus slechts een 25tal iteraties nodig om tot een goed resultaat te komen. Dit is ongeveer het dubbel van het aantal iteraties nodig bij de vorige methode. De rekentijd is echter veel korter. Zelfs met het in beschouwing nemen van een veelvoud van de gebruikte datapunten.

Controle

Om te controleren of de gevonden oplossing wel voldoet aan alledrie datasets kan men Figuur 5.10 bekijken. Hier is duidelijk dat voor het hele werkbereik van de softarm het geupdate wiskundig model goed overeenstemt met de gemeten data. Enkel voor de druk bij 2.5 bar (hoogste dataset in de figuur) is er een kleine afwijking zichtbaar bij hoge kracht. Deze is echter aanvaardbaar. In werkelijkheid zal het practisch nooit gebeuren dat zo een kracht hoeft opgewekt te worden.



(a) L (b) R

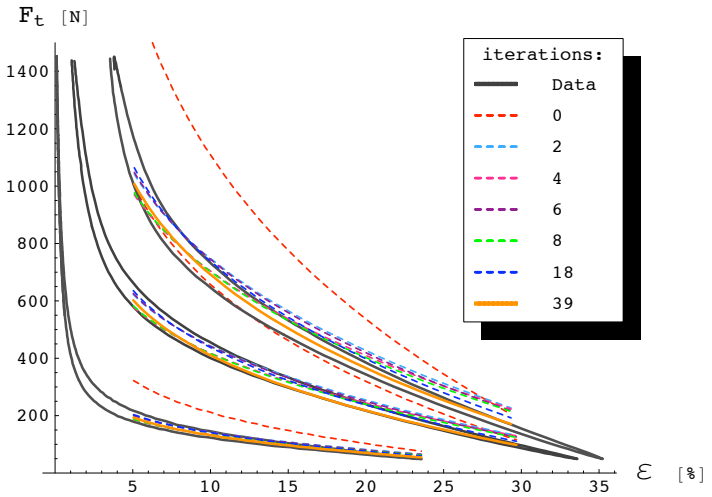
(c) V

Figuur 5.9: L, R en V in functie van het aantal iteraties, voor trekbankdata bij 3 verschillende drukken met gradiëntmethode

5.4 Conclusie

De gradiëntmethode is duidelijk een snellere methode dan de vorige ééndimensionale methode uit Hoofdstuk 4.1. De tijdswinst hangt af van het aantal nodige iteraties. Het kan gebeuren dat er veel meer iteraties nodig zijn dan voor de ééndimensionale methode of net omgekeerd. Dit hangt af van geluk en de vorm van de kostfunctie. In ieder geval is het sneller en laat deze methode dus beter toe om meer datapunten te gebruiken.

Toch is de convergentie nog niet naar tevredenheid. Het hangt teveel af van de vorm en geluk. Dit is vooral van belang bij een groot aantal datapunten. Hoe minder iteraties nodig zijn, hoe beter. Daarom is er op zoek gegaan naar een betere methode.



Figuur 5.10: Evolutie van F_t naar de best benaderende oplossing via de gradiëntmethode, voor trekbankdata bij 3 verschillende drukken (0.5 bar, 1.5 bar, 2.5 bar)

Hoofdstuk 6

Hogere orde methoden

6.1 Inleiding

Het is duidelijk dat de gradiëntmethode nog steeds niet volledig naar wens werkt. De bedoeling is om een identificatie te krijgen in een redelijke rekentijd (maximaal 10 minuten). Zo hoeft men geen dag te wachten om 4 artificiële spieren te identificeren. De noodzaak om een snellere methode te ontwikkelen is dus nog steeds aanwezig.

De gradiëntmethode is een methode die in feite iteratief de kostfunctie lineariseert in het huidige punt en dan een stap neemt volgens deze lineaire functie in de goede richting. Dit is voldoende indien het systeem lineair is in de parameters. Dit is hier niet het geval.

Een betere benadering van de kostfunctie bestaat uit een hogere-orde-functie [8].

6.2 Newton-Raphson methode

Hogere-orde-functies benaderen de kostfunctie door zijn Taylor-expansie, waarbij termen tot een gewenste orde behouden blijven, en termen hoger dan de gewenste orde verwaarloosd worden. Voor een 2de orde benadering is dit bijvoorbeeld:

$$V(\theta+\delta) = V(\theta) + \nabla V(\theta)\ \ \delta + \frac{1}{2}\,\delta^T\ \nabla^2 V(\theta)\ \delta + \cdots \tag{6.1}$$

De termen na de tweede orde zijn weggelaten. De term $\nabla^2 V(\theta)$ noemt men ook wel de *Hessiaan*. Deze is een matrix van volgende gedaante:

$$\nabla^2 V(\theta) = \begin{pmatrix} \frac{\partial^2 V}{\partial {\theta_1}^2} & \frac{\partial^2 V}{\partial \theta_1 \partial \theta_2} \\ \frac{\partial^2 V}{\partial \theta_2\, \partial \theta_1} & \frac{\partial^2 V}{\partial {\theta_2}^2} \end{pmatrix} \tag{6.2}$$

Zoals bij de hiervoor besproken methodes bestaat ook de Newton-Raphson methode erin om de afgeleide van de kostfunctie gelijk aan nul te stellen:

$$\frac{\partial}{\partial \delta} V(\theta+\delta) = 0 + \nabla V(\theta) + \nabla^2 V(\theta)\ \delta \quad \equiv \quad 0 \tag{6.3}$$

Een tweede-orde benadering voor δ kan men alzo schrijven als:

$$\delta = -\left(\nabla^2 V(\theta)\right)^{-1}\ \nabla V(\theta) \tag{6.4}$$

Indien de kostfunctie V perfect kwadratisch is zal een stap δ leiden tot het minimum van de kostfunctie.

$$\theta_{i+1} = \theta_i + \delta \tag{6.5}$$

Dit is echter niet zo. Gezien de kwadratische natuur van de kostfunctie zal dit echter al zeer dicht bij het minimum liggen.

Iteratief herhalen van het berekenen van een nieuwe δ en daaruit een nieuwe θ leidt snel tot het minimum van de kostfunctie en dus eveneens de optimale waarde voor θ.

6.2.1 Problemen

Het probleem bij deze methode bevindt zich net in die hogere orde, namelijk de berekening van de hessiaan $\nabla^2 V(\theta)$.

Voor de gradiënt van V werd reeds een lange formule analytisch opgesteld – zie vergelijking (5.7) uit Hoofdstuk 4.1, die gebruik maakt van vergelijkingen (3.14) en (3.17) uit Hoofdstuk 3. De analytische formule voor de hessiaan zal erg veel rekenwerk vragen van de computer. Dit is dus al een groot nadeel van deze methode.

Vervolgens is er nog een nadeel: convergentie is niet gegarandeerd, indien de hessiaan niet positief definiet is. In plaats van convergentie kan er dan net divergentie optreden [8].

Deze problemen maken dat deze methode niet geprobeerd is op dit model. In plaats daarvan is er op zoek gegaan naar nog een andere methode.

6.3 Gauss-Newton methode

De Gauss-Newton methode lost een van de problemen van de Newton-Raphson problemen op – namelijk de berekening van de hessiaan $\nabla^2 V(\theta)$. Deze methode zal namelijk een benadering doorvoeren voor de hessiaan. Dit door gebruik te maken van de kwadratische natuur van de kostfunctie. De kleinste kwadraten formulatie voor de kostfunctie was gegeven door:

$$V = \frac{1}{2} \sum_{k=1}^{N} \left(F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} - F_{\text{data}}(k) \right)^2 \tag{6.6}$$

De tweede orde afgeleide hiervan wordt dan gegeven door:

$$\begin{aligned} \frac{\partial^2 V(\theta)}{\partial \theta^{\,2}} = \sum_{k=1}^{N} \Bigg(& \left(\frac{\partial}{\partial \theta} F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right)^T \left(\frac{\partial}{\partial \theta} F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right) \\ & + \left(F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} - F_{\text{data}}(k) \right) \frac{\partial^2}{\partial \theta^{\,2}} F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} \Bigg) \end{aligned} \tag{6.7}$$

Als de tweede term klein wordt kan deze verwaarloosd worden in vergelijking (6.7) en dus kan vergelijking (6.7) benaderd worden door:

$$\frac{\partial^2 V(\theta)}{\partial \theta^{\,2}} \approx \sum_{k=1}^{N} \left(\left(\frac{\partial}{\partial \theta} F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right)^T \left(\frac{\partial}{\partial \theta} F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} \right) \right) \tag{6.8}$$

Dit gebeurt als $\left(F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} - F_{\text{data}}(k) \right)$ klein wordt.

Zo kan de hessiaan benaderd worden door:

$$\nabla^2 V(\theta) \approx J^T J \tag{6.9}$$

met J de Jacobiaan van F_t .

Gebruik makend van de Jacobiaan kan ook de gradiënt anders geschreven worden als (zie paragraaf 5.1):

$$\nabla V = J(\theta)^T \cdot e(\theta) \tag{6.10}$$

Vergelijking (6.10) en vergelijking (6.9) toepassen op vergelijking (6.4) uit de Newton-Raphson methode levert volgende formulatie voor δ:

$$\delta = \left(J^T J \right)^{-1} J^T e \tag{6.11}$$

Dit kan weer iteratief toegepast worden door telkens iteratief δ te berekenen en in te vullen in

$$\theta_{i+1} = \theta_i + \delta \tag{6.12}$$

6.3.1 Bespreking

Door toepassen van deze methode gebruikt men minder rekentijd dan bij de Newton-Raphson methode enerzijds, en hoeft men geen uitdrukking voor de hessiaan te berekenen anderzijds. Dit lost het probleem op van de moeilijk te vinden uitdrukking voor de hessiaan in dit geval.

Het gedrag is erg gelijkaardig met de Newton-Raphson methode. Door de benadering echter zal de convergentie iets trager verlopen. De hessiaan kon in de Newton-Raphson methode niet positief definiet zijn. Door de benadering door te voeren is deze nu wel positief definiet. Enkel in singuliere punten en zadelpunten is convergentie nog niet gegarandeerd [8].

In zadelpunten wordt de matrix $(J^T J)$ namelijk singulier en kan deze dus niet geïnverteerd worden in vergelijking (6.11). Het kan ook gebeuren dat de kostfunctie zelfs stijgt.

6.3.2 Toepassing

Deze methode werd eerst geprobeerd en gaf op de trekbankdata veel foutmeldingen. Ook steeg de kostfunctie bij enkele iteraties. Dit zal waarschijnlijk te wijten zijn aan de oorzaken vermeld in paragraaf 6.3.1.

Een nog andere methode dient zich dus aan.

6.4 Levenberg-Marquardt methode

De Levenberg-Marquardt methode [8] combineert de reeds uitgewerkte gradientmethode (Hoofdstuk 5), en de Gauss-Newton methode. Dit gebeurt door de Levenberg-Marquardt term toe te voegen in vergelijking (6.11) die δ beschrijft voor de Gauss-Newton methode:

$$\delta = \left(J^T J + \lambda I_{n_\theta}\right)^{-1} J^T e \tag{6.13}$$

Hier is λI_{n_θ} de Levenberg-Marquardt term die ervoor zorgt dat de methode begint als een gradiëntmethode en overgaat naar een Gauss-Newton methode naarmate λ kleiner wordt bij elke iteratie. De Levenberg Marquardt methode heeft als limiet de gradiëntmethode als $\lambda \to \infty$, en als limiet de Gauss-Newton methode als $\lambda \to 0$.

$$\lim_{\lambda \to \infty} \delta = \frac{1}{\lambda} J^T e \tag{6.14}$$

$$\lim_{\lambda \to 0} \delta = \left(J^T J\right)^{-1} J^T e \tag{6.15}$$

Dit brengt enkele voordelen met zich mee:

- De Levenberg-Marquardt term zal de matrix $(J^T J)$ beter conditioneren in de buurt van zadelpunten en andere punten waar de matrix singulier wordt. Er zullen dan minder numerieke fouten aanwezig zijn in de oplossing. Met betere conditionering wordt bedoeld dat het conditiegetal κ klein is ($\kappa \approx 1$).

$$\kappa = \frac{\text{grootste eigenwaarde}}{\text{kleinste eigenwaarde}} \tag{6.16}$$

- Indien de berekende stap δ te groot blijkt te zijn zal de oplossing eveneens divergeren bij de Gauss-Newton methode. Aan vergelijking (6.14) te zien zal een stijging van de factor λ een zodanige invloed hebben op de stap δ dat deze afneemt.

Het gebruik van deze methode garandeert dus dat er steeds convergentie zal optreden. Deze methode implementeren lijkt zodoende een goed idee.

6.4.1 Algoritme

De implementatie van de Levenberg-Marquardt methode gebeurt als volgt:

1. Een beginwaarde wordt gekozen voor L en R.
   $\theta_0 = \begin{pmatrix} L_0 \\ R_0 \end{pmatrix}$
   λ wordt groot gekozen
2. Berekening van de Jacobiaan $J(\theta)$
3. Berekening van de stap δ en θ_{i+1}

$$\begin{aligned} \delta &= \left(J^T J + \lambda I_2\right)^{-1} J^T e \\ \theta_{i+1} &= \theta_i + \delta \end{aligned}$$

4. Geeft θ_{i+1} aanleiding geeft tot een verbetering – d.w.z. indien $V(\theta_{i+1}) \leqslant V(\theta_i)$?

- Ja: $\lambda_{\text{nieuw}} = \frac{\lambda}{2}$
  Ga terug naar stap 2 voor een volgende iteratie.
- Nee: $\lambda_{\text{nieuw}} = 10\ \lambda$
  Ga terug naar stap 3 en bereken een nieuwe δ en θ tot het antwoord op de vraag *Ja* is.

Dit moet blijven geïtereerd worden tot het convergentiecriterium bereikt is.

6.5 Convergentiecriterium

Als convergentiecriterium is gekozen voor dezelfde als voorgesteld in paragraaf 5.1.1 voor de gradiëntmethode. De voorgestelde voorwaarde op convergentie was daar:

$$\left| \frac{\theta_i - \theta_{i-1}}{\theta_{i-1}} \right| < 0.0001$$

Er is geen reden waarom hier een ander criterium zou gebruikt worden.

6.6 Samenvatting

In dit hoofdstuk zijn tot nu toe hogere orde methoden onder de loep genomen met hun voordelen, nadelen en beperkingen.

Gezien de Newton-Raphson methode te ingewikkeld is om te implementeren en teveel rekentijd zou vereisen is deze niet gebruikt. Gezien de Gauss-Newton methode foutmeldingen en divergerende oplossingen vertoonde werd ook deze niet gebruikt. Zo is besloten de Levenberg-Marquardt methode toe te passen

Een studie van deze methode zal nu uitwijzen of dit betere resultaten geeft dan de reeds toegepaste gradiëntmethode.

6.7 Toepassing van Levenberg-Marquardt methode

Om te testen hoe de methode van Levenberg-Marquardt vergelijkt met de gradiënt-methode en de ééndimensionale methode zal hier het ruismodel dat tevens geschat is met de gradiëntmethode en de ééndimensionale methode geschat worden met deze Levenberg-Marquardt methode.

6.7.1 Toepassen op ruis

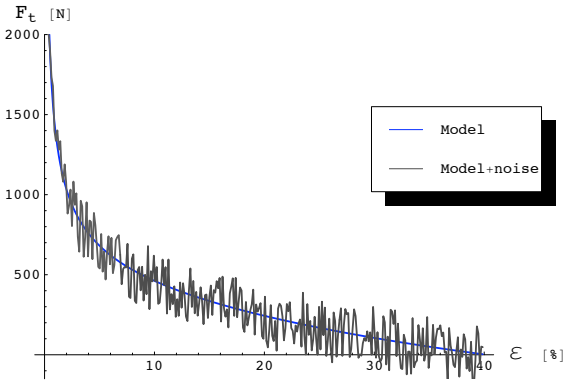
De ruis

Gezien de Levenberg-Marquardt methode verondersteld wordt veel sneller te convergeren dan de voorgaande methoden is het reeds een goede test te kijken hoe de methode omgaat met veel meer datapunten. Als te schatten data is er hetzelfde theoretisch model gekozen als gebruikt bij de vorige methode. Dit model heeft volgende karakteristieken:

- $L = 6$ cm
- $R = 1$ cm
- $p = 1.5$ bar

Deze waarden voor L en R zouden na identificatie bij benadering moeten gevonden worden.

Dezelfde uniforme ruis is gebruikt als in de vorige hoofdstukken, maar nu met veel meer punten. Het totaal gebruikt aantal punten waarmee de identificatie moet gebeuren is nu 300. De gebruikte ruis gesuperponeerd op het theoretische model is te zien in Figuur 6.1.



Figuur 6.1: Ruis op theoretisch model van F_t i.f.v. ε voor $L = 6$ cm, $R = 1$ cm en $p = 1.5$ bar.

Identificatie

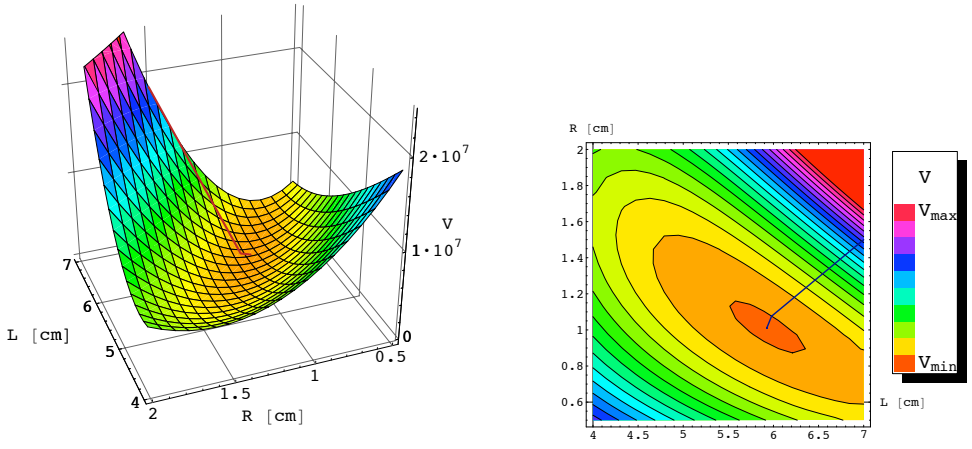
De identificatie verloopt volgens het algoritme beschreven in paragraaf 6.4.1. Als startwaarde voor λ is vrij willekeurig 10^7 gekozen. De gevonden waarde voor L en R bedraagt na 4(!) iteraties:

$$\begin{cases} L = 5.92868 \text{ cm} \\ R = 1.00989 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.87061$$

Dit wordt nog eens in Tabel 6.1 in detail getoond. Figuur 6.2 toont hier de grafische weergave van.

iteratie	L [cm]	R [cm]	V	L/R
0	7	1.5	18548043.1	4.6667
1	5.97908	1.07589	1981819.3	5.5574
2	5.92661	1.01100	1850644.2	5.8621
3	5.92867	1.00990	1850635.6	5.8706
4	5.92868	1.00989	1850635.6	5.8706

Tabel 6.1: slankheid L/R, V, L en R na enkele iteraties via de Levenberg-Marquardt methode toegepast op ruizige data van een theoretisch model



(a) 3D plot

(b) Contourplot

Figuur 6.2: Kostfunctie V i.f.v. L en R voor ruis op het theoretisch model van F_t met het gevolgde pad van de identificatieprocedure voor opeenvolgende iteraties

Zoals te zien is in Tabel 6.1 en Figuur 6.2 wordt er reeds na 1 iteratie een stap gezet die bijna uitkomt in het minimum van de kostfunctie. Dit is een fenomenaal verschil met de gradiëntmethode of de ééndimensionale methode. Het gebruiken van een 2de orde methode toont hier al onmiddellijk zijn nut. Telkens de stap dichter leidt bij het minimum geldt tevens dat de benadering voor de hessiaan – vergelijking (6.9)

– meer geoorloofd is. Dan is namelijk de term $\left(F_t(\theta) \Big|_{\substack{\varepsilon(k) \\ p(k)}} - F_{\text{data}}(k) \right)$ steeds kleiner. Zodoende zal de volgende stap zelfs een betere stap zijn dan de vorige.

Bij de ééndimensionale methode waren er 38 iteraties nodig en bij de gradiëntmethode zelfs 155 iteraties. De beginwaarde van θ speelt daardoor ook geen rol. Niet de gradiënt en dus een lineaire benadering van de kostfunctie wordt doorgevoerd, maar een kwadratische. Het spreekt voor zich dat deze methode dus veel sneller tot een oplossing leidt.

Omdat er een drastische vermindering is van het aantal nodige iteraties kan men de rekentijd die zo gewonnen is beter benutten door meer datapunten te gebruiken om de identificatie te laten gebeuren. Zo zal de oplossing relevanter zijn en minder afhankelijk van ruis.

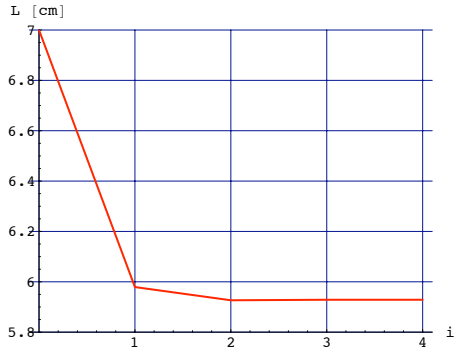
Convergentie

Om te tonen welke vorm de convergentie aanneemt is er nog Figuur 6.3 toegevoegd. Deze beschrijft L, R en V in functie van het aantal iteraties. Het valt op dat reeds bij de tweede iteratie een bijna finale oplossing gevonden is, die slechts nog verfijnd wordt door de volgende 2 iteraties.

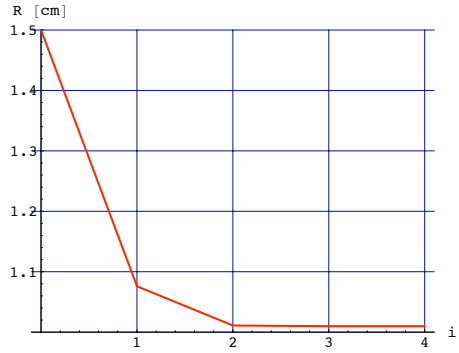
Controle

Hoe het resultaat overeenstemt met de ruis en het oorspronkelijk theoretisch model wordt getoond in Figuur 6.4. Het optimale wiskundige model past met deze methode inderdaad op het theoretisch model. Het is opmerkelijk hoe snel deze methode is in vergelijking met de vorig geteste methoden. Deze identificatie gebeurde in slechts 2 minuten, gebruik makend van 300 datapunten. Dit is een grote vooruitgang. Voor deze data is deze methode dus zonder twijfel de beste methode.

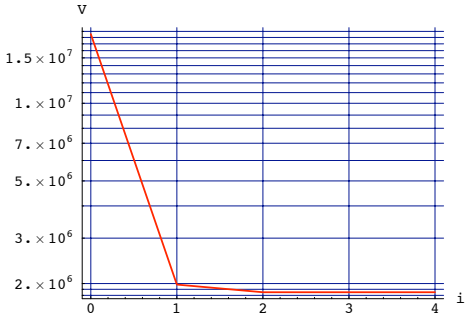
Om zeker te zijn dat dit geen toevalstreffer is, zal de methode nu nogmaals getest worden met data opgemeten via een trekbank.



(a) L

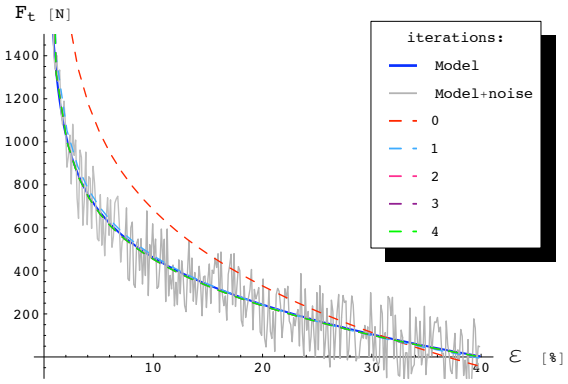


(b) R



(c) V

Figuur 6.3: L, R en V in functie van het aantal iteraties, voor ruis op een theoretisch model

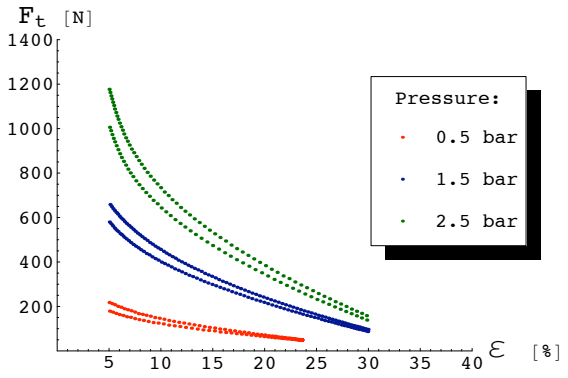


Figuur 6.4: Het optimale wiskundig model en het te benaderen ruisige model, bij verschillende iteraties, via de Levenberg-Marquardt methode

6.7.2 Toepassen op trekbankdata

Deze paragraaf zal de Levenberg-Marquardt methode toepassen op data opgemeten met een trekbank. De data is opgemeten bij 3 verschillende drukken. Aangezien deze methode snel bleek, werden er veel datapunten genomen voor extra relevantie. Er werden voor deze test 397 punten gebruikt. Als de methode erin slaagt om met zoveel punten de identificatie slechts in een redelijke tijd van enkele minuten te bewerkstelligen is de test geslaagd en kan verondersteld worden dat de methode klaar is om getest te worden op data afkomstig van de softarm.

Figuur 6.5 geeft de gebruikte data weer.



Figuur 6.5: Gebruikte data voor de identificatie met trekbankdata

Identificatie

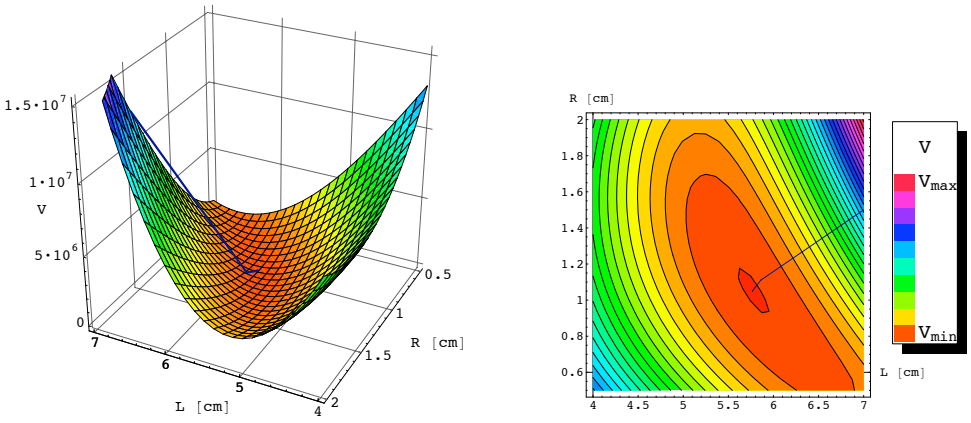
De bekomen resultaten door toepassing van de Levenberg-Marquardt methode op deze data zijn na 4 iteraties:

$$\begin{cases} L = 5.76726 \text{ cm} \\ R = 1.04983 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.4935$$

Tabel 6.2 toont het verloop in detail, en Figuur 6.6 toont hier de grafische weergave van.

iteratie	L [cm]	R [cm]	V	L/R
0	7	1.5	12428887.0	4.6667
1	5.86071	1.11099	278068.1	5.2752
2	5.76705	1.05090	187470.1	5.4877
3	5.76726	1.04982	187465.0	5.4936
4	5.76726	1.04983	187465.0	5.4935

Tabel 6.2: slankheid L/R, V, L en R na enkele iteraties via de Levenberg-Marquardt methode toegepast op trekbankdata bij 3 verschillende drukken



(a) 3D plot

(b) Contourplot

Figuur 6.6: Kostfunctie V i.f.v. L en R voor trekbankdata met het gevolgde pad van de identificatieprocedure voor opeenvolgende iteraties

Convergentie

Om te tonen welke vorm de convergentie aanneemt is er nog Figuur 6.7 toegevoegd. Deze beschrijft L, R en V in functie van het aantal iteraties. Het valt op dat reeds bij de tweede iteratie een bijna finale oplossing gevonden is, die slechts nog verfijnd wordt door de volgende 2 iteraties.

Controle

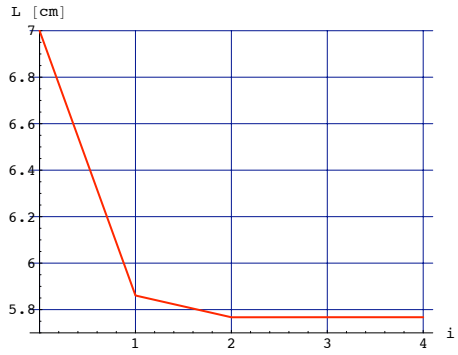
Hoe het resultaat overeenstemt met de ruis en het oorspronkelijk theoretisch model wordt getoond in Figuur 6.8. De identificatieprocedure verloopt ook hier zeer vlot en leidt tot een goede fit van het wiskundig model op de gemeten data bij 3 verschillende drukken. Dit in slechts enkele minuten tijd voor een groot aantal bemonsterde punten.

6.8 Conclusie

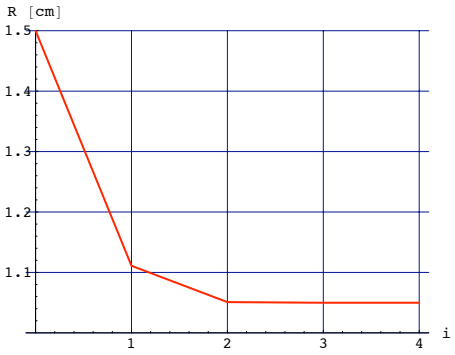
De Levenberg-Marquardt methode is getest en leidt tot lovenswaardige resultaten. De identificatie gebruikt een zeer klein aantal iteraties. Dit komt de rekentijd erg ten goede: slechts in enkele minuten tijd is de identificatie volbracht.

Het wiskundig model past ook goed op de data voor een heel bereik van drukken, en een bereik van ε tussen 5% en 30%. Onder de 5% wordt de contractie te klein en de resulterende kracht te hoog. Dan is het wiskundig model niet meer geldig omdat het geen rekening houdt met het elastisch gedrag van de materialen. Vermits om deze reden het werkingsgebied van de softarm boven $\varepsilon = 5\%$ is, is dit geen noemenswaardig probleem.

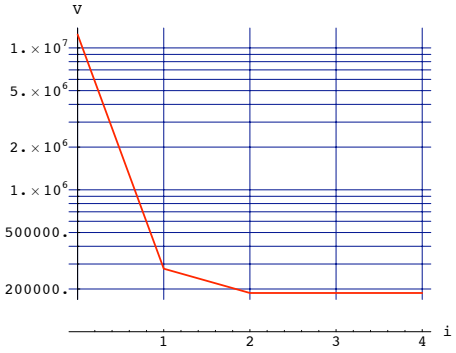
Zo is er nu een methode ontwikkeld die toegepast kan worden op data gemeten op de softarm.



(a) L

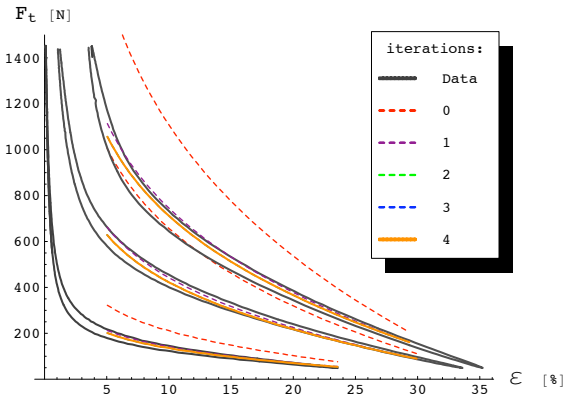


(b) R



(c) V

Figuur 6.7: L, R en V in functie van het aantal iteraties, voor de trekbankdata via de Levenberg-Marquardt methode



Figuur 6.8: Het optimale wiskundig model voor de trekbankdata, bij verschillende iteraties, via de Levenberg-Marquardt methode

Hoofdstuk 7

Sturing en bemeten van de Softarm

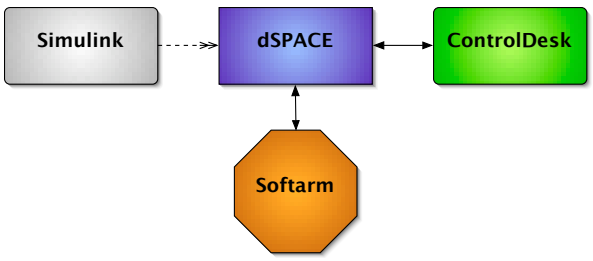
7.1 Inleiding

In dit hoofdstuk wordt eerst beschreven hoe de data van de softarm gemeten wordt. Vervolgens wordt uitgelegd welke bewerkingen nodig zijn op de gemeten data om ze geschikt te maken voor gebruik in een identificatiemethode.

In eerste instantie is er via Simulink een sturingsalgoritme ontwikkeld dat flexibel is en toelaat verschillende soorten gecontroleerde metingen uit te voeren.

Het Simulink algoritme wordt gecompileerd en wordt geladen op de digitale controller "DS1103 PPC Controller Board" van het merk *dSPACE*. Deze verzorgt ook de wisselwerking tussen de Softarm en de computer. Naar deze controller zal voortaan gerefereerd worden als "dSPACE".

Gekoppeld aan de sturing is een grafische interface via het programma ControlDesk. Dit programma laat toe om in Simulink opgestelde variabelen en metingen enerzijds weer te geven, anderzijds te manipuleren. ControlDesk zal het contact tussen dSPACE en de computer verzorgen. Simulink komt dus niet meer tussenbeide. Het dient enkel om het stuurprogramma te programmeren. Het sturingspricipe is schematisch weergegeven in Figuur 7.1



Figuur 7.1: Sturing van de softarm

Metingen worden op de softarm uitgevoerd via krachtsensoren, druksensoren en

optische hoekencoders. De metingen dienen dan geëxporteerd te worden naar bruikbare formaten voor identificatie. Met behulp van een Matlabprogramma wordt deze data verder verwerkt om bruikbaar te worden gemaakt voor de identificatie.

7.2 Simulink sturing

7.2.1 Principe

Het principe is zodanig dat via Simulink opgesteld wordt welke beweging de Softarm zal maken, m.a.w. op welke drukken de GPAS gebracht worden. Voor deze toepassing is het echter niet nodig een precieze beweging op te leggen. Het volstaat de softarm een vlotte continue beweging te laten maken door continu variërende drukken op te leggen.

7.2.2 Sturing

Om een zo continu mogelijk variërende beweging te hebben – dus niet schokkerig – werd gekozen om de drukken sinusoïdaal te laten variëren.

Er wordt een gemiddelde druk vastgelegd waarop de spieren van de softarm gebracht worden. Bovenop deze druk wordt een sinusoïdaal drukverloop gesuperponeerd. Twee spieren die antagonistisch opgesteld zijn krijgen net een tegengesteld en gelijk in amplitude sinusoïdaal verloop. Het verloop van de druk in 2 antagonistische spieren is dus als volgt:

$$p_{\text{spier}_1} = p_m + \Delta p \tag{7.1}$$

$$p_{\text{spier}_2} = p_m - \Delta p \tag{7.2}$$

met p_m de gemiddelde druk en Δp als volgt gedefinieerd:

$$\Delta p = A \sin\left(\frac{2\pi}{T}\, t\right) + B \tag{7.3}$$

A is de amplitude, B een bias, T de periode en t de tijd.

Door een spierpaar als zodanig te sturen treedt er een beweging op die continu is. Indien spier 1 de bovenste spier is en spier 2 de onderste, zal de beweging van dit paar een positieve hoekverdraaiing teweegbrengen als Δp positief is.

De ingestelde drukken zijn relatief t.o.v. de atmosferische druk. Dit wil zeggen dat als er 0 bar ingesteld wordt, er in principe enkel de luchtdruk heerst. Verder zal er enkel met relatieve drukken gewerkt worden.

Er is een begrenzing ingesteld die ervoor zorgt dat de drukken niet hoger dan 3 bar kunnen worden. Meer dan 3 bar zou de spieren kunnen beschadigen. Zeker indien dit langdurig gebeurt. Deze begrenzing is dus een beveiliging.

De drukkleppen zijn verbonden met relatieve druksensoren en werken zo dat ze via deze meting en de gevraagde druk de druk in de spier regelen op de gevraagde druk.

Flexibiliteit

Om flexibiliteit te garanderen in bewegingsmogelijkheden is het mogelijk om zowel de amplitude A, de bias B, én de periode T aan te passen terwijl het sturingsprogramma werkt. Dit kan apart ingesteld worden voor elke link van de softarm. Het is zelfs

mogelijk om de gemiddelde druk sinusoïdaal te laten variëren met dezelfde variabele parameters.

Er dient echter opgemerkt te worden dat indien deze parameters gewijzigd worden terwijl de druk aanstaat en de arm beweegt, deze parameters slechts geleidelijk (stapsgewijs) mogen gewijzigd worden om wilde bewegingen te vermijden. De periode mag enkel gewijzigd worden indien de druk weg is vermits er dan naar een andere sinus overgeschakeld wordt. De softarm kan dan plots uitschieten en eventueel iets beschadigen.

Overgangen

Om diverse redenen is het handig dat drukken die gestuurd worden naar de spieren op nul bar kunnen gebracht worden en dat weer druk toegelaten kan worden als de operator dat wenst. Het is natuurlijk onaanvaardbaar dat deze in- en uitschakeling ogenblikkelijk gebeurt. De overgang moet geleidelijk gebeuren. Dit vermijdt wilde bewegingen van de softarm die gevaarlijk zijn en schade kunnen veroorzaken.

Hiervoor is een Simulink-gedeelte gemaakt dat sturing van de drukken kan in- en uitschakelen, terwijl het stuurprogramma loopt. Zo is het niet langer nodig om de druk manueel af te sluiten door een kraantje, en bij het weer opendraaien van het kraantje kan men zich ervan verzekeren dat de druk die gevraagd word in de spieren nul bar bedraagt. Doordat dit kan gebeuren terwijl het programma loopt, is het niet langer nodig om een ander programma te laden dat de druk op 0 bar instelt.

Een bijkomend voordeel van hetzelfde programma continu te kunnen blijven gebruiken is dat gelogde data niet uit het geheugen verdwijnt. Dit zou wel het geval zijn bij het laden van een ander programma.

De softarm bezit 2 geledingen. Voor de sturing wordt er een zelfde gemiddelde druk p_m aangebracht in de spieren van beide geledingen. Voor het eerste gelid wordt hierop volgens vergelijkingen (7.1) en (7.2) gedefinieerde Δp_1 op gesuperponeerd; voor gelid 2 een Δp_2. Aangezien de sturing gebruik maakt van p_m, Δp_1 en Δp_2 moeten deze alledrie van dezelfde soort overgang voorzien worden.

In bijlage B.1 zijn de simulinkdiagramma's van deze overgangen toegevoegd, samen met de simulinkdiagramma's van het totale drukgedeelte met al zijn componenten.

Voor verdere referentie in dit eindwerk worden volgende naamgevingen gehanteerd: Δp_1 stuurt de eerste link van de softarm aan (tegen de muur), en Δp_2 stuurt de tweede link aan. p_1 wordt gevoed aan spier 1, p_2 aan spier 2, p_3 aan spier 3 en p_4 aan spier 4. De nummering van de spieren is dezelfde als weergegeven in paragraaf 2.3.2.

7.2.3 Kracht-offset resetten

Om krachten te meten zijn krachtsensoren aangebracht. Door montage op de softarm meten deze sensoren reeds een offset. Deze is echter vanzelfsprekend niet gewenst. De bedoeling is dus deze te kunnen resetten terwijl het programma loopt door even op een knop te klikken terwijl er geen druk heerst in de spieren. Van elke krachtmeting wordt nu de gemeten offset afgetrokken. De kracht die gemeten wordt zonder aanwezige drukken in de spieren wordt dus nul Newton. Dit is in overeenstemming met het wiskundig model van de spieren – zie vergelijking (3.2).

Hier toont zich meteen het nut van de drukovergangen (paragraaf 7.2.2). Indien een ander programma nodig was om de drukken op 0 bar te brengen en de meting van de offset uit te voeren, zou deze meting van de offset uit het geheugen verdwijnen bij het laden van het stuurprogramma.

Aangezien er ruis zit op de krachtmetingen kan het zijn dat de offset net gekozen is op een piek in de ruis als de "reset offset" knop afgezet wordt. Om dit te vermijden wordt een laagdoorlaatfilter geplaatst op de krachtmeting die de ruis wegfiltert. De filter heeft volgende karakteristiek in het Laplace-domein:

$$\mathrm{H} = \frac{T}{s+T} \tag{7.4}$$

De gebruikte constante T wordt 4 genomen. Het gebruik van een filter veroorzaakt echter een *delay*. Er moet dus kort gewacht worden tot de metingen zich gestabiliseerd hebben om een correcte offset te meten. Deze filter wordt niet gebruikt voor de eigenlijke krachtmetingen voor de identificatie. Hij dient louter voor visualisatiedoeleinden en de offset reset.

Vanaf het moment dat een knop aangevinkt wordt zal de offset de waarde van de gemeten kracht krijgen. Van de krachtmeting zal dan deze waarde afgetrokken worden. De kracht wordt dus 0. Als de knop afgezet wordt, zal de gemeten offset in het geheugen bewaard worden (een steeds herhalende lus in Simulink) en zal deze waarde afgetrokken worden van de gemeten kracht. In bijlage B.1 wordt getoond hoe dit in Simulink gerealiseerd is.

7.3 ControlDesk

ControlDesk is het programma dat communiceert met de digitale controller van het merk *dSPACE* die de softarm stuurt (zie Figuur 7.1). Het laat toe om data die dSPACE krijgt van de sensoren weer te geven op allerlei manieren. Tevens laat het toe om enkele stuurparameters te wijzigen.

De parameters die kunnen gewijzigd worden zijn getoond in Tabel 7.1.

Parameter	Betekenis	Default
Start/Stop	schakelaar die overgang (paragraaf 7.2.2) start om druk aan of af te zetten	Stop
Offset	knop die toelaat de offset te resetten	Uit
p_m	De gemiddelde druk voor alle spieren	1.5 bar
Amplitude p_m	De amplitude van de sinusgolf voor de gemiddelde druk	0 bar
Periode p_m	De periode van de sinusgolf voor de gemiddelde druk	20 s
Amplitude 1	De amplitude van de sinusgolf voor Δp_1 (link 1)	1 bar
Periode 1	De periode van de sinusgolf voor Δp_1 (link 1)	20 s
Bias 1	De bias van de sinusgolf voor Δp_1 (link 1)	0 bar
Amplitude 2	De amplitude van de sinusgolf voor Δp_2 (link 2)	0.9 bar
Periode 2	De periode van de sinusgolf voor Δp_2 (link 2)	20 s
Bias 2	De bias van de sinusgolf voor Δp_2 (link 2)	0 bar

Tabel 7.1: Wijzigbare parameters in ControlDesk

De data die gemeten wordt is getoond in Tabel 7.2. Buiten deze data worden ook

nog de gewenste drukken[1] , de kracht-offsets, ... gevisualiseerd.

Data	Betekenis
p_1	druk in spier 1
p_2	druk in spier 2
p_3	druk in spier 3
p_4	druk in spier 4
$q1$	relatieve hoek van link 1
$q2$	relatieve hoek van link 2
$q1d$	relatieve hoeksnelheid van link 1
$q2d$	relatieve hoeksnelheid van link 2
$F1$	kracht gemeten bij spier 1
$F2$	kracht gemeten bij spier 2
$F3$	kracht gemeten bij spier 3
$F4$	kracht gemeten bij spier 4

Tabel 7.2: Opgemeten parameters in ControlDesk

In bijlage B.2 wordt de interface getoond die ontwikkeld is om data van de softarm te visualiseren en de softarm te sturen.

Het is duidelijk dat men zowel over grafische als numerieke data kan beschikken. Dit programma is dus een krachtig instrument in samenwerking met dSPACE om de softarm te sturen en data te visualiseren terwijl de experimenten gebeuren.

De data die gedurende een bepaald tijdsinterval gevisualiseerd is kan dan geëxporteerd worden naar bijvoorbeeld een Matlab bestandstype, namelijk een ".mat bestand".

7.4 Dataverwerking

De identificatie gebeurt met het programma Mathematica. Mathematica laat namelijk toe om de elliptische integraalvergelijkingen (vergelijkingen 3.4) op te lossen. Er zijn echter een aantal redenen waarom de data uit het ".mat bestand" niet onmiddellijk kan doorgegeven worden aan Mathematica.

- Zoals gezien in de hoofdstukken die de identificatiemethoden beschrijven is de identificatie van deze spieren een rekenintensief proces. Om een aanvaardbare rekentijd te bekomen kunnen slechts een 300 à 400 punten uit deze data gebruikt worden (zie hoofdstuk 6). ControlDesk logt echter met een periode[2] 0.001 s. Als de data gedurende 80 s gelogd wordt betekent dit dus 80000 punten. Uiteraard is dit meer dan de identificatieprocedure aankan. De data moet dus *gedownsampled* worden indien dit niet door ControlDesk gebeurd is.

- Het ".mat bestand" is opgebouwd uit *structures*[3]. Met Mathematica is het onmogelijk om zo een bestand te importeren. Er is dus verwerking nodig van

[1] Deze stemmen niet noodzakelijk overeen met de gemeten drukken; het zijn de drukken die gevraagd worden aan de drukregelkleppen, en die de controller in die kleppen dus probeert in te stellen.

[2] Deze periode kan in ControlDesk eenvoudig aangepast worden.

[3] Een opbouwstructuur van Matlab waarbij 1 variabele een hele boomstructuur van data kan bevatten.

deze data tot een eenvoudiger structuur. Deze moet dan bewaard worden in een bestandsformaat dat Mathematica wel kan inlezen.

Een matlabprogramma zorgt voor het uit elkaar rafelen van de *structures* en de downsampling van deze data zodat er een 300 à 400 punten overgehouden worden. Deze data wordt in een grote matrix gezet en bewaard als een ".csv bestand" [4]. Dit is een bestandstype waaruit Mathematica de data wel kan importeren.

Om het proces voor de gebruiker te vereenvoudigen is er een soort van interface in Matlab gemaakt die het ".mat bestand" opvraagt via een dialoogvenster en het ".csv bestand" ook laat bewaren via een dialoogvenster. Er wordt verwezen naar bijlage B.3 voor meer informatie hieromtrent.

De data is nu in een door Mathematica compatibel formaat gebracht en klaar voor toepassing bij de identificatie van de spieren.

7.5 Conclusie

In dit hoofdstuk zijn de voorbereidende taken beschreven die toelaten om data te loggen die gebruikt wordt voor de identificatie. Met Simulink is de weg gelegd om via de grafische interface die gemaakt is in ControlDesk verschillende sturingen te voorzien. Zo is het mogelijk om variatie te brengen in de data die gemeten wordt.

[4] CSV staat voor Comma Separated Value

Hoofdstuk 8

Identificatie op softarmdata

8.1 Inleiding

In dit hoofdstuk wordt besproken hoe de Levenberg-Marquardt methode (zie hoofdstuk 6 toegepast wordt op data die opgenomen is op de softarm terwijl deze in beweging is (zie hoofdstuk 7). De spieren hoeven dus niet gedemonteerd te worden en getest op een trekbank.

De data die afkomstig is van de softarm bevat nog niet de data die gebruikt wordt voor de identificatie. Er dienen eerst nog omrekeningen te gebeuren.

8.2 Omrekeningen

8.2.1 Krachten

Krachtsensoren

De gebruikte krachtsensoren zijn van het merk *Futek* (`http://www.futek.com`). Ze leveren een bepaald voltage per Newton naargelang hun calibratie. Tabel 8.1 toont de fabrieksinstellingen voor elk van de 4 krachtsensoren. Deze zijn gekoppeld met versterkers. De kolom *Calibratie* in deze tabel is dan ook met inbegrip van een versterker.

Type			Serienr	Bereik	Calibratie
LTH300	(L2760)	druk	194006	4448 N	444.5022 N/V
LTH300	(L2760)	druk	194007	4448 N	445.6199 N/V
LTH300	(L2760)	druk	194008	4448 N	444.7200 N/V
LCM300	(L1650)	druk / trek	90234	4448 N	443.5138 N/V

Tabel 8.1: Gebruikte krachtsensoren

Het bereik en de calibratiefactor samen tonen aan dat de sensor waarden tussen 0 en 10V maximaal geeft.

Figuur 8.1 toont beide gebruikte types. Figuur 8.2 toont de implementatie ervan in de softarm.

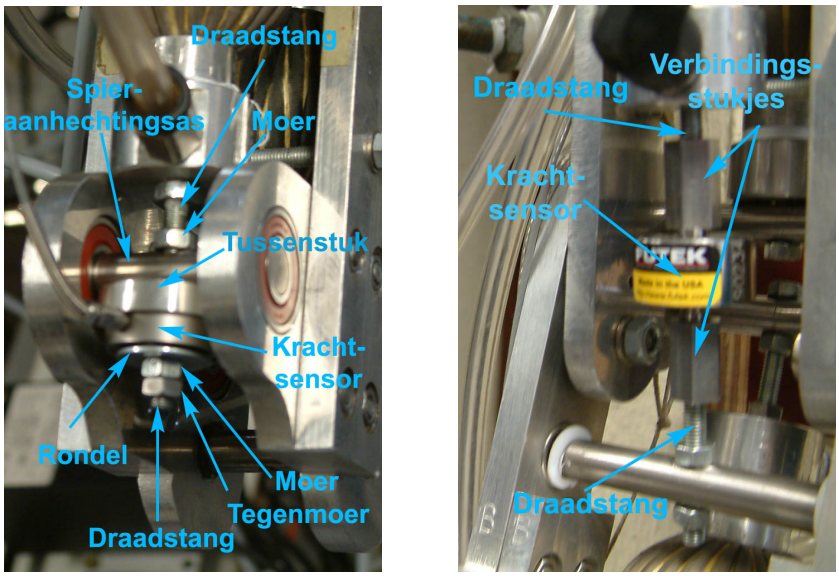
Type LTH300 heeft een donut-vorm waardoor een draadstang loopt die verbonden is met een spier. De sensor zal langs 1 kant geblokkeerd zijn door een rondel en moer,

en de andere kant door een tussenstuk[1] en spieraanhechtingsas waardoor de draadstang ook loopt. Beide kanten van de sensor zijn zo geblokeerd, zodat als de spier aan de draadstang trekt deze de moer sterker tegen de sensor aandrukt. Zo wordt deze sensor in druk belast.

Type LCM300 is een sensor die tussen 2 draadstangen op 1 lijn kan gemonteerd worden. Deze sensor kan zowel in trek als in druk belast worden door te trekken aan beide draadstangen in tegengestelde richting of naar elkaar toe te drukken. Dit type wordt gebruikt voor spier 4 omdat er niet genoeg plaats is om een krachtsensor van het type LTH300 te monteren.

(a) LTH300 (b) LCM300

Figuur 8.1: De twee gebruikte types krachtsensor



(a) LTH300 (b) LCM300

Figuur 8.2: Implementatie van beide gebruikte types krachtsensoren

[1]Dit stuk dient om de sensor enkel te belasten op de plaatsen die daarvoor bestemd zijn.

Calibratie

Er is vastgesteld dat de krachten die spier 4 uitoefende veel groter waren dan de krachten uitgeoefend door spier 3. Dit is hoogstmerkwaardig omdat men zou verwachten dat spier 3 meer kracht uitoefent dan spier 4 gezien deze link bijna steeds werkt bij een absolute hoek $(q_1 + q_2) > -90°$. In deze standen ondervindt de tweede link van de softarm namelijk een positief koppel. Om dit te genereren moet de kracht van spier 3 groter zijn dan die van spier 4 of tenminste van dezelfde grootte-orde[2]. Er is dus iets fout.

Een logische verklaring zou kunnen zijn dat de calibratie niet juist is. Daarom is de calibratie opnieuw zelf uitgevoerd. De gebruikte opstelling hiervoor is te zien in bijlage C

Aan de draadstang is onderaan een schijfje gemonteerd. Hierop zijn 4 massa's van elk ongeveer 10 kg geplaatst. De kracht die een massa van 10kg uitoefent is gekend:

$$F = m\,g = 10\text{kg}\ 9.81\text{m/s}^2 = 98.1\text{N} \tag{8.1}$$

Op deze manier worden de sensoren belast door 4 zulke massa's die een totale kracht van 392.4 N teweeg brengen. De massa's zijn apart gewogen en zijn gekend op 0.1g nauwkeurig. Ze worden stapsgewijs toegevoegd. Op deze manier is het mogelijk de sensor op 5 punten te calibreren en een lineaire fit doorheen deze 5 punten te nemen.

De bekomen nieuwe waarden van de calibratiefactor op deze manier zijn gegeven in Tabel 8.2

Type	Serienr	Calibratie Fabriek	Calibratie Zelf
LTH300	194006	444.5022 N/V	475.29 N/V
LTH300	194007	445.6199 N/V	477.37 N/V
LTH300	194008	444.7200 N/V	475.64 N/V
LCM300	90234	443.5138 N/V	442.50 N/V

Tabel 8.2: Nieuwe calibratiefactoren voor de krachtsensoren

Het valt op dat voor het type LTH300 de nieuwe calibratiefactoren allemaal rond dezelfde waarde zitten, maar ongeveer 30N/V verschillend van de calibratiefactoren gegeven door de fabrikant. Voor het type LCM300 is het verschil tussen de nieuwe waarde en de waarde van de fabrikant miniem in vergelijking met het verschil bij LTH300.

Gebruik van deze nieuwe calibratiefactoren zal al deels een stap in de goede richting zijn voor het vermeldde probleem van spier 3 en 4. Dit zal echter zeker geen factor 3 compenseren.

Na alles meermaals gecontroleerd te hebben, en de hoop op correcte data bijna opgegeven was, is geprobeerd om de krachtsensoren losjes te monteren zodanig dat de offset die de krachtsensoren meten bij montage zeer klein is. Het losjes monteren gebeurt door een moer en een tegenmoer (zie Figuur 8.2) tegen elkaar te spannen in plaats van een enkele moer tegen de sensor te spannen. Dit experiment bleek de oplossing van het probleem te zijn. F3 en F4 zijn nu van dezelfde grootte-orde en het koppel dat ze samen uitoefenen op de link is volgens de verwachtingen: positief bij

[2]indien de lastarm overeenkomstig met spier 3 groter is dan die overeenkomstig met spier 4 kan het koppel ook positief zijn bij de kracht van spier 4 groter dan van spier 3

$(q_1 + q_2) > -90°$, 0Nm bij $(q_1 + q_2) = -90°$ en negatief bij $(q_1 + q_2) < -90°$.

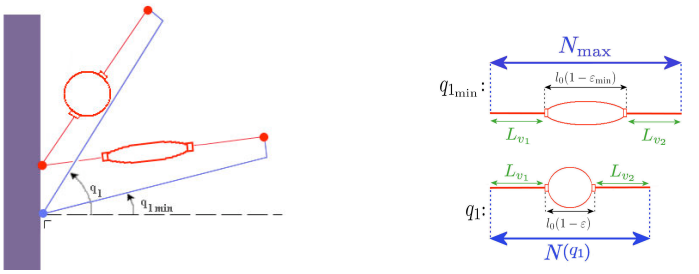
In Mathematica worden eerst deze waarden afkomstig van de sensor omgerekend naar Newton via de calibratiefactoren uit Tabel 8.2.

8.2.2 Contracties

Op de softarm is het onmogelijk om contracties rechtstreeks op te meten. Het wiskundig model dat geoptimaliseerd moet worden bevat echter contracties. Het is dus noodzakelijk om deze te verkrijgen.

Wat er echter wel opgemeten is, zijn de relatieve hoeken q_1 en q_2. Via kennis van deze hoeken, de designparameters van de softarm en de plaatsing van de spier is het mogelijk de contracties te berekenen. Hoe dit gebeurt zal nu beschreven worden.

Om de situatie beter te begrijpen is er Figuur 8.3 toegevoegd. Deze toont hoe de arm beweegt bij een contractie van spier 1. De softarm heeft een bepaald werkbereik. De hoek q_1 is dus begrensd door 2 hoeken: $q_{1\min}$ en $q_{1\max}$. Bij $q_{1\min}$ is de contractie dan ook op zijn minimale waarde $\varepsilon_{\min}$ en is N – de afstand tussen de aanhechtingspunten van de spier – maximaal met de waarde $N_{\max}$.



(a) Opstelling op softarm (b) Afzonderlijke spier

Figuur 8.3: Contractie van een spier

Laat l_0 de maximale lengte zijn van de spier indien de contractie 0% bedraagt. Bij $q_{1\min}$ is de lengte die de spier inneemt van de afstand $N_{\max}$ gelijk aan $l_0(1-\varepsilon_{\min})$. Bij $q_{1\min}$ kan $N_{\max}$ uitgeschreven worden als:

$$N_{\max} = l_0(1-\varepsilon_{\min}) + L_{v_1} + L_{v_2} \tag{8.2}$$
$$= l_0(1-\varepsilon_{\min}) + L_v \tag{8.3}$$

L_{v_1} en L_{v_2} zijn de stukjes die geen deel van de spier uitmaken tussen de aanhechtingspunten. Hun som is L_v en wordt de verbindingslengte genoemd.

Uit vergelijking (8.3) kan zo L_v berekend worden die nodig is om bij de minimale hoek $q_{1\min}$ een contractie van $\varepsilon_{\min}$ te bekomen[3]

$$L_v = N_{\max}(q_{1\min}) - l_0(1-\varepsilon_{\min}) \tag{8.4}$$

Deze verbindingslengte L_v moet zo ingesteld worden op de link zodanig dat bij $q_{1\min}$ een contractie van $\varepsilon_{\min}$ plaatsvindt.

[3]Voor de softarm is gekozen : $\varepsilon_{\min} = 5\%$ en $\varepsilon_{\max} = 30\%$

Voor gelijk welke hoek q_1 geldt:

$$N(q_1) = l_0\,(1 - \varepsilon(q_1)) + L_v \tag{8.5}$$

De in te stellen L_v is berekend in vergelijking (8.4). De contractie ε kan nu geschreven worden in functie van q_1 door:

$$\varepsilon(q_1) = \frac{l_0 + L_v + N(q_1)}{l_0} \tag{8.6}$$

N is een functie van q_1 die af te leiden is uit het ontwerp van de opstelling.

Op de softarm

Voor een seriële spier van de softarm bevat de verbindingslengte nog meer componenten, namelijk ook de stukjes tussen elk deel van de spier dat zijn eigen contractie ondergaat.

De maximale lengte l_0 is op de softarm gedefinieerd als de lengte van 1 spier van de seriële opstelling. De gebruikte l_0 moet dus vervangen worden door (nsp l_0), waarbij nsp het aantal spieren in serie is. Vergelijkingen (8.4) en (8.6) kunnen dus herschreven worden als:

$$L_v = N_{\max}(q_{1_{\min}}) - \text{nsp } l_0(1 - \varepsilon_{\min}) \tag{8.7}$$

$$\varepsilon(q_1) = \frac{\text{nsp } l_0 + L_v + N(q_1)}{\text{nsp } l_0} \tag{8.8}$$

Dit proces kan herhaald worden voor elke spier om de contractie van elke spier te verkrijgen in functie van de verantwoordelijke hoek. Voor link 1 zal dit de hoek q_1 zijn, voor link 2 de hoek q_2. Voor spieren aan de onderkant wordt ervoor gezorgd dat bij de minimale hoek de contractie niet groter wordt dan $\varepsilon_{\max}$. De N voor die spieren is dan immers minimaal en de contractie maximaal. Voor deze spieren geldt dus voor de berekening van L_v:

$$L_v = N_{\min}(q_{1_{\min}}) - \text{nsp } l_0(1 - \varepsilon_{\max}) \tag{8.9}$$

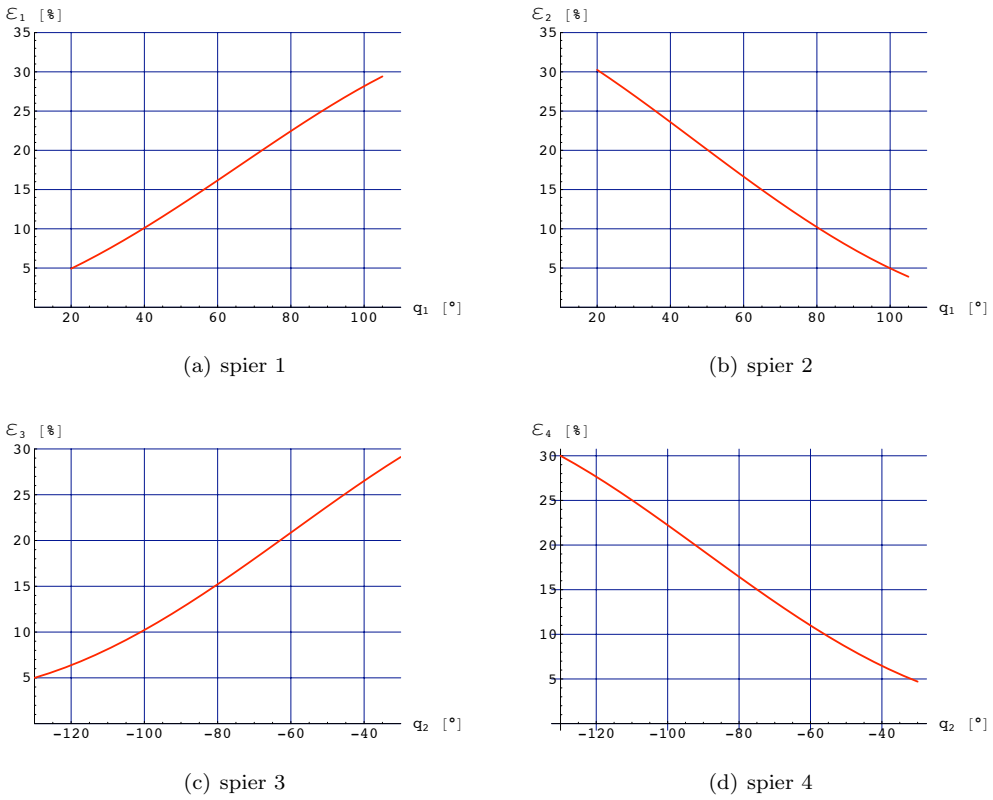
Nog op te merken valt dat de waarde l_0 hier niet vervangen wordt door zijn te identificeren waarde, maar door de reële waarde waarmee hij ontworpen is (6cm).

De instellingen voor L_v die dienen gemaakt te worden zijn te vinden in Tabel 8.3. Het is belangrijk deze zo goed mogelijk in te stellen om fouten in ε te voorkomen.

De functies die ε weergeven in functie van hun overeenkomstige relatieve hoek q zijn te zien in Figuur 8.4

spier	$q_{\min}$	$q_{\max}$	L_v
spier 1	20°	105°	11.7035 cm
spier 2	20°	105°	7.45416 cm
spier 3	−130°	−30°	9.43038 cm
spier 4	−130°	−30°	13.9809 cm

Tabel 8.3: In te stellen waarden voor L_v



(a) spier 1

(b) spier 2

(c) spier 3

(d) spier 4

Figuur 8.4: Contractie ε in functie van de overeenkomstige relatieve hoeken

Foutenanalyse

Aangezien het zeer moeilijk is om L_v correct in te stellen gebeurt een foutenanalyse die de invloed van een afwijking op de ingestelde waarde omzet in een afwijking op ε. Deze zal op zijn beurt een afwijking veroorzaken op F_t.

Fout op ε In de veronderstelling dat er geen meetfout aanwezig is op q_1 en q_2 en de ontwerpparameters van de softarm is de fout op N gelijk aan nul in vergelijking (8.8). Als tevens de fout op de parameter l_0 verwaarloosd wordt, kan men de absolute fout van ε voor deze vergelijkingen herschrijven als:

$$\Delta\varepsilon = \frac{\Delta L_v}{\text{nsp } l_0} \tag{8.10}$$

In deze vergelijking is duidelijk dat $\Delta\varepsilon$ proportioneel is met de afwijking van de berekende waarde van L_v. De fout is tevens onafhankelijk van de relatieve hoek q. Tabel 8.4 toont $\Delta\varepsilon$ voor een afwijking van 1 mm voor L_v.

spier	nsp	$\Delta\varepsilon$
spier 1	4	0.4167 %
spier 2	3	0.5556 %
spier 3	4	0.4167 %
spier 4	3	0.5556 %

Tabel 8.4: Fout op ε bij ($\Delta L_v = 1$ mm) en ($l_0 = 6$ cm)

Procentueel gezien is er echter wel een afhankelijkheid van q_1. Voor het geval $L_v = 1$ mm is de procentuele fout op ε gegeven in Figuur 8.5. Om bij andere L_v dit verloop te kennen kan men deze fout proportioneel aanpassen met L_v – zie vergelijking (8.10).

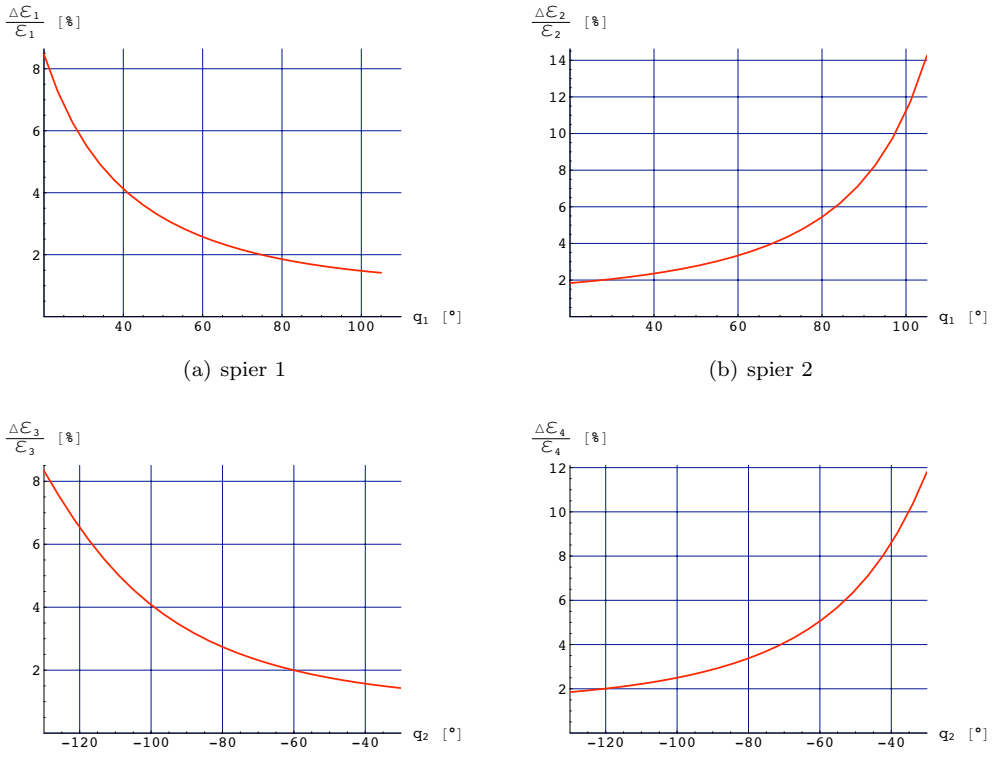
Fout op F_t Wat het effect van ΔL_v nu is op F_t is niet zomaar analytisch te berekenen door de elliptische integraalvergelijkingen (3.4) die telkens numeriek dienen opgelost te worden.

Men kan het effect bekijken bij verschillende drukken en bij verschillende contracties. F_t is rechtevenredig met de druk p. De fout ΔF_t is ook evenredig met p. Dit wil zeggen dat de procentuele fout $\frac{\Delta F_t}{F_t}$ onafhankelijk is van de druk p gezien de p's in teller en noemer elkaar wegdelen.

Door de proportionaliteit met p kan $p = 1$ bar genomen worden bij het bekijken van het verloop van ΔF_t i.f.v. ΔL_v bij verschillende ε. Bij andere drukken hoeft dit verloop dan slechts vermenigvuldigd worden met de druk in bar.

Figuur 8.6 toont de absolute en de procentuele fout op F_t [4] bij verschillende ε i.f.v. ΔL_v. Het verloop bij een negatieve ΔL_v is anders dan voor een positieve. Bij $\varepsilon = 5\%$ blijkt de absolute fout het grootst te zijn; zeker voor grote afwijkingen van L_v. Voor grotere contracties zijn de absolute fouten kleiner. Voor procentuele fouten is het echter net andersom met uitzondering van $\varepsilon = 5\%$. Het is dus duidelijk dat een onnauwkeurige instelling van L_v een grote invloed heeft op de gemeten krachten.

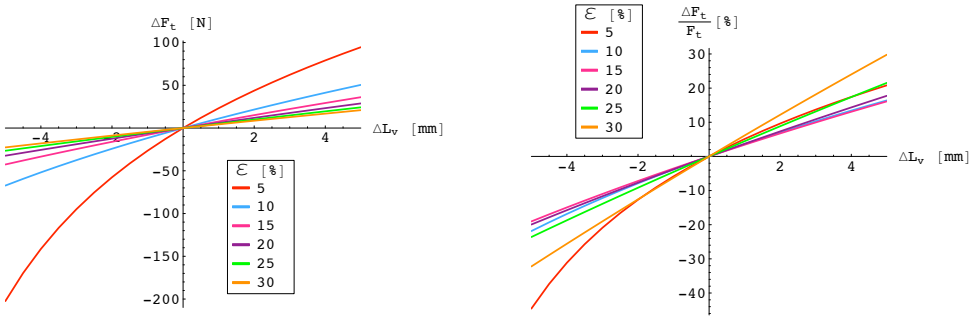
[4] F_t is berekend met nsp = 3. De fout bij nsp = 4 zal iets kleiner zijn.



(a) spier 1 (b) spier 2

(c) spier 3 (d) spier 4

Figuur 8.5: Procentuele fout $\frac{\Delta\varepsilon}{\varepsilon}$ in functie van de overeenkomstige relatieve hoeken bij $L_v = 1$ mm



(a) absolute fout (b) procentuele fout

Figuur 8.6: Fout op F_t i.f.v. ΔL_v bij verschillende ε

8.3 Identificatie door softarm data

Na vele identificatiemethoden getest te hebben, de sturing van de softarm en de captatie van data verzorgd te hebben, en alle opgemeten data omgezet te hebben naar data die gebruikt kan worden voor de identificatieprocedure kan eindelijk de identificatie door deze data volbracht worden.

Als gebruikte data zullen twee soorten gebruikt worden.

- De softarm kan men gewoon laten bewegen zonder eraan te komen. De krachten die zo teweeg worden gebracht zijn echter niet groot.

- Tijdens de beweging van de softarm kan men handmatig de beweging van de softarm tegenwerken. Zo zal bij een bepaalde druk de contractie sterk verkleind worden wat resulteert in een grotere kracht – zie hoofdstuk 3 Figuur 3.1 . Deze laat dus identificatie toe op een groter bereik van krachtdata.

De methode die gebruikt is is de Levenberg-Marquardt methode die in hoofdstuk 6 uitgebreid getest is. De convergentie wordt met deze data daarom niet meer besproken.

8.3.1 Opgenomen data

De data die opgemeten is bij een vrije beweging van de arm zal voortaan naar gerefereerd worden als *vrije data*. De data waarbij de arm handmatig belast wordt zal voortaan naar gerefereerd worden als *belaste data*.

De data is opgenomen gedurende 80 s met een sampleperiode van 0.001 s. Dit geeft aanleiding tot 80000 punten. Deze zijn dus nog eens *gedownsampled* via een sampleperiode van 0.250 s. Dit geeft aanleiding tot 320 punten. De opgemeten krachten in elke spier zullen verder getoond worden bij de identificatie van elke spier afzonderlijk.

Vrije data

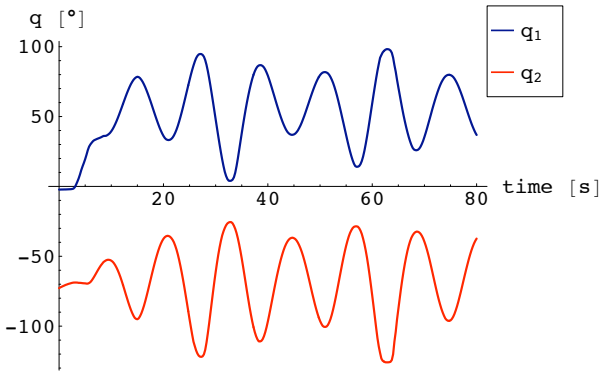
Als instellingen om deze data te maken werden volgende waarden ingevoerd in de grafische interface van ControlDesk:

	p_m	Δp_1	Δp_2
Bias [bar]	1.5	0.25	0
Amplitude [bar]	0.5	0.75	1
Periode [s]	30	12	12

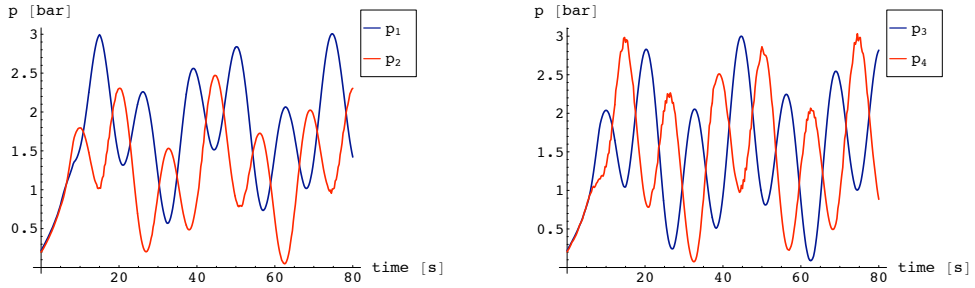
De gevolgde beweging van de arm kan gevisualiseerd worden door de relatieve hoeken q_1 en q_2, respectievelijk horend bij link 1 en link 2 van de softarm. Figuur 8.7 laat dit zien.

De onderarm (link 2) beweegt zoals te zien is in tegenfase met de bovenarm (link 1). De drukken verantwoordelijk voor deze beweging zijn gegeven in Figuur 8.8

Het is duidelijk dat de hoeken in Figuur 8.7 dezelfde periode hebben als hun respectievelijke drukken in Figuur 8.8. De hoek is gerelateerd met het verschil tussen deze drukken.



Figuur 8.7: relatieve hoeken bij vrije beweging



(a) bovenarm (b) onderarm

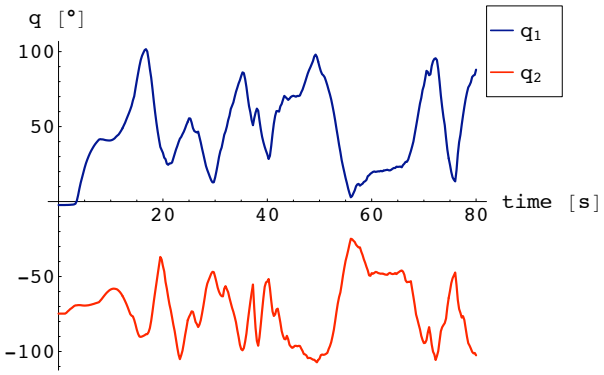
Figuur 8.8: Drukken die in de spieren heersen bij de vrije beweging

Belaste data

Als instellingen om deze belaste data te maken werden volgende waarden ingevoerd in de grafische interface van ControlDesk:

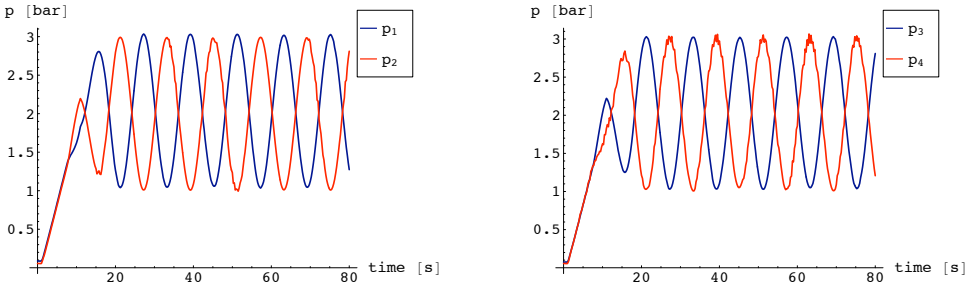
	p_m	Δp_1	Δp_2
Bias [bar]	2	0	0
Amplitude [bar]	0	1	1
Periode [s]	30	12	12

De gevolgde beweging van de arm kan voor belaste data gevisualiseerd worden door de relatieve hoeken q_1 en q_2, respectievelijk horend bij link 1 en link 2 van de softarm. Figuur 8.9 laat dit zien.



Figuur 8.9: relatieve hoeken bij beweging bij belasting

Deze hoeken zijn niet drukgestuurd maar handmatig bewerkstelligd door de arm tegen te houden. De drukken die echter in de spieren heersten zijn gegeven in Figuur 8.10



(a) bovenarm (b) onderarm

Figuur 8.10: Drukken die in de spieren heersen bij de belaste beweging

Het is duidelijk dat er geen verband is tussen de drukken in Figuur 8.10 en de hoeken in Figuur 8.9.

8.3.2 Identificatie van spier 1

Vrije data

Een identificatie is gebeurd op deze data en heeft met gebruik van vrije data volgend resultaat voor R en L na 4 iteraties.

$$\begin{cases} L = 5.36155 \text{ cm} \\ R = 1.04815 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.11527$$

De vergelijking tussen het resulterende wiskundige model en de krachtmetingen kunnen op 2 manieren gevisualiseerd worden: enerzijds in functie van de tijd wat het duidelijkst is, anderzijds in functie van de contractie ε wat een beter beeld geeft bij welke contracties welke krachten optreden. Deze vergelijkingen worden gegeven door Figuur 8.11.
In deze figuur staat "model first" voor het niet-geupdate model waarbij $L = 6$ cm en $R = 1$ cm. Er is door de identificatie een merkbare verbetering veroorzaakt. Over de hele lijn wordt de kracht beter voorspeld met deze geupdate waarden voor L en R.

Belaste data

Voor belaste data kan men dezelfde methode toepassen. De uitkomst voor L en R wordt dan na 4 iteraties:

$$\begin{cases} L = 5.38862 \text{ cm} \\ R = 1.04473 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.15789$$

Deze waarden wijken niet veel af van de gevonden waarden bij vrije data. De vergelijking tussen het model en de data zijn voor belaste data gegeven in Figuur 8.12. Hier dient gelet te worden op de grote schaal. De kracht die inwerkt gaat tot 1400N. Dit is tot 1000N meer dan bij vrije data. Dit toont meteen het nut van deze data.

8.3.3 Identificatie van spier 2

Vrije data

Het resultaat voor R en L bij gebruik van vrije data voor spier 2 is na 3 iteraties:

$$\begin{cases} L = 6.01391 \text{ cm} \\ R = 1.27477 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 4.71765$$

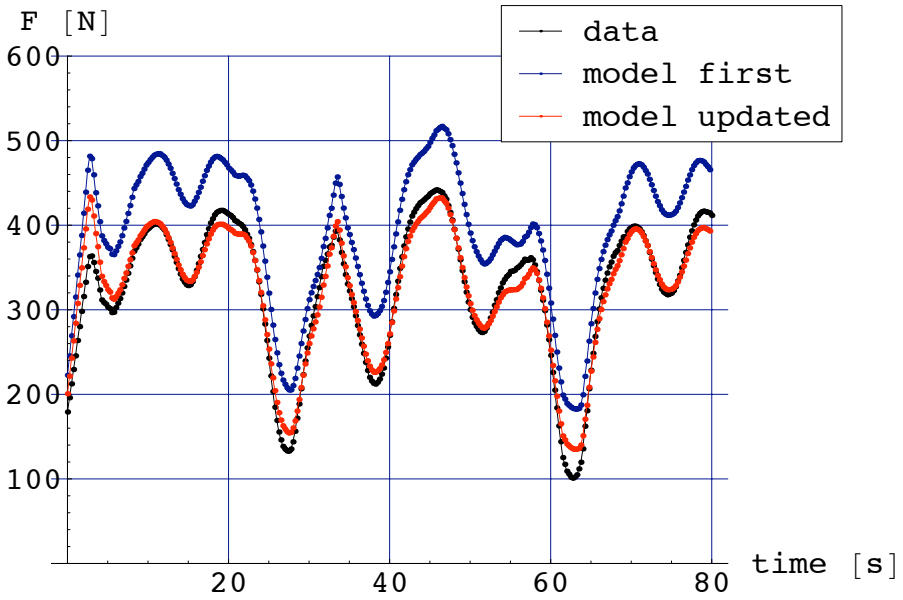
De vergelijking tussen het model en de data is voor vrije data gegeven in Figuur 8.13. Het verschil tussen het geupdate model en het oorspronkelijke model is niet zo groot.

Belaste data

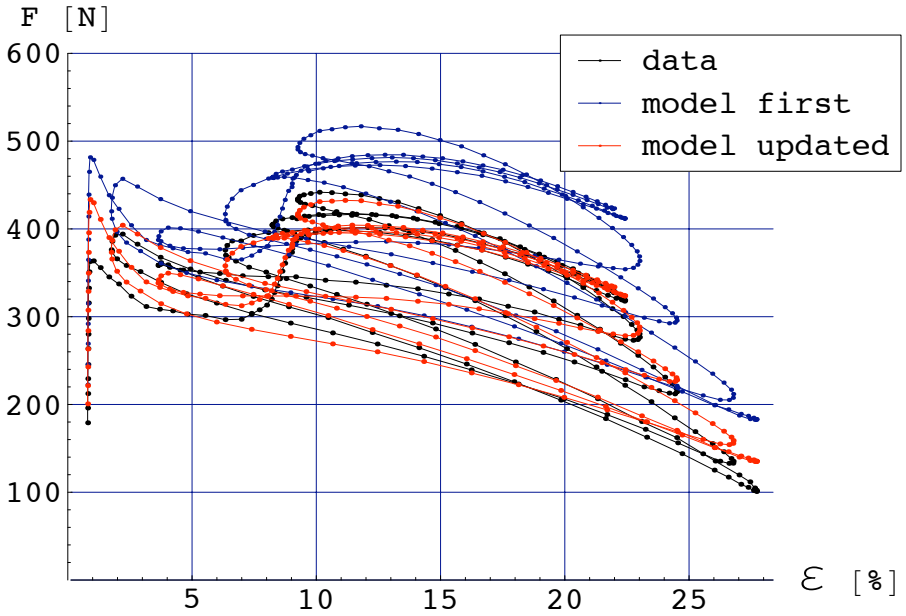
De uitkomst voor L en R wordt bij gebruik van belaste data na 3 iteraties:

$$\begin{cases} L = 6.17052 \text{ cm} \\ R = 1.11965 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 5.51112$$

De weergave van het model op de data is voor belaste data gegeven in Figuur 8.14. Het updaten van het model heeft hier duidelijk zin: het verschil tussen het geupdate model en het oorspronkelijk model is significant.

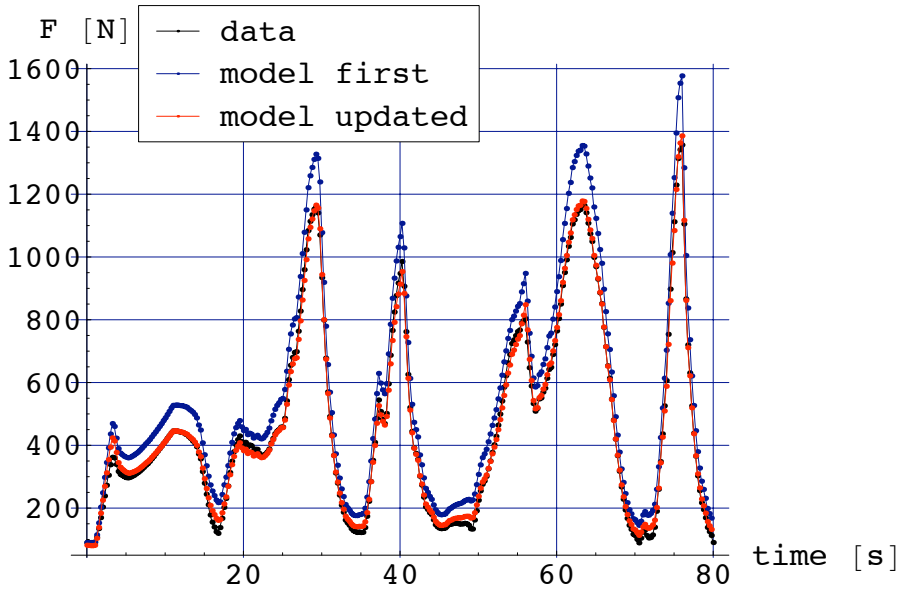


(a) kracht i.f.v. tijd

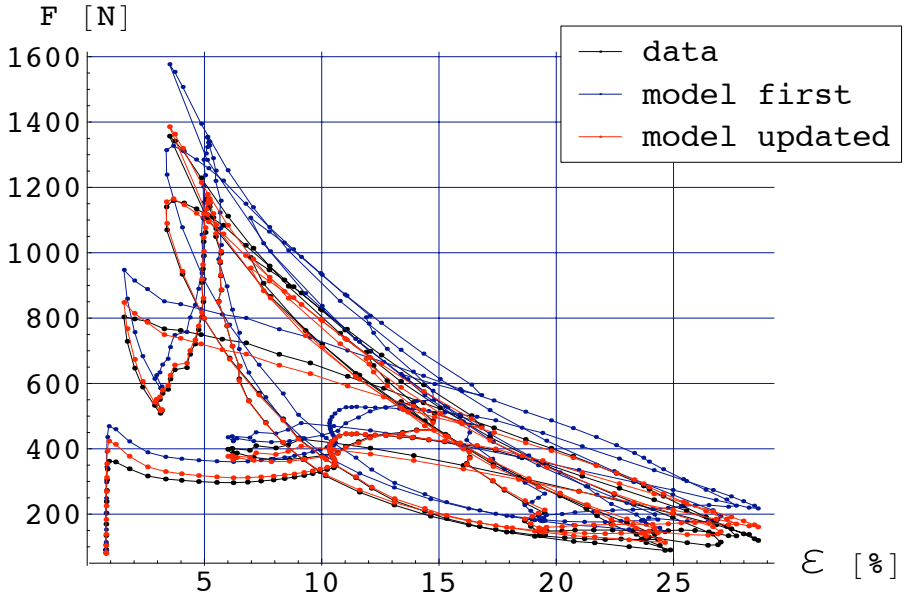


(b) kracht i.f.v. contractie

Figuur 8.11: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 1 met vrije data

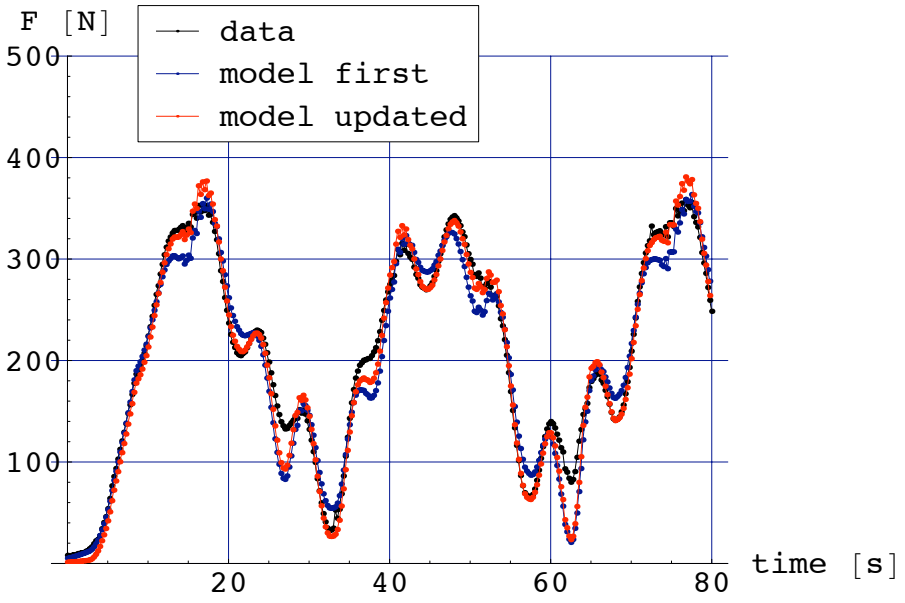


(a) kracht i.f.v. tijd

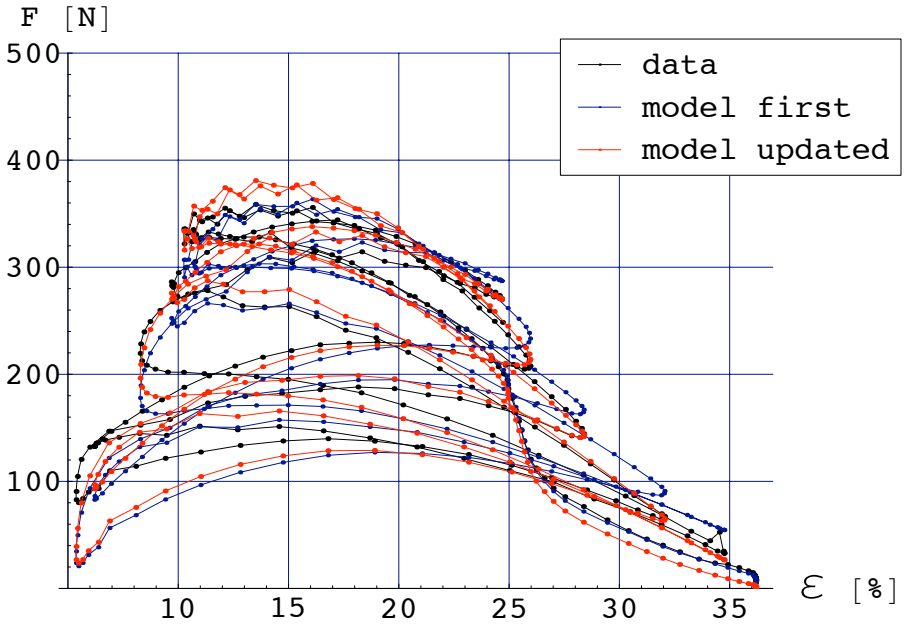


(b) kracht i.f.v. contractie

Figuur 8.12: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 1 met belaste data

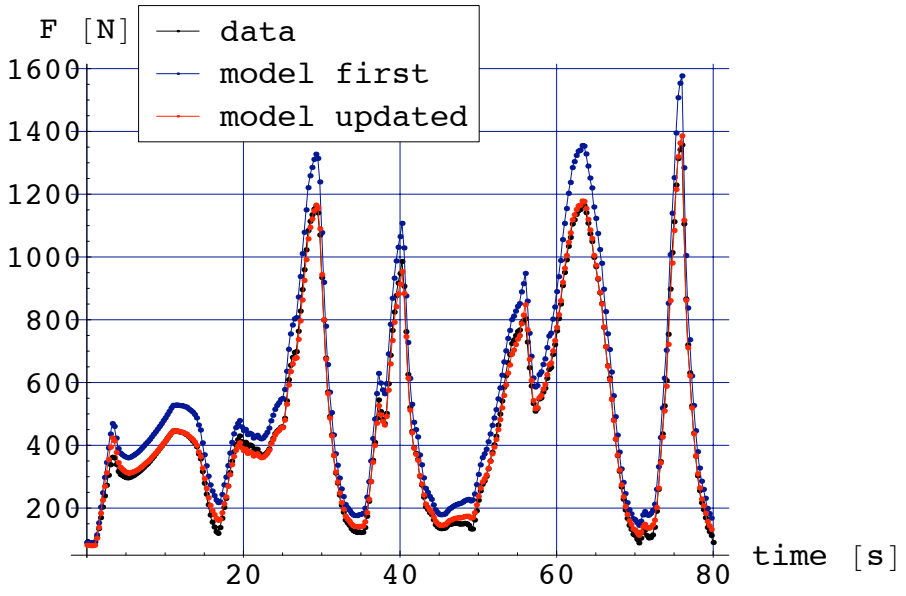


(a) kracht i.f.v. tijd

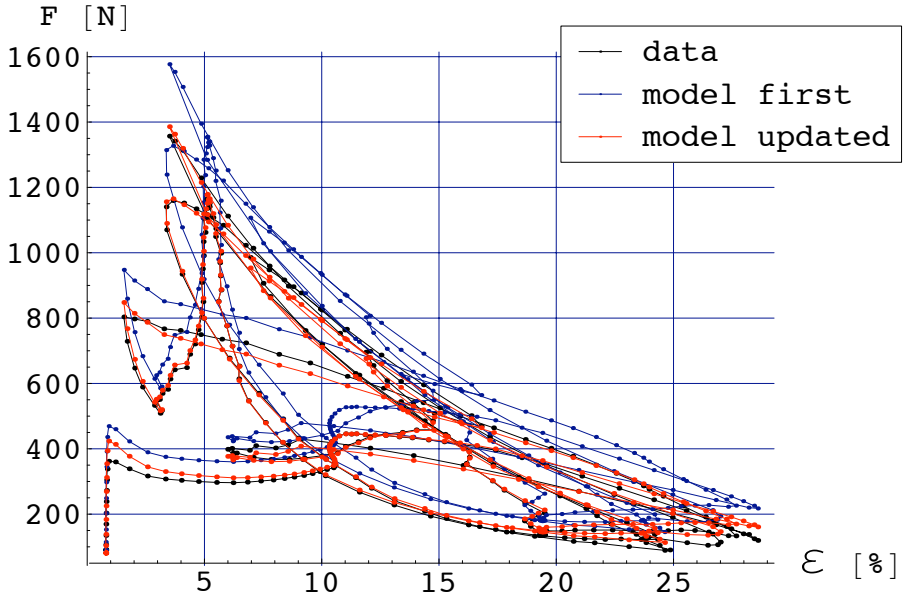


(b) kracht i.f.v. contractie

Figuur 8.13: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 2 met vrije data



(a) kracht i.f.v. tijd



(b) kracht i.f.v. contractie

Figuur 8.14: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 2 met belaste data

8.3.4 Identificatie van spier 3

Vrije data

Het resultaat voor R en L bij gebruik van vrije data voor spier 3 is na 3 iteraties:

$$\begin{cases} L = 5.6847 \text{ cm} \\ R = 1.19552 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 4.754981$$

De vergelijkingen tussen het model en de data zijn voor vrije data gegeven in Figuur 8.15. Ook voor deze spier blijkt de identificatie tot een goede verbetering te leiden ten opzichte van het oorspronkelijk model.

Belaste data

De uitkomst voor L en R wordt bij gebruik van belaste data na 3 iteraties:

$$\begin{cases} L = 5.73983 \text{ cm} \\ R = 1.22117 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 4.70026$$

Deze waarden zijn vergelijkbaar met diegene gevonden voor vrije data. Dit toont aan dat beide datasets ongeveer dezelfde informatie bevatten om een degelijke identificatie te volbrengen. De vergelijkingen tussen het model en de data zijn voor belaste data gegeven in Figuur 8.16.

8.3.5 Identificatie van spier 4

Vrije data

Het resultaat voor R en L bij gebruik van vrije data voor spier 4 is na 4 iteraties:

$$\begin{cases} L = 4.39479 \text{ cm} \\ R = 1.18051 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 3.72278$$

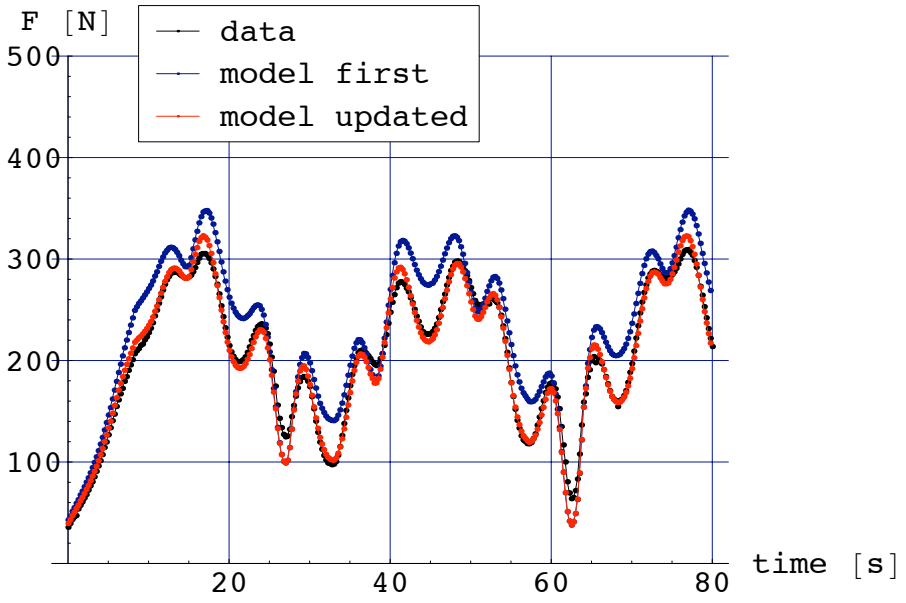
Deze waarden wijken vrij veel af van de oorspronkelijke waarden. De vergelijkingen tussen het model en de data zijn voor vrije data gegeven in Figuur 8.17. Het enorme verschil tussen het oorspronkelijk model en het geupdate model is opmerkelijk. Vooral voor deze spier is het nut van de identificatie heel duidelijk. Het oorspronkelijk model overschat de metingen telkens met ongeveer 50% !

Belaste data

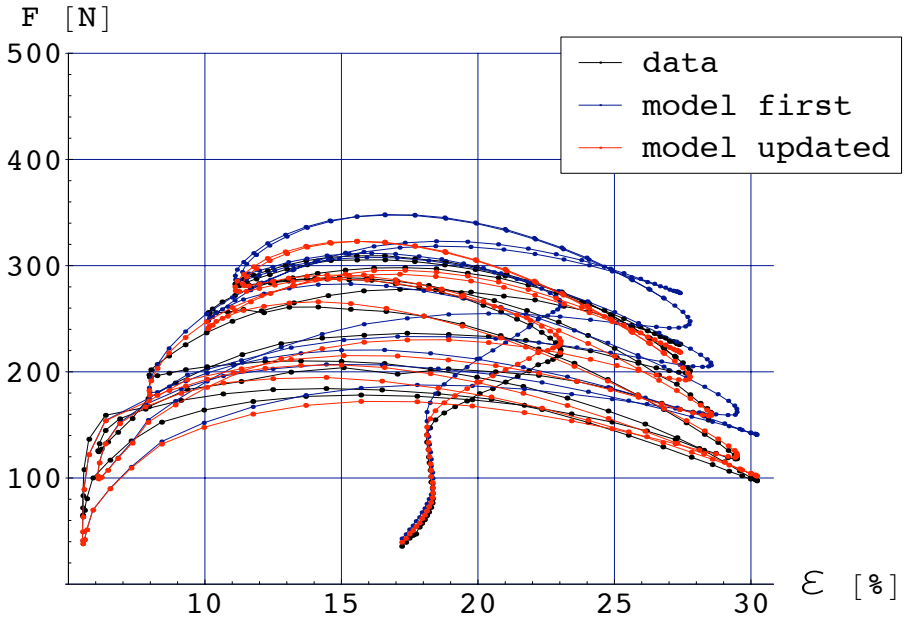
De uitkomst voor L en R wordt bij gebruik van belaste data:

$$\begin{cases} L = 4.43024 \text{ cm} \\ R = 1.09606 \text{ cm} \end{cases} \Rightarrow \text{slankheid } L/R = 4.04196$$

De vergelijkingen tussen het model en de data zijn voor belaste data gegeven in Figuur 8.18. Ook bij belaste data komt men tot dezelfde vaststelling: het oorspronkelijk model overschat de metingen met minstens 50% . Het geupdate model echter volgt de metingen quasi perfect.

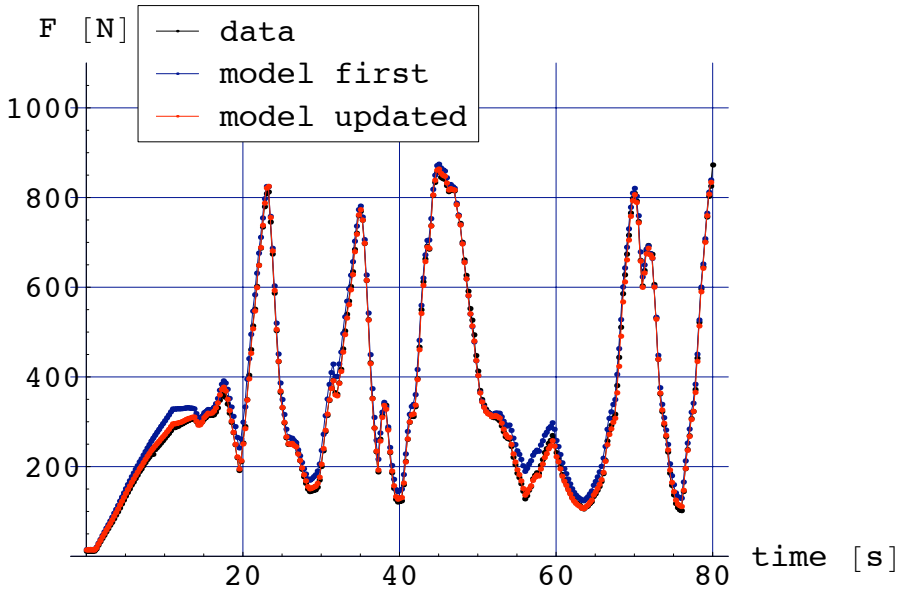


(a) kracht i.f.v. tijd

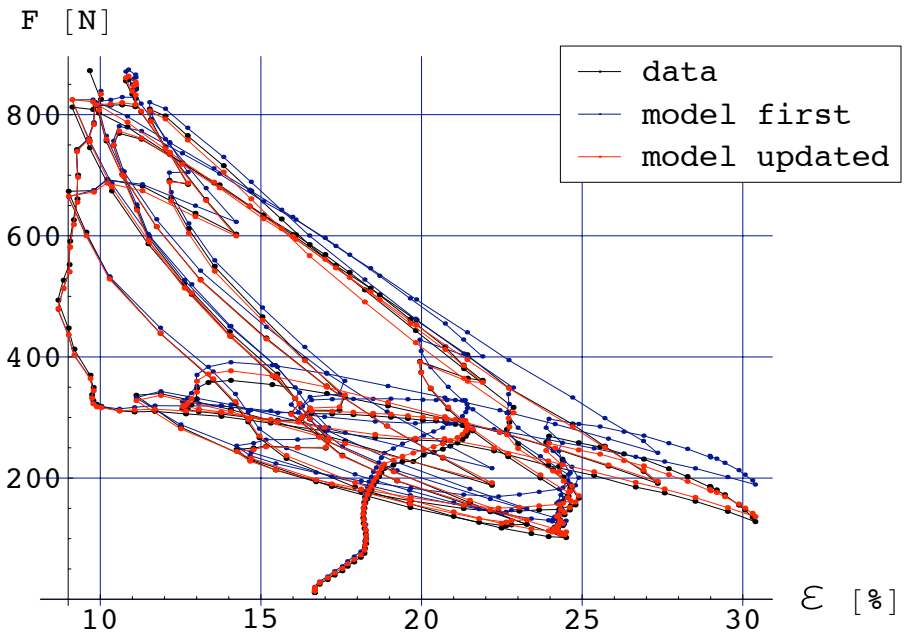


(b) kracht i.f.v. contractie

Figuur 8.15: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 3 met vrije data

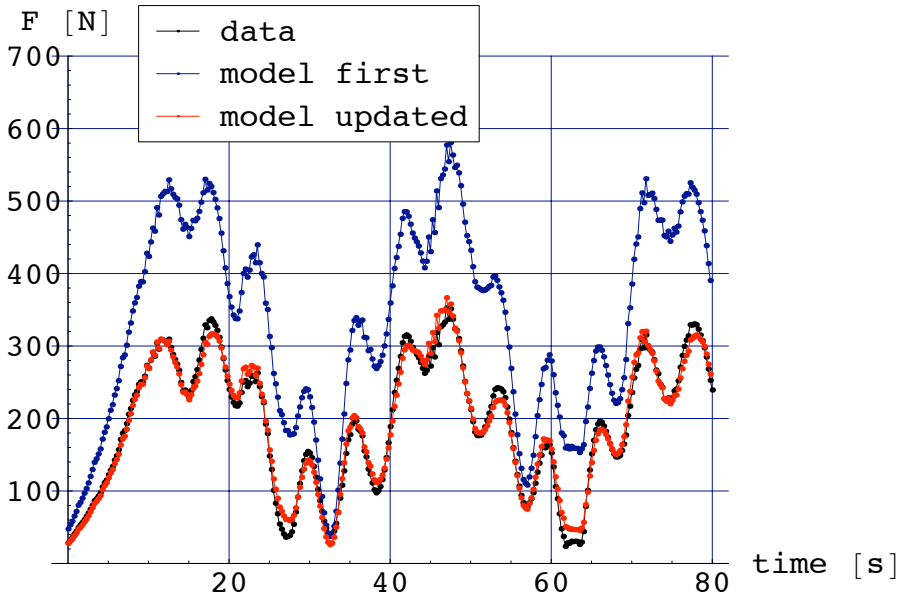


(a) kracht i.f.v. tijd

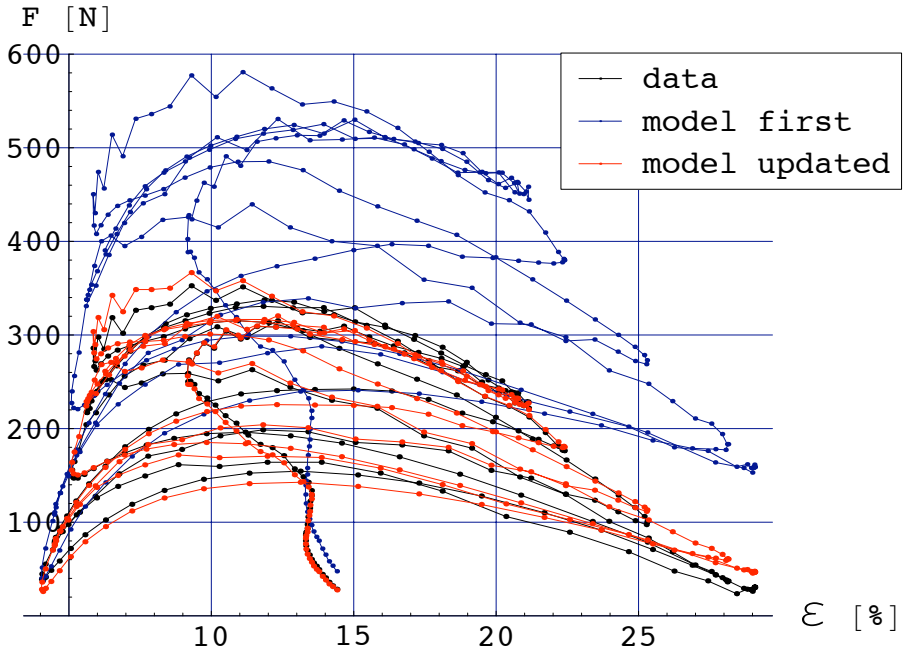


(b) kracht i.f.v. contractie

Figuur 8.16: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 3 met belaste data

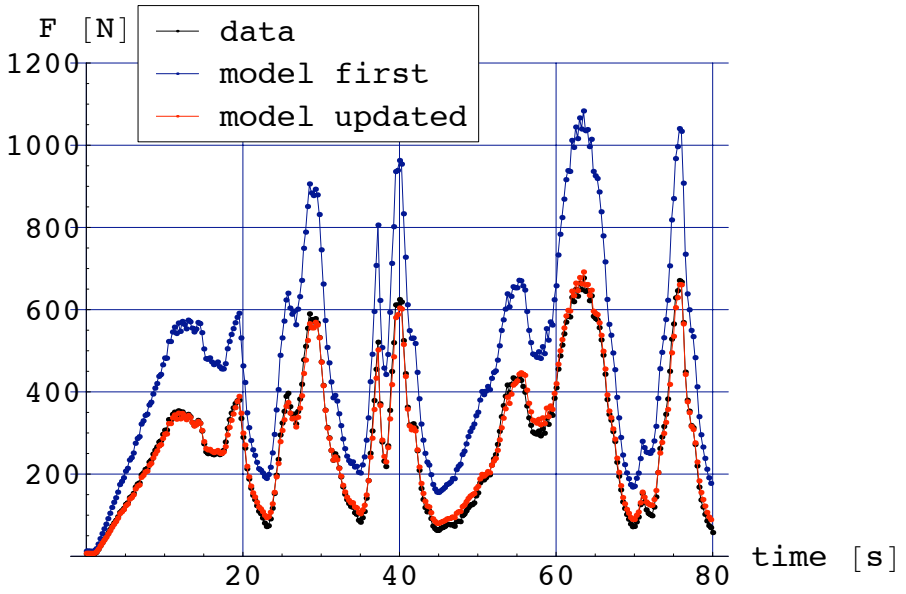


(a) kracht i.f.v. tijd

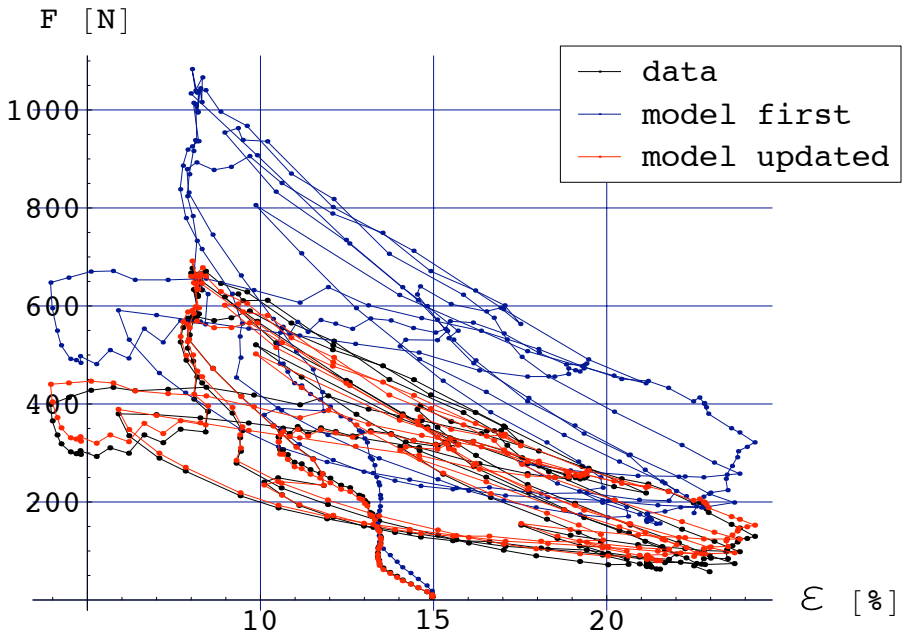


(b) kracht i.f.v. contractie

Figuur 8.17: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 4 met vrije data



(a) tijd



(b) contractie

Figuur 8.18: Weergave van geupdate model op de gemeten data in functie van tijd en contractie voor spier 4 met belaste data

8.3.6 Samenvatting

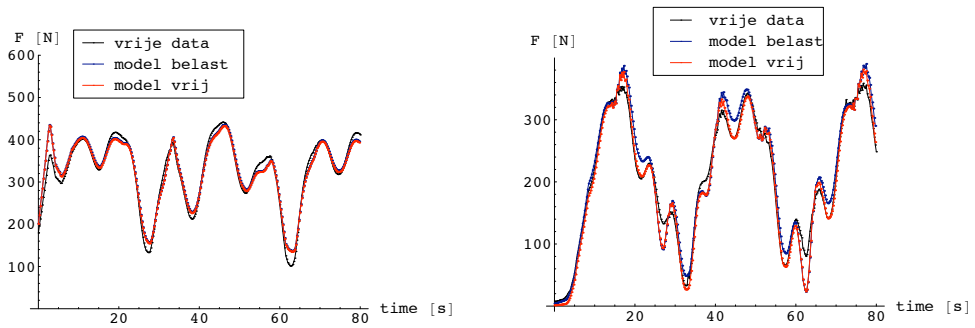
De geupdate waarden van L , R en de slankheid kunnen nog in een tabel gezet worden om een beter overzicht te krijgen van de resultaten.

specificatie		L [cm]	R [cm]	L/R
spier 1	vrij	5.36	1.05	5.12
	belast	5.39	1.04	5.16
spier 2	vrij	6.01	1.27	4.72
	belast	6.17	1.12	5.51
spier 3	vrij	5.48	1.20	4.75
	belast	5.74	1.22	4.70
spier 4	vrij	4.39	1.18	3.72
	belast	4.43	1.10	4.04

De bekomen waarden wijken zeker voor spier 4 af van wat ontworpen is. Het is dus nuttig deze identificatie uit te voeren. De geoptimaliseerde wiskundige modellen die met deze parameters gevormd worden volgen quasi altijd goed de metingen waarop de identificatie gebeurd is.

8.3.7 Validatie

Vooraleer kan besloten worden of identificatie van de spieren een succes is moeten de geoptimaliseerde wiskundige modellen getest worden met andere data dan met welke de identificatie verlopen is. Er wordt getest hoe het model gevonden via belaste data de vrije data kan voorspellen. Voor spier 1 en spier 2 is deze vergelijking te zien in Figuur 8.19



(a) Spier 1

(b) Spier 2

Figuur 8.19: Validatie van model gevonden via belaste data op vrije data

Voor spier 1 is dit duidelijk een goed geslaagde validatie. Voor figuren van de validatie van de andere spieren wordt verwezen naar bijlage D. Er blijkt dat de grootste afwijking tussen beide modellen optreedt voor spier 2. Dit is echter nog zeker aanvaardbaar. Opmerkelijk is ook dat voor spier 2 de slankheden ook meer van elkaar verschillen dan bij andere spieren het geval is. Hier ligt dus hoogstwaarschijnlijk de oorzaak van de grotere afwijking. Door het gemiddelde te nemen van de resultaten

voor L en R bij vrije en belaste data past het overeenkomstige model iets minder bij de belaste data, maar beter bij de vrije data in vergelijking met het model dat gevalideerd werd. Dit is een goed compromis; de verbetering is nog steeds veel groter ten opzichte van het oorspronkelijk model.

Er kan dus besloten worden dat de systeemidentificatie van de spieren een succes is. Voor de uiteindelijk te gebruiken waardes van L en R worden de gemiddeldes gebruikt van de waarden bekomen uit vrije en belaste data. Deze worden in volgende tabel op een rijtje gezet:

spier	L [cm]	R [cm]	L/R
spier 1	5.38	1.05	5.14
spier 2	6.09	1.20	5.10
spier 3	5.61	1.21	4.64
spier 4	4.41	1.14	3.87

In bijlage D wordt getoond hoe het model met de gemiddelde parameters zich verhoudt tot de 2 modellen waar de parameters gevonden zijn via belaste en vrije data.

8.4 Conclusie

In dit hoofdstuk is de Levenberg-Marquardt methode toegepast op data afkomstig van de softarm. Er is een foutenanalyse gebeurd van de fouten die ontstaan door het foutief instellen van de verbindingslengte.

De zo bekomen wiskundige modellen kunnen onmiddellijk toegepast worden in controle-algoritmes. De gemeten contracties gebruikt in deze algoritmes zijn immers op dezelfde manier gemeten. Zo zal zelfs als er foutief ingestelde waarden voor de verbindingslengtes L_v gebruikt worden deze fout opgehoffen worden vermits de controle-algortimes van dezelfde fout gebruik maken. De gewenste beweging zal zo via deze updated wiskundige modellen beter kunnen gecontroleerd worden, wat de uiteindelijke bedoeling is waarom dit eindwerk gemaakt is.

Deel III

Identificatie van manipulator

Hoofdstuk 9

Inleiding

9.1 Inleiding

De koppels die inwerken in de gewrichten van de softarm kunnen gemodelleerd worden met het "dynamisch model". Dit model maakt gebruik van de "dynamische parameters". Deze zijn: de massa's, de massamiddelpunten en de massatraagheidsmomenten van elk gelid. Via deze parameters wordt berekend welke koppels nodig zijn om de softarm een bepaalde positie in te laten nemen. De drukken die nodig zijn om deze koppels in te stellen worden berekend via het spiermodel dat uitgebreid aan bod is gekomen in Deel II.

Het probleem is dat de dynamische parameters niet precies gekend zijn. Foutieve waarden voor deze parameters resulteren in andere posities dan verwacht. Zelfs indien deze parameters volkomen exact gekend zouden zijn, zou dit nog steeds niet het geval zijn, vermits in het gebruikte model enkele effecten niet gemodelleerd zijn zoals o.a. wrijving in de gewrichten van de softarm.

Door de dynamische parameters te identificeren kan men er wel voor zorgen dat de gemaakte fouten geminimaliseerd worden. Dit is dan ook het doel van dit deel van het eindwerk.

In dit hoofdstuk zal het dynamisch model opgesteld worden. Dit model moet echter in een andere vorm gebracht worden om de parameterschattingen uit te voeren. Ook dit wordt in dit hoofdstuk beschreven.

In hoofdstuk 10 worden de parameterschattingen uitgevoerd met een *offline-methode*. Dit wil zeggen dat de identificatie gebeurt na acquisitie van data

In hoofdstuk 11 worden de parameterschattingen uitgevoerd met een *online-methode*. Hiermee wordt bedoeld dat de identificatie gebeurt met data terwijl deze gemeten wordt.

9.2 Dynamisch model

Het dynamisch model van de softarm beschrijft het koppel in functie van de relatieve hoeken, de relatieve hoeksnelheidheden en de relatieve hoekversnellingen.

In de literatuur (zie bvb. [6]) wordt beschreven dat het dynamisch systeem als volgt kan geschreven worden:

$$\tau_i = \frac{d}{dt}\left(\frac{dL}{d\dot{q}_i}\right) - \frac{dL}{dq_i} \tag{9.1}$$

Hierbij stelt τ een vector voor met als componenten de koppels in de gewrichten van de geledingen. L is de Lagrangiaan van het systeem, q de vector met als componenten de relatieve hoeken zoals gedefinieerd in Figuur 2.6 en $\dot{q}$ een vector met als componenten de afgeleiden naar de tijd van de componenten van q.
De Lagrangiaan is gedefinieerd als het verschil tussen de kinetische energie en de potentiële energie:

$$L = E_k - U \tag{9.2}$$

Deze kinetische energie en potentiële energie zijn de som van respectievelijk die voor de eerste link en die van de tweede link van de softarm, telkens genomen in het massamiddelpunt van elke link. Voor de kinetische energie:

$$\begin{aligned} E_k &= E_{k_1} + E_{k_2} && (9.3)\\ &= \frac{1}{2}\left(m_1 v_{G_1}^2 + I_{z_{G_1}} \dot{q_1}^2\right) + \frac{1}{2}\left(m_2 v_{G_2}^2 + I_{z_{G_2}} (\dot{q_1} + \dot{q_2})^2\right) && (9.4) \end{aligned}$$

Voor de potentiële energie:

$$\begin{aligned} U &= U_1 + U_2 && (9.5)\\ &= m_1\, g\, y_{G_1} + m_2\, g\, y_{G_2} && (9.6) \end{aligned}$$

In de bovenstaande vergelijkingen stellen m_1 en m_2 de massa voor van respectievelijk het eerste gelid en het tweede gelid van de softarm. G_1 en G_2 stellen de massamiddelpunten voor van beide geledingen. v_{G_1} is de snelheid van G_1 en v_{G_2} is de snelheid van G_2. $I_{z_{G_1}}$ en $I_{z_{G_2}}$ stellen respectievelijk de massatraagheidsmomenten om assen rond het massamiddelpunt van respectievelijk gelid 1 en gelid 2. y_{G_i} stelt de y-coördinaat voor van G_i. De $y-as$ verloopt vertikaal naar boven tegengesteld aan de richting van de zwaartekracht. De x-as ligt in het vlak loodrecht op de zwaartekracht en vormt samen met de $y-as$ het vlak waarin de softarm kan bewegen. Voor de eenvoud van schrijfwijze zal het subscript G worden weggelaten in het vervolg van dit eindwerk.

De coördinaten van G_1 en G_2 zijn afhankelijk van de hoeken q_1 en q_2. Deze afhankelijkheid is gegeven via vergelijkingen (9.7) en (9.8). Hierin stelt L_G de afstand voor tussen de scharnier van het gelid en de loodrechte projectie van het massamiddelpunt van dat gelid op de langsrichting van het gelid. d_G stelt de korste afstand voor tussen het massamiddelpunt van het gelid en de langsrichting. L_1 stelt de lengte van gelid 1 voor.

$$G_1 = \begin{cases} L_{G_1}\cos q_1 + d_{G_1}\sin q_1 \\ L_{G_1}\sin q_1 - d_{G_1}\cos q_1 \end{cases} \tag{9.7}$$

$$G_2 = \begin{cases} L_1\cos q_1 + L_{G_2}\cos(q_1+q_2) + d_{G_2}\sin(q_1+q_2) \\ L_1\sin q_1 + L_{G_2}\sin(q_1+q_2) - d_{G_2}\cos(q_1+q_2) \end{cases} \tag{9.8}$$

Substitutie van alle deze vergelijkingen in elkaar levert uiteindelijk de 2 bewegingsvergelijkingen die de softarm beschrijven. Met wrijvingseffecten is geen rekening

gehouden. De bewegingsvergelijkingen zijn functie van q, $\dot{q}$ en $\ddot{q}$. Ze kunnen ook in matrixvorm geschreven worden via volgende vergelijking.

$$\tau = H\,\ddot{q} + C\,\dot{q} + G \tag{9.9}$$

H noemt men de massamatrix (2x2 matrix) van het systeem, C de coriolis- en centrifugaalmatrix (2x2 matrix) , en G de gravitatiematrix (2x1 matrix). In bijlage E zijn deze matrices gegeven in vergelijkingen (E.1), (E.2) en (E.3).

De matrices H, C en G zijn functie van de parameters m_1, m_2, I_{z_1}, I_{z_2}, L_{G_1}, d_{G_1}, L_{G_2} en d_{G_2}. Goede kennis van deze parameters laat toe via de bewegingsvergelijkingen (9.9) de beweging van de arm te voorspellen in die mate dat de verwaarloosde effecten geen grote rol spelen. Deze dynamische parameters zijn via CAD-tekeningen berekend. De op deze manier berekende waarden zijn gegeven in Tabel 9.1.

Parameter	Waarde	Parameter	Waarde
m_1 [kg]	1.458	m_2 [kg]	1.057
I_{z_1} [kg m^2]	0.035	I_{z_2} [kg m^2]	0.02308
L_{G_1} [cm]	19.27	L_{G_2} [cm]	19.61
d_{G_1} [cm]	0.02	d_{G_2} [cm]	-0.22

Tabel 9.1: Dynamische parameters

Deze waarden zijn echter niet de werkelijke waarden. De waarden zijn berekend uit de modelering van de softarm zonder drukleidingen, sensoren, ... Het is niet gekend hoeveel de sensoren en allerhande leidingen zoals drukleidingen en signaalleidingen bijdragen tot de massa en het massamiddelpunt. Een identificatie van deze parameters is de opdracht die volbracht dient te worden in dit deel van het eindwerk.

9.3 Dynamisch model in functie van dynamische parameters

De vergelijking (9.9) is niet lineair in de te identificeren dynamische parameters. Dit kan men nagaan via vergelijkingen (E.1), (E.2) en (E.3) in bijlage E. Dit betekent dat opnieuw een iteratieve methode zou moeten toegepast worden om de identificatie te volbrengen. Dit is natuurlijk geen goed nieuws.

Het is echter wel mogelijk om een andere set parameters te vinden waarin vergelijking (9.9) *wel* lineair kan geschreven worden. Als deze nieuwe set parameters geïdentificeerd zijn zijn enkele van de dynamische parameters eenvoudig te vinden door een stelsel op te lossen.
Een vector θ kan gedefinieerd worden als de vector met de nieuwe set parameters die geïdentificeerd zal worden.

$$\theta = \begin{pmatrix} I_{z_1} + (d_{G_1}^2 + L_{G_1}^2)m_1 \\ I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2 \\ m_2 \\ L_{G_2}m_2 \\ d_{G_2}m_2 \\ L_{G_1}m_1 \\ d_{G_1}m_1 \end{pmatrix} \tag{9.10}$$

Dit zijn echter maar 7 parameters, terwijl er 8 dynamische parameters zijn. Deze definities voor de parameters θ vormen samen met hun gevonden waarden een stelsel van 7 vergelijkingen en 8 onbekenden. Men kan dus niet de 8 oorspronkelijke dynamische parameters vinden. Enkel voor m_2, I_{z_2}, L_{G_2} en d_{G_2} bestaat er een gesloten stelsel van 4 vergelijkingen en 4 onbekenden om deze allevier te vinden. Dit is echter niet erg, want het dynamisch model kan ook beschreven worden met de "lineaire" parameters. Het is dus niet noodzakelijk om de waardes van de oorspronkelijke parameters individueel te kennen om het koppel te berekenen.

τ in vergelijking (9.9) kan nu herschreven worden als:

$$\tau = K\,\theta \tag{9.11}$$

K wordt de observatiematrix genoemd. Het is een 2x7 matrix die het dynamisch model beschrijft in functie van de dynamische parameters θ. Deze matrix bevat enkel exact-gekend onderstelde waarden (gravitatieconstante g en lengte L_1 tussen de 2 aanwezige scharnierpunten) en opgemeten waarden van de hoeken q, hoeksnelheden $\dot{q}$ en hoekversnellingen $\ddot{q}$. Deze opgemeten waarden worden beschouwd als de input van het systeem. De matrix K is getoond in bijlage E met vergelijking E.4.

9.4 Samenvatting

In dit hoofdstuk is besproken waarom de dynamische parameters van de manipulator moeten geschat worden. Met niet-gemodelleerde effecten kan geen rekening gehouden worden.

Het dynamisch model is getoond en omgezet in een vorm die lineair is in een set parameters. Het zijn deze parameters die geïdentificeerd worden in dit deel.

Hoofdstuk 10

Offline identificatie

In dit hoofdstuk wordt besproken welk identificatie-algoritme gebruikt wordt. Dit wordt dan toegepast op data. Er wordt beschreven welke data hiervoor gebruikt wordt en hoe de data tot stand is gekomen. Vervolgens gebeurt met deze data de identificatie zoals beschreven in hoofdstuk 9. Met offline identificatie wordt bedoeld dat de identificatie gebeurt nadat de metingen gebeurd.

10.1 Identificatie-algoritme

Als identificatieschema is gekozen voor het kleinste kwadraten principe. Op te merken valt dat indien de ruis op de output groot is er een bias aanwezig kan zijn op de geëstimeerde waarden [8]. Deze bias wordt in dit eindwerk verwaarloosd. Controle via andere methoden is in dit eindwerk niet meer kunnen gebeuren.

De kostfunctie wordt via het kleinste kwadraten principe gedefinieerd als:

$$V(\theta) = \frac{1}{2} e^T e \tag{10.1}$$

$$= \frac{1}{2} (\tau_m - K\,\theta)^T (\tau_m - K\,\theta) \tag{10.2}$$

Hierin is τ_m (subscript m voor *measured*) een vector van de opgemeten output van het systeem. In dit geval zijn dat dus koppels van gelid 1 en gelid 2.

Zoals tevens in Deel II het geval is wordt het minimum gevonden door de afgeleide van de kostfunctie naar θ gelijk aan nul te stellen.

$$\frac{\partial V}{\partial \theta} = \left(\frac{\partial e}{\partial \theta} \right)^T e \tag{10.3}$$

$$= (-K)^T e \tag{10.4}$$

$$= -K^T (\tau_m - K\,\theta) \tag{10.5}$$

Vergelijking (10.5) gelijkstellen aan nul levert de waarden voor θ die de kostfunctie minimaal maken, m.a.w. de afwijkingen tussen de gemeten waarden van τ en het dynamisch model minimaal maken.

$$\hat{\theta} = \left(K^T K \right)^{-1} K^T \tau_m \tag{10.6}$$

Het numeriek uitrekenen op deze manier kan echter grote numerieke fouten teweegbrengen indien het conditiegetal κ van K groot is ($\kappa \gg 1$). Zoals reeds vermeld in paragraaf 6.4 is κ gedefinieerd als de verhouding van de grootste eigenwaarde op de kleinste eigenwaarde. Door gebruik te maken van QR-decompositie [11], LU-decompositie [13], Cholesky Decompositie [12] of Singuliere Waarden Decompositie [14] – is het mogelijk de vergelijking op te lossen naar $\hat{\theta}$ met kleinere numerieke fouten.
$\left(K^T\,K\right)^{-1}\,K^T$ komt overeen met de pseudo-inverse van K. Bij het berekenen van de pseudo-inverse via het commande *pinv(K)* in Matlab of *PseudoInverse[K]* in Mathematica wordt automatisch QR-decompositie toegepast zodat de numerieke fouten niet zo groot zijn als eerst het geval was.

10.2 Data-acquisitie

Om een identificate uit te voeren is het vanzelfsprekend dat er data nodig is. Aangezien het dynamisch model een model is dat koppels berekent moet de data nodig voor de identificatie de gemeten koppels die in de scharnierpunten van de softarm bevatten – zie vergelijking (10.6). De K-matrix in deze vergelijking is functie van de relatieve hoeken, relatieve hoeksnelheden en relatieve hoekversnellingen. Deze moeten dus ook gemeten worden.

10.2.1 Gemeten koppels

Op de softarm zijn geen koppelsensoren. Er zijn echter wel krachtsensoren aanwezig. Via de kennis van de gemeten krachten en de ontwerpparameters (zie bijlage A) is het mogelijk om de koppels die inwerken te berekenen. Volgende vergelijkingen laten toe om de 2 koppels te berekenen via de gemeten krachten F_1, F_2, F_3 en F_4, en de gemeten relatieve hoeken q_1 en q_2.

$$\tau_{m_1} = \frac{F1(0.007257\cos(q_1) + 0.011704\sin(q_1))}{\sqrt{0.023409\cos(q_1) - 0.014514\sin(q_1) + 0.102016}} - \frac{F_2(0.004439\cos(q_1) + 0.006567\sin(q_1))}{\sqrt{-0.013134\cos(q_1) + 0.008878\sin(q_1) + 0.049523}} \tag{10.7}$$

$$\tau_{m_2} = \frac{F_3(0.004376\cos(q_2) - 0.010483\sin(q_2))}{\sqrt{-0.020967864044189943\cos(q_2) - 0.008753\sin(q_2) + 0.083697}} - \frac{F_4(0.000900\cos(q_2) - 0.008550\sin(q_2))}{\sqrt{0.017100\cos(q_2) + 0.001800\sin(q_2) + 0.083025}} \tag{10.8}$$

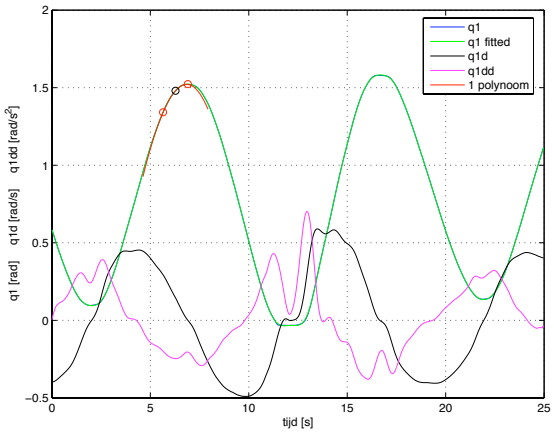
10.2.2 Hoekversnellingen

Via *hoek-encoders* worden de hoeken q_1 en q_2 gemeten. Om de hoeksnelheden $\dot{q_1}$ en $\dot{q_2}$ te meten is een elektronisch bordje beschikbaar dat de signalen van de hoek-encoders omzet naar hoeksnelheden. Om de hoeversknellingen $\ddot{q_1}$ en $\ddot{q_2}$ te meten is er echter niets beschikbaar. Eerst was het elektronisch bordje nog niet beschikbaar en was het dus ook niet mogelijk om de hoeksnelheden $\dot{q_1}$ en $\dot{q_2}$ te meten.
Daarom is een matlabprogramma gemaakt die de gemeten hoeken afleidt. Dit is echter geen sine qua non vermits de hoek-encoders geen continu meetsignaal afgeven. Ze

werken namelijk door op verschillende hoeken markeringen te tellen. Het signaal van deze encoders is dus trapsgewijs. Het is dan ook onbegonnen werk om de afgeleide met standaard formules voor numerieke afleiding te berekenen. Zeker de tweede afgeleide is problematisch.

De gebruikte methode bestaat erin om een polynoom te fitten van een gewenste graad over een gewenst aantal punten links en rechts van een bepaald punt. Dit voor elk punt doen levert een polynoom voor elk punt. De afgeleiden van een polynoom zijn analytisch eenvoudig te berekenen met kennis van de coëfficiënten die net bepaald zijn. Voor elk punt kan zo dus een afgeleide en tweede afgeleide berekend worden. Door meer punten te nemen om de polynomen te fitten is het mogelijk om gladdere afgeleiden te bekomen, maar verliest men wel informatie die misschien belangrijk is. Er dient dus via *trial and error* een gulden middenweg bepaald te worden.

Figuur 10.1 geeft een voorbeeld hoe deze afgeleiden er uitzien door gebruik te maken van een 3degraadspolynoom. De rode lijn is een voorbeeld van zo een 3degraadspolynoom met de zwarte markering als middelpunt. De rode markeringen tonen het bereik van het gebruikt aantal punten waarop de polynoom gefit is. q1d staat voor $\dot{q_1}$ en q1dd staat voor $\ddot{q_1}$.



Figuur 10.1: Berekening van afgeleiden

Vanaf de beschikbaarheid van het elektronisch bordje is het niet meer nodig om de hoeksnelheden te berekenen uitgaande van de relatieve hoeken. Het is ook aan te raden om de hoekversnellingen nu vanuit de hoeksnelheden te berekenen.

Alle metingen die nodig zijn om de koppels en de observatiematrix K te berekenen zijn nu gekend.

10.3 Databespreking

In tegenstelling tot Deel II is het hier absoluut uit den boze om data op te nemen waarbij de arm belast wordt. Het zijn immers de massatraagheidsmomenten, massa's en massamiddelpunten van de softarm die wensen bepaald te worden. Indien een last aanwezig zou zijn zouden niet deze parameters van de softarm bepaald worden maar

die van de softarm met de last erbij.

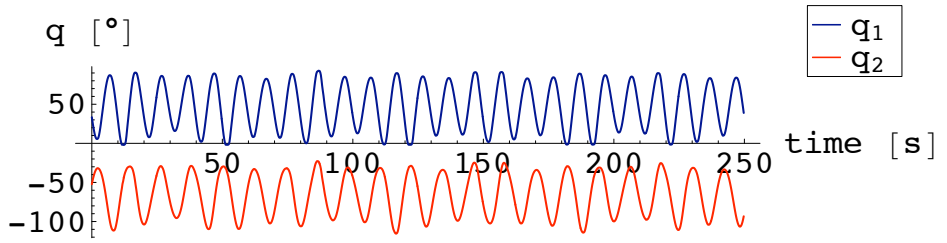
Er zullen twee reeksen data bekeken worden. Eerst zal datareeks 1 in twee gesplitst worden. Het eerste deel zal gebruikt worden voor identificatie, het tweede deel voor validatie. Voor datareeks 2 zal hetzelfde gedaan worden. Vervolgens worden beide datareeksen nog eens gevalideerd door elkaar.

10.3.1 Datareeks 1

Voor datareeks 1 is er een sinusoïdaal variërende gemiddelde druk ingesteld zoals beschreven in hoofdstuk 7. Volgende tabel toont de instellingen:

	p_m	Δp_1	Δp_2
Bias [bar]	1.5	0	0
Amplitude [bar]	0.2	1	1
Periode [s]	35	10	12

Deze data is gedurende 500s gelogd met een sampletijd van 0.01 s. De gemeten hoeken q_1 en q_2 voor de eerste 250 s zijn grafisch weergegeven in Figuur 10.2.

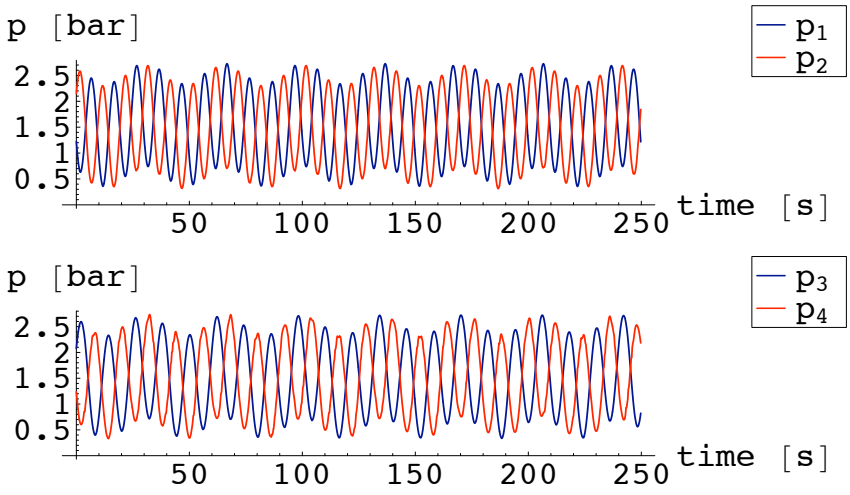


Figuur 10.2: q_1 en q_2 voor de eerste 250 s van datareeks 1

De gestuurde drukken die deze hoeken teweegbrengen zijn gegeven in Figuur 10.3.

Het is duidelijk dat de hoeken gerelateerd zijn met deze drukken. De hoeken bewegen met dezelfde periode als de drukken; de variatie van de gemiddelde druk (amplitude 0.2 bar) heeft duidelijk ook een effect op de hoeken. Dit toont aan dat de positie van de arm ook afhangt van de gemiddelde druk, zoals reeds is vermeld in paragraaf 2.2.

De koppels die berekend zijn via krachtmetingen worden verder weergegeven bij de bespreking van de identificatie.
Voor de gemeten krachten veroorzaakt door de spieren evenals voor de gegevens van seconde 250 tot 500 wordt verwezen naar bijlage F.

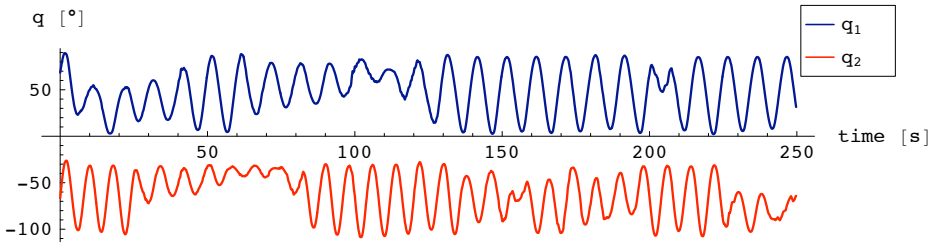


Figuur 10.3: Gestuurde drukken voor de eerste 250 s van datareeks 1

10.3.2 Datareeks 2

Voor datareeks 2 is er geen bepaalde instelling van amplitude en gemiddelde druk zoals bij datareeks 1. Hier is wat met de knoppen in de grafische interface gewerkt om zoveel mogelijk variaties te verkrijgen in de data. Enkel de periode van de beweging van de armen staat vast: voor Δp_1 is deze 10 s en voor Δp_2 is deze 8 s.

Deze data is gedurende 500s gelogd met een sampletijd van 0.01 s. De gemeten hoeken q_1 en q_2 voor de eerste 250 s zijn grafisch weergegeven in Figuur 10.4.

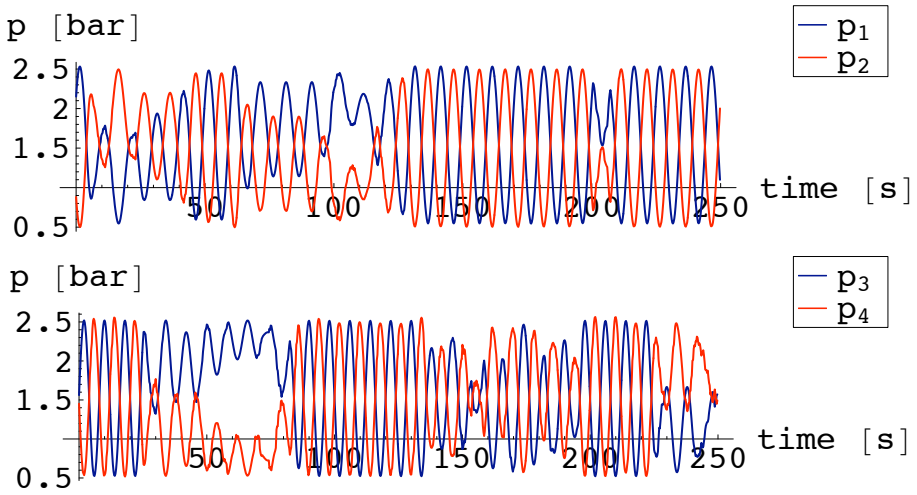


Figuur 10.4: q_1 en q_2 voor de eerste 250 s van datareeks 2

De gestuurde drukken die deze hoeken teweegbrengen zijn gegeven in Figuur 10.5.

Te zien is dat de periode waarmee de beweging van de onderarm gebeurt groter is dan die van de bovenarm.

De koppels die berekend zijn via krachtmetingen worden verder weergegeven bij de bespreking van de identificatie.
Voor de gemeten krachten veroorzaakt door de spieren evenals voor de gegevens van seconde 250 tot 500 wordt verwezen naar bijlage F.



Figuur 10.5: Gestuurde drukken voor de eerste 250 s van datareeks 2

10.4 Identificatie

10.4.1 Datareeks1

De identificatie verloopt volgens de methode beschreven in paragraaf 10.1. Toepassing van deze methode op de eerste 250s van datareeks 1 levert als resultaat voor de waarden weergegeven in Tabel 10.1. Hier zijn zowel de lineaire parameters als de parameters bekomen door het oplossen van het stelsel voor het 2de lid (zie paragraaf 9.3) weergegeven.

lineaire parameter	waarde
$I_{z_1} + (d_{G_1}^2 + L_{G_1}^2)m_1$ (kg m^2)	-2.622
$I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2$ (kg m^2)	0.341
m_2 (kg)	-0.052
$L_{G_2}m_2$ (kg m)	0.274
$d_{G_2}m_2$ (kg m)	0.017
$L_{G_1}m_1$ (kg m)	0.614
$d_{G_1}m_1$ (kg m)	-0.127

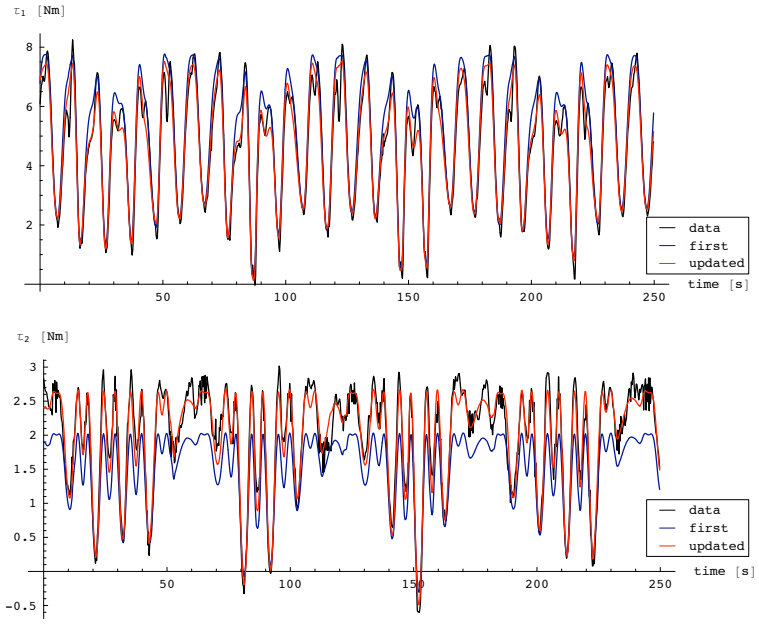
parameter	waarde
I_{z_2} (kg m^2)	1.794
m_2 (kg)	-0.052
L_{G_2} (m)	-5.290
d_{G_2} (m)	-0.325

Tabel 10.1: Dynamische parameters na offline identificatie via datareeks 1

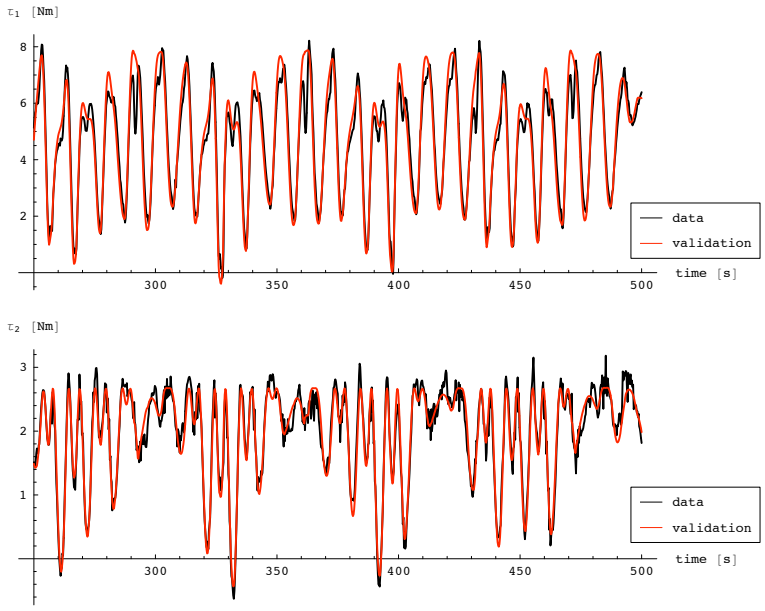
Het is duidelijk dat deze waardes niet realistisch zijn. De koppels voor gelid 1 en 2 die berekend worden gebruik makend van deze parameters worden in Figuur 10.6 getoond, samen met de koppels berekend met de oorspronkelijk geschatte parameters.

Op deze figuur is echter te zien dat ondanks de onrealistische waarden er toch een hele verbetering is voor τ_2, het koppel horend bij het tweede gelid. Voor τ_1 is er geen groot verschil. Validatie van deze parameters op het gedeelte van deze datareeks van seconde 250 tot 500 is zichtbaar in Figuur 10.7.

De validatie van deze onrealistische parameters blijkt in orde te zijn: τ_1 en τ_2

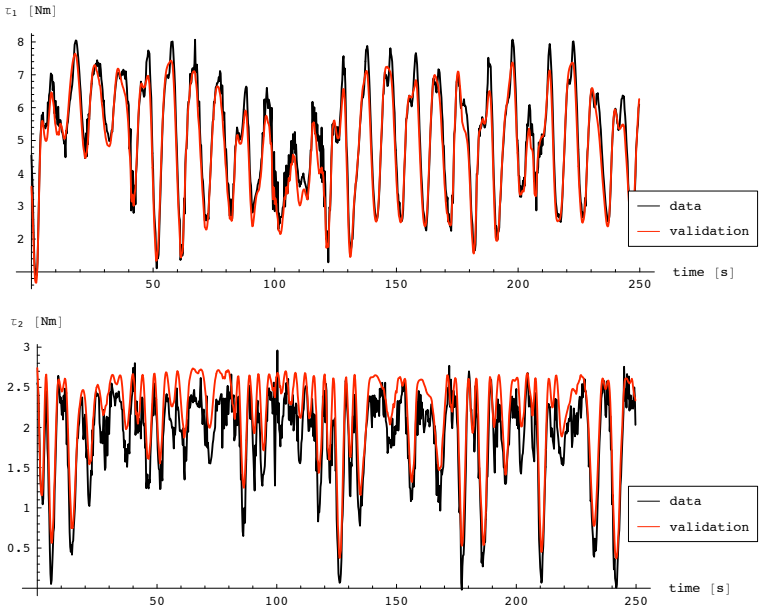


Figuur 10.6: Vergelijking van de data, het oorspronkelijk model en het geupdate model voor datareeks 1



Figuur 10.7: Validatie van het model door middel van het tweede gedeelte van datareeks 1

worden nog steeds vrij goed gevolgd. Natuurlijk is dit dezelfde soort data als de data van het eerste gedeelte waarmee de identificatie gebeurd is. Om te controleren of dit ook nog goed werkt voor andere data is de validatie met de eerste 250s van datareeks 2 ook nog gebeurd. Dit is te zien in Figuur 10.8.



Figuur 10.8: Validatie van het model door middel van het eerste gedeelte van datareeks 2

Hier wordt τ_1 nog steeds goed gevolgd maar wordt τ_2 systematisch overschat. De overschatting is echter niet totaal onaanvaardbaar groot.

10.4.2 Datareeks2

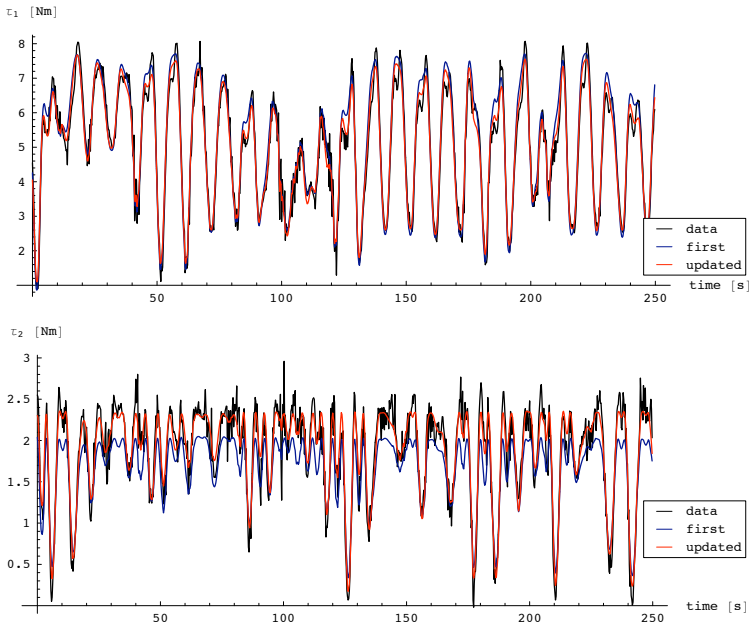
Toepassing van de identificatiemethode op de eerste 250 s van datareeks 2 levert als resultaat voor de waarden weergegeven in Tabel 10.2. Hier zijn zowel de lineaire parameters als de parameters bekomen door het oplossen van het stelsel voor het 2de lid (zie paragraaf 9.3) weergegeven.

lineaire parameter	waarde
$I_{z_1} + (d_{G_1}^2 + L_{G_1}^2)m_1$ (kg m^2)	-0.825118
$I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2$ (kg m^2)	-0.12368
m_2 (kg)	0.0936682
$L_{G_2}m_2$ (kg m)	0.237119
$d_{G_2}m_2$ (kg m)	0.00915822
$L_{G_1}m_1$ (kg m)	0.559763
$d_{G_1}m_1$ (kg m)	-0.0394783

parameter	waarde
I_{z_2} (kg m^2)	-0.724834
m_2 (kg)	0.0936682
L_{G_2} (m)	2.53147
d_{G_2} (m)	0.097773

Tabel 10.2: Dynamische parameters na offline identificatie via datareeks 2

Ook deze parameters zijn niet realistisch. De koppels voor gelid 1 en 2 die berekend worden gebruik makend van deze parameters worden in Figuur 10.9 getoond, samen met de koppels berekend met de oorspronkelijk geschatte parameters.



Figuur 10.9: Vergelijking van de data, het oorspronkelijk model en het geupdate model voor datareeks 2

Op deze figuur is echter te zien dat ondanks de onrealistische waarden er toch ook hier een hele verbetering is voor τ_2, het koppel horend bij het tweede gelid. Voor τ_1 is er geen groot verschil. Validatie van deze parameters op het gedeelte van deze datareeks van seconde 250 tot 500 is zichtbaar in Figuur 10.10.

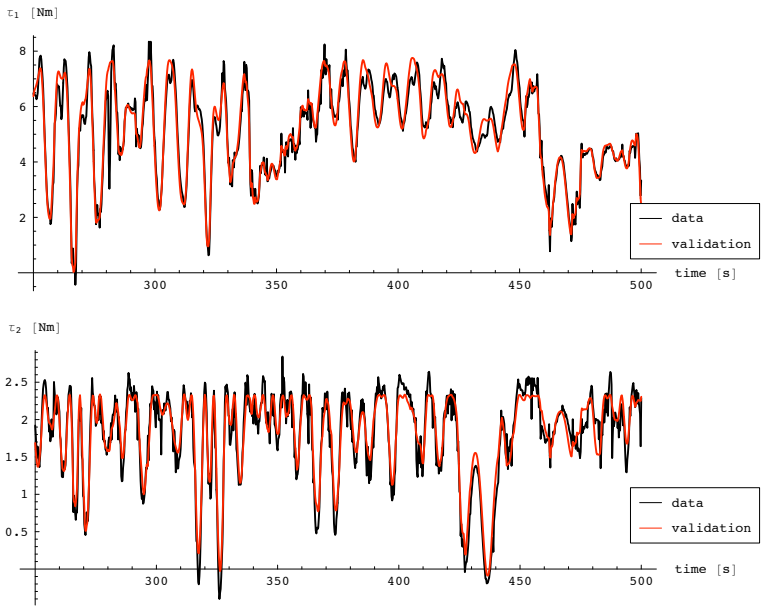
De validatie van deze onrealistische parameters gevonden via datareeks 2 blijkt ook hier in orde te zijn: τ_1 en τ_2 worden nog steeds vrij goed gevolgd. De data waarmee deze identificatie gebeurd is is wel van dezelfde reeks, maar zoals in paragraaf 10.3.2 besproken is, is de variatie in druksturing in deze reeks groot. Een tweede validatie met data van datareeks 1 is te zien in Figuur 10.11.

Hier wordt τ_1 nog steeds goed gevolgd maar wordt τ_2 wordt nu net systematisch onderschat. In vergelijking met het oorspronkelijk model is dit echter nog steeds een verbetering.

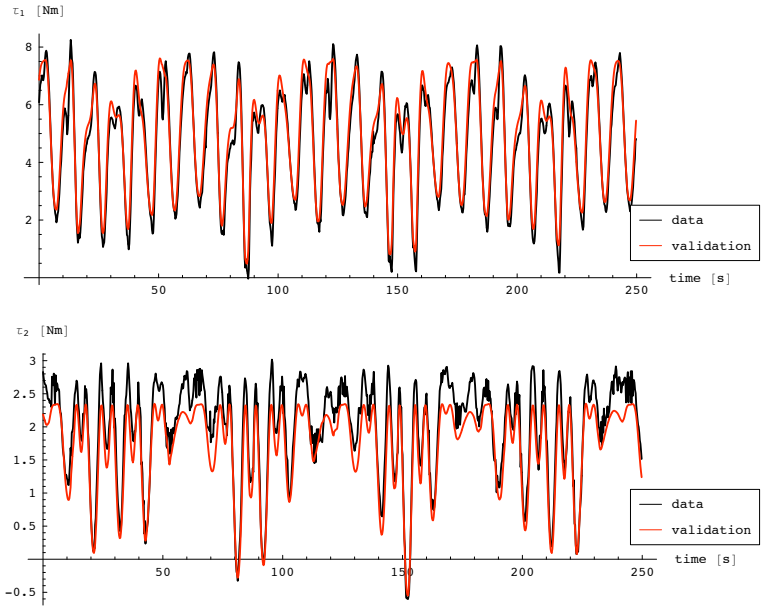
10.5 Bespreking

Met deze identificatie-resultaten is het moeilijk om een algemene conclusie te trekken. Allereerst zijn de gevonden resultaten voor datareeks 1 en de resultaten voor datareeks 2 helemaal verschillend. Deze zijn ook nog eens helemaal verschillend van de schattingen via CAD-tekeningen.

De modellen met deze resultaten komen goed overeen met de metingen. De validatie voor τ_1 is steeds goed. Er is voor τ_1 geen noemenswaardige verbetering vastge-



Figuur 10.10: Validatie van het model door middel van het tweede gedeelte van datareeks 2



Figuur 10.11: Validatie van het model door middel van het eerste gedeelte van datareeks 1

steld t.o.v. het oorspronkelijk model dat gebruik maakt van de schattingen via CAD-tekeningen. Voor τ_2 is de validatie goed indien de data die gebruikt wordt voor de validatie van dezelfde reeks is. Dit is merkwaardig; zeker voor het geval van datareeks 2 aangezien daar beide delen van deze reeks een ander patroon hebben. Dit zou eventueel te wijten kunnen zijn aan een andere instelling van de offset van één van de krachtsensoren van het tweede gelid. Het is ook mogelijk dat de verbindingslengte L_v (zie paragraaf 8.2.2) een beetje veranderd is door een moer die een beetje verschoven is ten opzichte van de andere datareeks.

Dat de waarden van de geïdentificeerde parameters zo onrealistisch zijn, en dat er zoveel verschil ligt tussen verschillende identificaties duidt op het feit dat er meerdere verschillende waarden zijn voor de parameters waarbij de data zo goed mogelijk gevolgd wordt. Naargelang enkele details in het patroon dat de data volgt kan dit aanleiding geven tot een erg verschillende set parameters.

Met de gebruikte methode is het dus niet gelukt om een realistische set parameters te vinden. De gevonden onrealistische parameters kunnen echter wel gebruikt worden, alhoewel ze tegen de principes van de fysica indruisen.

Waarom er zoveel verschillende en onrealistische resultaten bestaan kan te wijten zijn aan ongemodelleerde effecten. In het dynamisch model – vergelijking (9.9) – is namelijk geen rekening gehouden met wrijving in de gewrichten. Deze zou wel eens een niet te verwaarlozen rol kunnen spelen. De spieren trekken namelijk tussen 2 geledingen. De ontstane reactiekrachten worden dus doorgegeven aan de gewrichten waardoor de wrijving hierin groter wordt naarmate de spieren harder trekken.

Ook is er geen rekening gehouden met de verandering van massamiddelpunten door contractie van de spieren. De darmpjes voor de drukken zullen ook koppels en krachten uitoefenen. Deze kunnen zelfs veranderen bij contractie van de spieren. Er zijn enkele kabels die te kort zijn om helemaal langs de arm heen vastgemaakt te worden om dan naar hun connecties te lopen. Deze kabels zijn in Figuur 2.8 links onder te zien. Bij het optillen van de arm worden deze mee de lucht in getrokken. Deze kunnen zeker ook een invloed hebben op de identificatie.

Al deze diverse effecten kunnen een variërende invloed hebben die zorgen dat andere resultaten bekomen worden.

10.6 Conclusie

In dit hoofdstuk is besproken welke data dient te worden opgemeten. Er is uitgelegd hoe de numerieke afleiding gebeurt om de relatieve hoekversnellingen te verkrijgen die nodig zijn voor de identificatie. Dan is offline identificatie toegepast op 2 verschillende datareeksen. De resultaten zijn besproken met de mogelijke verklaringen waarom deze niet met de verwachtingen overeenstemmen.

Hoofdstuk 11

Online identificatie

11.1 Inleiding

In dit hoofdstuk wordt een andere schatter gebruikt om de dynamische parameters te identificeren. Deze schatter kan *online* gebruikt worden. Dit wil zeggen dat hij *real-time* – terwijl nieuwe data gegenereerd wordt – deze nieuwe data onmiddellijk kan gebruiken om de schattingen te updaten.

De motivatie voor deze methode is dat de methode werkt met een door de gebruiker ingegeven startwaarde voor de parameters. Dit laat zoals zal blijken iets meer controle toe op het realistisch zijn van de resultaten.

Eerst wordt de methode uitgelegd, daarna wordt deze methode toegepast op dezelfde data als gebruikt in hoofdstuk 10. Een vergelijkende studie gebeurt dan met de bekomen resultaten.

11.2 Methode

De gebruikte methode wordt ook wel naar gerefereerd als de recursieve kleinste kwadraten identificatiemethode [8]. Na elke nieuwe sample wordt er een update van de schattingen gemaakt. Dit laat toe om online data te verwerken. Door een *vergeetfactor* toe te voegen aan de kostfunctie kan de methode aangepast worden aan in de tijd variërende te schatten parameters.

In het hoofdstuk 9 werd een kleinste kwadraten formulatie gegeven om de parameters θ te identificeren:

$$\hat{\theta}(N) = \left(K_N^T\, K_N\right)^{-1}\, K_N^T\, \tau_{m_N} \tag{11.1}$$

De index N staat voor het aantal meetpunten dat gebruikt is. K_N is een $(2N \times n_\theta)$ matrix. τ_{m_N} is een $(2N \times 1)$ vector die de N metingen van τ_m bevat ($2N$ punten vermits er 2 koppels gemeten worden). Deze vergelijking kan herschreven worden als :

$$\hat{\theta}(N) = P_N\, K_N^T\, \tau_{m_N} \tag{11.2}$$

met $P_N = \left(K_N^T\, K_N\right)^{-1}$. Dit steeds een $(n_\theta \times n_\theta)$ matrix.
In [8] staat dat P_N kan geschreven worden als functie van P_{N-1}, wat deze matrix is met een zelfde dimensie maar berekend door een meting minder. Toegepast op deze

opgave is dit:

$$P_N = P_{N-1} - P_{N-1}\, K^T(N)\, \left(I_2 + K(N)\, P_{N-1}\, K^T(N)\right)^{-1}\, K(N)\, P_{N-1} \qquad (11.3)$$

De matrix $K(N)$ stelt de (2×7) observatiematrix voor zoals gedefinieerd is in vergelijking (E.4), waarbij de waarden van q, $\dot{q}$ en $\ddot{q}$ van meetpunt N gebruikt worden. I_2 is een (2×2) eenheidsmatrix.

Een andere vergelijking is in [8] gegeven die het gedeelte $K_N^T\, \tau_{m_N}$ in vergelijking (11.2) schrijft in functie van zijn oude waarde bij $(N-1)$:

$$K_N^T\, \tau_{m_N} = K_{N-1}^T\, \tau_{m_{N-1}} + K^T(N)\, \tau_m(N) \qquad (11.4)$$

$\tau_m(N)$ is de nieuwe meting van τ_1 en τ_2. Substitutie van vergelijking (11.3) en (11.4) in vergelijking (11.2) levert na wat rekenwerk een uitdrukking voor $\hat{\theta}(N)$ in functie van $\hat{\theta}(N-1)$. Deze recursieve formulatie wordt gegeven door:

$$\hat{\theta}(N) = \hat{\theta}(N-1) - \mathcal{K}_N\, \Big(K(N)\, \hat{\theta}(N-1) - \tau_m(N)\Big) \qquad (11.5)$$

$\mathcal{K}$ in deze vergelijking wordt gegeven door:

$$\mathcal{K} = P_{N-1}\, K^T(N)\, \left(I_2 + K(N)\, P_{N-1}\, K^T(N)\right)^{-1} \qquad (11.6)$$

Vergeetfactor toevoegen

Een vergeetfactor laat toe om de invloed van vroegere metingen exponentieel te laten afnemen naarmate ze ouder zijn. Zo is het mogelijk bij wijzigingen van de parameters deze wijziging te volgen in de schatting. Indien alle metingen in beschouwing zouden genomen worden met een even grote invloed zou de wijziging niet goed gevolgd worden en kan het resultaat foutief zijn. In [8] wordt aangetoond dat hiervoor slechts een zeer kleine wijziging nodig is in vorige formuleringen.

$$P_N = \frac{1}{g}\left(P_{N-1} - P_{N-1} K^T(N)\, \left(gI_2 + K(N)\, P_{N-1} K^T(N)\right)^{-1}\, K(N)\, P_{N-1}\right) \qquad (11.7)$$

$$\mathcal{K} = P_{N-1}\, K^T(N)\, \left(g\, I_2 + K(N)\, P_{N-1}\, K^T(N)\right)^{-1} \qquad (11.8)$$

g is de vergeetfactor en heeft een waarde tussen 0 en 1. Bij de waarde 1 zijn de formulaties identiek met de vorige formulaties. Hoe kleiner g, hoe sneller *vergeten* wordt (informatie gaat verloren) en dus hoe meer de wijzigingen gevolgd worden.
Het nadeel van een grote vergeetfactor is echter dat de onzekerheid op het resultaat groter wordt [8].

11.3 Implementatie

In de observatiematrix K komen metingen van q, $\dot{q}$ en $\ddot{q}$ voor. Voor online implementatie kan deze matrix niet gebruikt worden want zoals vermeld in paragraaf 10.2.2 is het niet mogelijk metingen van de hoekversnellingen te hebben. Om de redenen die in die paragraaf vermeld zijn, is het af te raden om real-time numerieke afleidingen van de hoeksnelheden uit te voeren om de hoekversnellingen te bekomen.

In [4] staat hiervoor een oplossing. Hier wordt aangetoond dat bij het filteren met een laagdoorlaatfilter van beide leden van

$$\tau_m = K(q, \dot{q}, \ddot{q})\,\theta$$

de bekomen gefilterde observatiematrix kan geschreven worden als $K_f(q, \dot{q})$, waarin de hoekversnellingen niet meer voorkomen zodanig dat de methode dan kan toegepast worden op:

$$\tau_{m_f} = K_f(q, \dot{q})\,\theta \tag{11.9}$$

11.3.1 Filteren van de observatiematrix

De gebruikte filter is een laagdoorlaatfilter met als karaktersitiek in het Laplacedomein:

$$\text{TF} = \frac{\lambda}{s + \lambda} \tag{11.10}$$

De waarde voor λ dient zodanig gekozen te worden dat de bandbreedte zo is dat de filter de signaalfrequenties doorlaat, maar de ruis wegfiltert. De gekozen waarde voor λ is 2.5 .

Online filteren van de ongefilterde observatiematrix is niet mogelijk aangezien dan de hoekversnellingen vereist zijn voor de berekening van deze ongefilterde matrix. Het is nodig een uitdrukking voor K te hebben die onafhankelijk is van $\ddot{q}$.

De oplossing dient zich aan door de filtering in het tijdsdomein uit te werken [4]. De filtering toepassen op vergelijking (9.9) levert:

$$\int_0^t w(t-r)\tau(r)dr = \int_0^t w(t-r)\,[H\,\ddot{q} + C\,\dot{q} + G]\,dr \tag{11.11}$$

$w(t)$ is het impulsantwoord van de laagdoorlaat filter en is gelijk aan $w(t) = e^{-\lambda\,t}$. Enkel de eerste term van het tweede lid is functie van $\ddot{q}$. Het is dus deze term waar geen online filtering op kan toegepast worden. De term kan herschreven worden door er partiële integratie op toe te passen [4]:

$$\begin{aligned}\int_0^t w(t-r)\,[H\,\ddot{q}]\,dr &= \Big[w(t-r)\,H\dot{q}\Big]_0^t - \int_0^t \frac{d}{dr}\,[w(t-r)\,H]\;\dot{q}\,dr \\ &= w(0)\,H(q)\,\dot{q} - w(t)\,H(q_0)\,\dot{q}_0 - \int_0^t w(t-r)\,H\,\dot{q} - \dot{w}(t-r)\,H\,\dot{q}\;\;dr\end{aligned} \tag{11.12}$$

In deze vergelijking stellen q_0 en $\dot{q}_0$ de snelheid en hoeksnelheid voor bij $t = 0$. Zoals te zien is is er geen afhankelijkheid meer van $\ddot{q}$. De eerste 2 termen in deze vergelijking zijn in feite reeds gefilterde delen. $\dot{w}(t-r)$ kan geschreven worden – gebruik makend van zijn definitie – als $\big(-\lambda\,w(t-r)\big)$. De laatste 2 termen zijn ook geen functie meer van $\ddot{q}$. Het is dus wel mogelijk om de filtering van deze delen online uit te voeren.

Toepassing van deze redenering leidt tot een geslaagde methode om een gefilterde observatiematrix te bekomen zonder gebruik te maken van $\ddot{q}$.

11.3.2 Implementatie

Alhoewel deze methode als een online schatter bedoeld is, is hij nog niet geïmplementeerd om rechtstreeks data van de softarm te gebruiken. Voor onderzoeksdoeleinden is er echter gekozen dezelfde data te gebruiken als voor de offline identificatie.

De methode is geïmplementeerd in Simulink. De Simulinkschema's die de uitwerking tonen zijn toegevoegd in bijlage G.

De gemeten data wordt eerst in de Matlab-workspace geladen. Van daaruit wordt *gestreamed* naar Simulink, alsof de data real-time gemeten is. Het is heel eenvoudig dit te implementeren om te doen werken met real-time data van de softarm door enkele Simulink-blokken te vervangen door de gemeten input.

Via *scopes* is het mogelijk tal van signalen te visualiseren. Er kan bekeken worden hoe de filtering gebeurt van de koppels, hoe het recursief geupdate model overeenstemt met deze gefilterde koppels, en hoe de parameters variëren in de tijd door de recursieve identificatie.

11.4 Identificatie

De gebruikte datareeksen zijn reeds beschreven in paragraaf 10.3. Er zal enkel een identificatie gebeuren gebruik makend van datareeks 2 aangezien deze de grootste variaties toont.

11.4.1 Korte identificatie

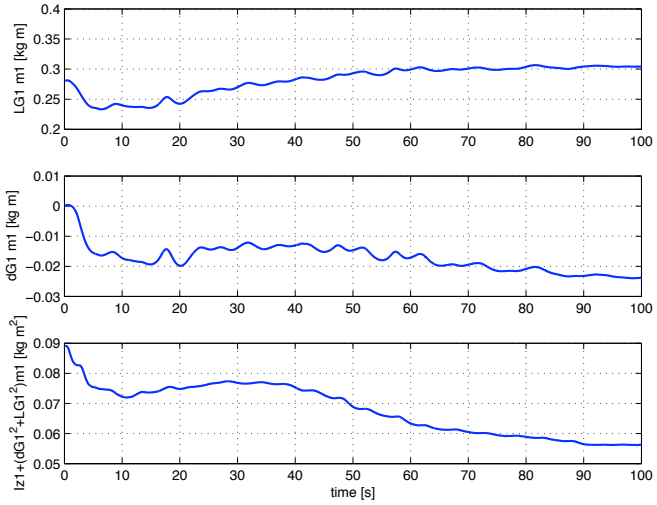
Van datareeks 2 worden eerst de eerste 100s gebruikt voor de identificatie, met een sampletijd van 0.01 s. Aangezien het een recursief algoritme is dat gebruikt wordt is het mogelijk de evolutie van de parameters in de tijd te bekijken. Als beginvoorwaarde voor θ worden de waarden gekozen die geschat zijn via de CAD-tekeningen – zie Tabel 9.1. In Figuur 11.1 wordt de evolutie van de parameters van de eerste link getoond in functie van de tijd; in Figuur 11.2 die voor de tweede link. De waarden van de parameters die na 100 s gevonden zijn, zijn te zien in Tabel 11.1

lineaire parameter	waarde
$I_{z_1} + (d_{G_1}^2 + L_{G_1}^2)m_1$ (kg m^2)	0.0563
$I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2$ (kg m^2)	-0.0734
m_2 (kg)	0.964
$L_{G_2}m_2$ (kg m)	0.2258
$d_{G_2}m_2$ (kg m)	0.0258
$L_{G_1}m_1$ (kg m)	0.3043
$d_{G_1}m_1$ (kg m)	-0.0237

parameter	waarde
I_{z_2} (kg m^2)	-0.1270
m_2 (kg)	0.964
L_{G_2} (m)	0.2342
d_{G_2} (m)	0.0268

Tabel 11.1: Dynamische parameters na online identificatie via datareeks 2 na 100 s

Deze waarden zijn *wel* realistisch, met uitzondering van de parameter I_{z_2}. In Figuur 11.3 wordt getoond hoe tijdens de online identificatie het wiskundig model en de data beter en beter met elkaar overeenkomen met verloop van de tijd.



Figuur 11.1: Parameterevolutie voor link 1

Validatie

Om deze gevonden parameters te valideren is zoals bij de offline identificatie seconde 250 tot 500 van dezelfde datareeks genomen (datareeks 2). De vergelijking van het geupdate wiskundig model en de data is getoond in Figuur 11.4.

De validatie is in dit geval goed. Zeker de data van τ_1 wordt zoals steeds goed gevolgd. Voor τ_2 zijn er een paar afwijkingen in enkele pieken te zien. Het geupdate model voor τ_2 is veel beter dan het oorspronkelijk model, dat getoond is in Figuur 10.9.

11.4.2 Langere identificatie

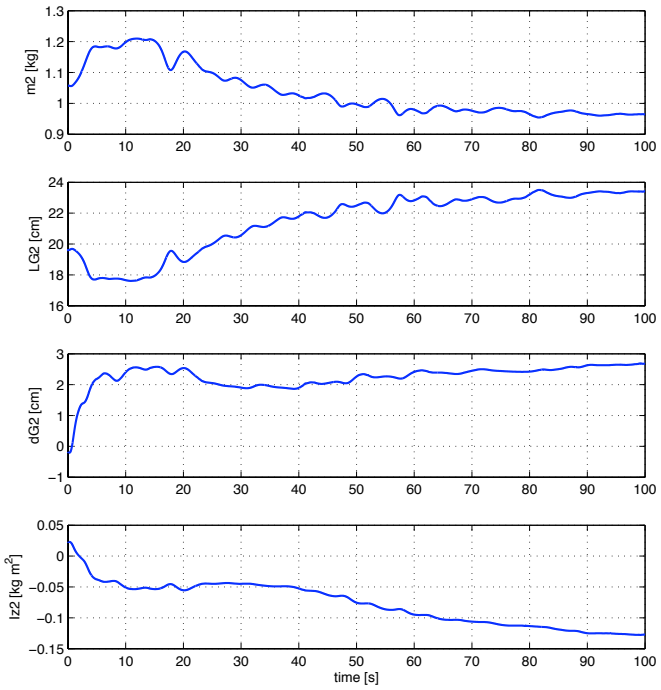
Indien de identificatie gebeurt over de volle 500 s van de data zijn de resultaten merkwaardig genoeg toch nog vrij verschillend. Ze zijn iets minder realistisch. De parameters hebben dan waarden zoals gegeven in Tabel 11.2.

lineaire parameter	waarde
$I_{z_1} + (d_{G_1}^2 + L_{G_1}^2)m_1$ (kg m^2)	0.0009
$I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2$ (kg m^2)	-0.2075
m_2 (kg)	0.2827
$L_{G_2}m_2$ (kg m)	0.2304
$d_{G_2}m_2$ (kg m)	0.0151
$L_{G_1}m_1$ (kg m)	0.4848
$d_{G_1}m_1$ (kg m)	-0.0152

parameter	waarde
I_{z_2} (kg m^2)	-0.3961
m_2 (kg)	0.2827
L_{G_2} (m)	0.8150
d_{G_2} (m)	0.0534

Tabel 11.2: Dynamische parameters na online identificatie via datareeks 2 na 500 s

Deze zijn inderdaad erg verschillend met die wanneer maar 100 s gemeten zijn. Na de 100 s verwijderen de parameters zich weer van de waarde waarnaar ze eerst



Figuur 11.2: Parameterevolutie voor link 2

geëvolueerd waren. De verbeteringen die het model met deze parameters met zich meebrengt zijn echter minimaal ten opzichte van het model met parameters die slechts na 100 s gevonden zijn.

Een mogelijke verklaring is dat na 100 s meer en meer de ongemodelleerde effecten proberen ingecalculeerd te worden.

11.4.3 Validatie met datareeks 1

De gevonden parameters kunnen ook gevalideerd worden met datareeks 1. Dit geeft echter net zoals bij de offline identificatie een onderschatting (zie Figuur 10.11).

Dit doet vermoeden dat er bij het meten van de data een kleine kracht-offset fout moet opgetreden zijn bij één of meerdere van de krachtmetingen, ofwel dat de verbindingslengte L_v (zie paragraaf 8.2.2) van een spier een beetje veranderd is door het verschuiven van een moer. Dit kan enkel nagekeken worden door veel meer datareeksen op te nemen, en de validatie voor elk van die reeksen uitvoeren. Zo kan er statistisch gezien worden waar deze onder- of overschattingen zich voordoen.

Dit is echter nog niet gebeurd, en dient nog te gebeuren om een parameterset te identificeren die het dynamisch model zo goed mogelijk doet overeenstemmen met de werkelijkheid.

11.4.4 Gebruik van de vergeetfactor

Het gebruik van een vergeetfactor toegepast op deze data brengt geen verklaringen aan het licht. Met de vergeetfactor werd gehoopt te kunnen zien hoe de massa's en massamiddelpunten variëren in de tijd. Dit heeft natuurlijk enkel nut indien deze realistische waarden hebben. Met gebruik van de vergeetfactor variëren de parameters zoveel dat er sprongen zijn voor bvb. m_2 van 1 kg naar 4000 kg.

Dit toont aan dat – zoals reeds vermoed werd na hoofdstuk 10 – er meerdere sets waarden zijn voor de dynamische parameters die alle ongeveer een zelfde model beschrijven. Dit kan o.a. te wijten zijn aan ongemodelleerde effecten zoals wrijving, of enkele parameters die bij een kleine wijziging een groot effect hebben op de overige parameters van de set.

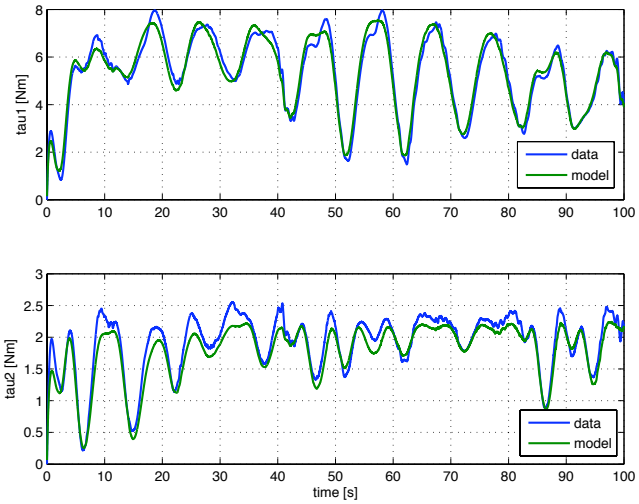
11.5 Conclusie

In dit hoofdstuk is een online schatter gebruikt om de identificatie uit te voeren. Door de mogelijkheid een beginwaarde op te leggen zijn de resultaten via deze methode realistischer dan bij de offline-methode die besproken is in hoofdstuk 10.

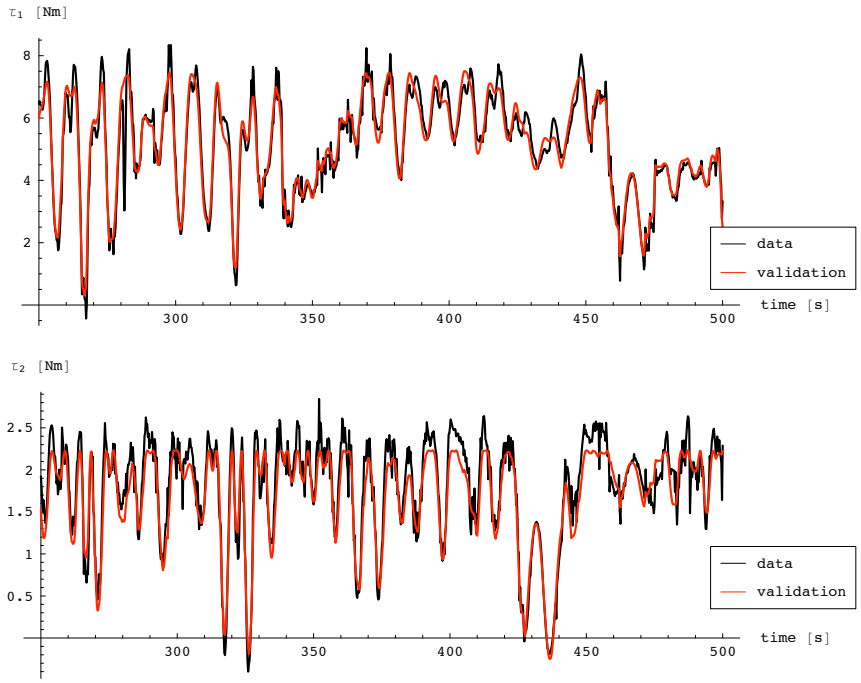
Voor dezelfde datareeks is de validatie in orde. Bij validatie met een andere datareeks is het voor de tweede link niet helemaal in orde. Dit kan te maken hebben met foutief gemeten data. Door meerdere datareeksen te gaan valideren kan hier een betere conclusie uit getrokken worden. De gevonden parameters geven in ieder geval een verbetering ten opzichte van de oorspronkelijk gebruikte parameters.

Gebruik van de vergeetfactor heeft geen enkel nut. Bij een waarde hiervoor, waarbij zelfs maar een klein beetje vergeten wordt, worden de parameters onrealistisch. Er is geen gulden middenweg gevonden voor een vergeetfactor die kan aantonen hoe de parameters variëren op een realistische wijze.

Er is geen groot verschil in het *tracken* van de data via het wiskundig model wanneer er nu slechts 100 s geïdentificeerd wordt of 500 s, alhoewel de parameters hierbij toch wel vrij verschillend zijn.



Figuur 11.3: Evolutie van de berekende koppels tijdens online identificatie



Figuur 11.4: Validatie van het model met verdere data van dezelfde reeks

Hoofdstuk 12

Conclusies en toekomstperspectieven

De belangrijkste punten die bereikt zijn bij de identificatie van de softarm zijn als volgt samen te vatten:

Deel *Identificatie van de actuatoren*

- Om de spierfuncties te optimaliseren is een identificatiemethode toegepast die de parameters L en R in deze functies zo geschat heeft dat de spierfuncties de werkelijkheid zo goed mogelijk benaderen.
- De toegepaste methode maakt gebruik van data die gemeten is zonder dat de spieren uit de softarm genomen worden. Zo is de data die gebruikt wordt opgenomen in dezelfde omstandigheden als de normale werking van de softarm.
- De gebruikte methode is de iteratieve Levenberg-Marquardt methode. Deze methode bezit het voordeel van een klein aantal nodige iteraties waardoor de rekentijd aanvaardbaar is.
- Toepassing van deze methode op allevier de spieren levert bevredigende resultaten op. Gebruik van de zo bekomen spierfuncties zou de controle van de softarm een heel stuk moeten verbeteren.

Deel *Identificatie van de manipulator*

- Het dynamisch model van de manipulator is omgevormd tot een model dat lineair is in de parameters die geoptimaliseerd werden zodanig dat het model het gedrag van de manipulator zo goed mogelijk voorspelt.
- Eén van de toegepaste methodes is een kleinste kwadraten schatting. De resultaten met deze methode zijn onrealistisch, maar zorgen wel voor een model dat de gebruikte data goed kan voorspellen voor 1 gebruikte datareeks en iets minder goed voor een andere.
- De andere toegepaste methode is een recursieve kleinste kwadraten schatting die real-time kan gebeuren. De resultaten via deze schatter zijn realistischer maar nog steeds niet zodanig dat men er tevreden van kan zijn. Hetzelfde geld als voor

de kleinste kwadraten schatter: voor één datareeks kan het gevonden model de data goed voorspellen, voor een andere datareeks iets minder goed. Dit doet vermoeden dat meer datareeksen moeten opgenomen worden om zich ervan te vergewissen dat één van de datareeksen niet foutief gemeten is.

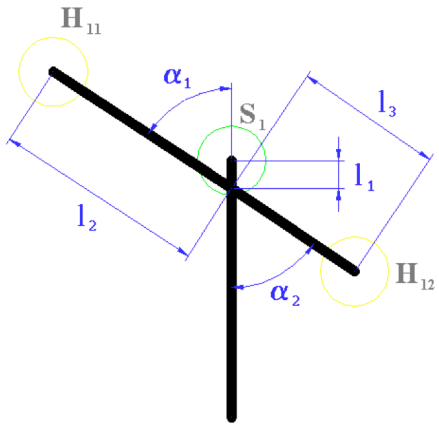
- Er is nog verdere studie nodig om tot een resultaat te komen waar men algemeen tevreden over kan zijn. De recursieve methode lijkt hiervoor een stap in de goede richting te zijn.

Bijlage A

Ontwerp

A.1 Sokkel

De sokkel is ontworpen met een tekening gegeven in Figuur A.1 en overeenkomstige parameters gegeven in Tabel A.1.
H's komen overeen met aanhechtingspunten, S met scharnierpunten



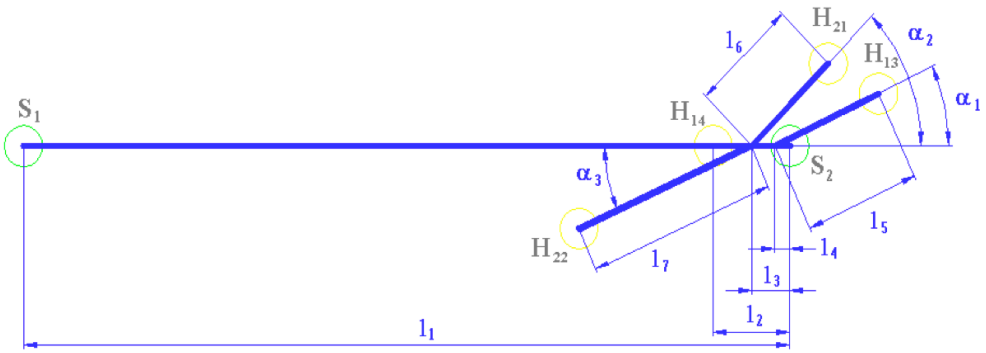
Figuur A.1: Ontwerp van de sokkel

Parameter	Oorspronkelijk ontwerp	Schaalmodel
l_1 (mm)	20	6
l_2 (mm)	155	46.5
l_3 (mm)	108	32.4
α_1 (°)	57	57
α_2 (°)	56	56

Tabel A.1: Ontwerpparameters voor de sokkel

A.2 Bovenarm

De bovenarm is ontworpen met een tekening gegeven in Figuur A.2 en overeenkomstige parameters gegeven in Tabel A.2.



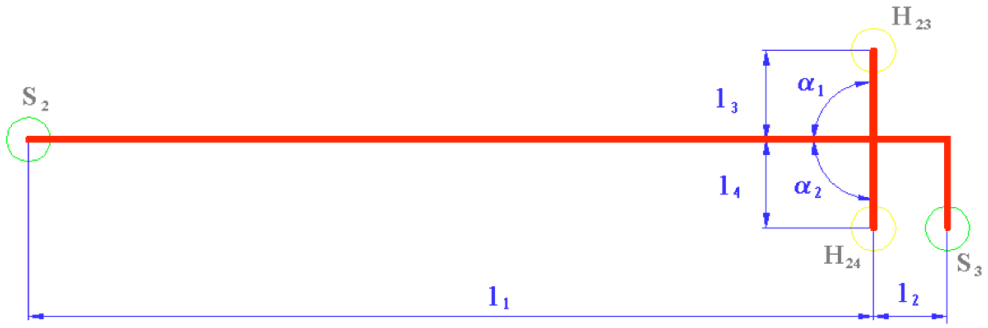
Figuur A.2: Ontwerp van de bovenarm

Parameter	Oorspronkelijk ontwerp	Schaalmodel
l_1 (mm)	1000	300
l_2 (mm)	100	30
l_3 (mm)	50	15
l_4 (mm)	20	6
l_5 (mm)	150	45
l_6 (mm)	141.4	42.42
l_7 (mm)	246.2	73.86
α_1 (°)	25	25
α_2 (°)	45	45
α_3 (°)	24	24

Tabel A.2: Ontwerpparameters voor de bovenarm

A.3 Onderarm

De bovenarm is ontworpen met een tekening gegeven in Figuur A.3 en overeenkomstige parameters gegeven in Tabel A.3.



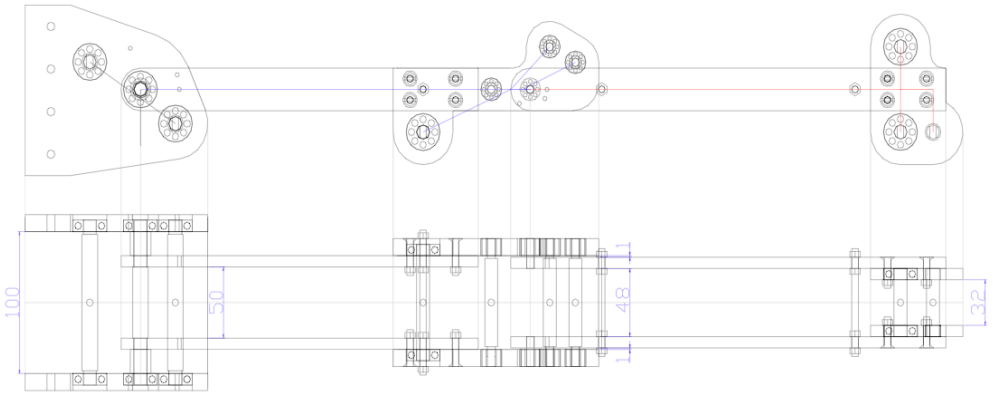
Figuur A.3: Ontwerp van de onderarm

Parameter	Oorspronkelijk ontwerp	Schaalmodel
l_1 (mm)	950	285
l_2 (mm)	50	25
l_3 (mm)	100	30
l_4 (mm)	100	30
α_1 (°)	90	90
α_2 (°)	90	90

Tabel A.3: Ontwerpparameters voor de onderarm

A.4 Samenbouw

Samenbouw van de sokkel, bovenarm en onderarm voor het schaalmodel is getoond in Figuur A.4



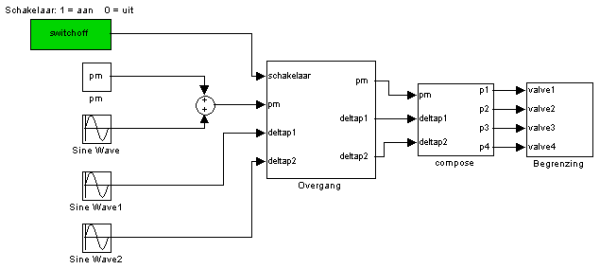
Figuur A.4: Samenbouw van sokkel, bovenarm en onderarm

Bijlage B

Communicatie met operator

B.1 Simulink schema's

B.1.1 Druksturingsgedeelte



Figuur B.1: Totale druksturingsgedeelte

Overgang

Het overgangsgedeelte in Figuur B.1 is in detail bekeken in Figuur B.2

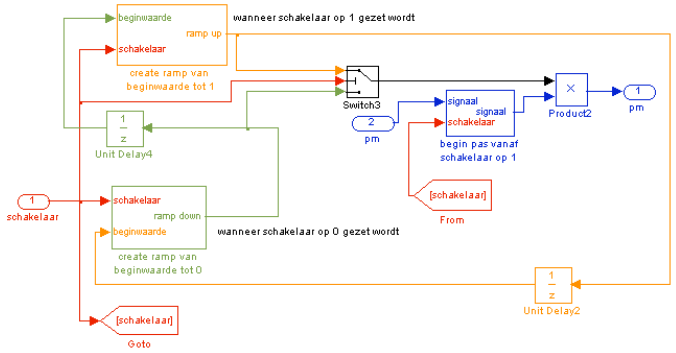
Dit is slechts getoond voor het signaal p_m, de gemiddelde druk. Het blok *Overgang* in Figuur B.1 bevat tevens dezelfde sturing als Figuur B.2 voor Δp_1 en Δp_2. In dit blok komen subsystemen voor genaamd "ramp up". Deze geven een stijgende waarde in de tijd die geactiveerd wordt door een schakelaar. Als de schakelaar op 1 gezet wordt, zal de "ramp" beginnen, vertrekkend van een "beginwaarde" op dat moment tot de waarde 1 bereikt is. Hoe dit in zijn werk gaat is getoond in Figuur B.3

Compose

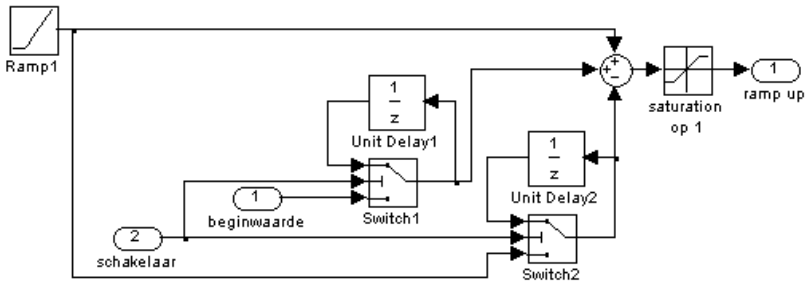
Dit blok zet de drukken p_m, Δp_1 en Δp_2 om in drukken die naar de spieren afzonderlijk kunnen worden gestuurd.

Begrenzing

Dit blok zal de drukken die gestuurd worden begrenzen op 3 bar. Dit is een beveiliging.



Figuur B.2: Overgangsgedeelte uit Figuur B.1



Figuur B.3: Het blok ramp up uit Figuur B.2

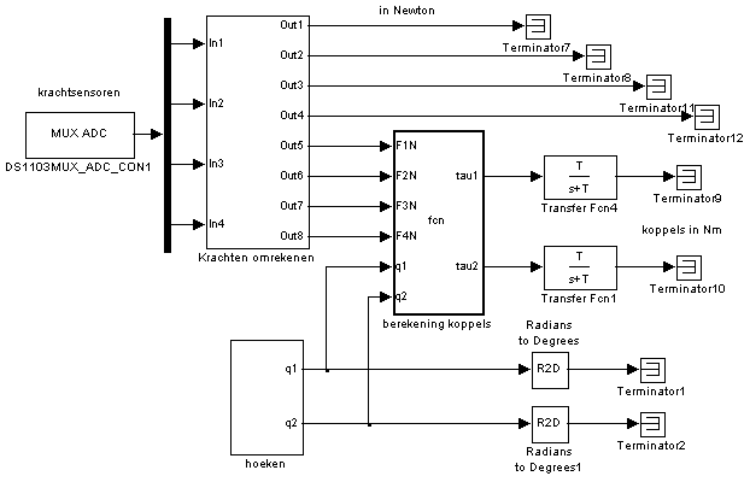
B.1.2 Krachtgedeelte

Om de krachten te kunnen inlezen is het nodig om ze eerst in simulink te laten invoegen. Zo weet de sturingseenheid "dSPACE" dat hij deze moet opmeten. Figuur B.4 bevat het gedeelte dat de krachten inleest. Tevens bevat deze figuur het gedeelte dat de hoeken q_1 en q_2 inleest. Met deze inlezingen en ontwerpparameters van de softarm kan dan het koppel in de scharnieren berekend worden om real-time te kunnen volgen. Omdat dit enkel voor visualisatie-doeleinden gebruikt wordt is dit gefilterd met een laagdoorlaat filter. De gebruikte tijdsconstante T is gezet op 4.

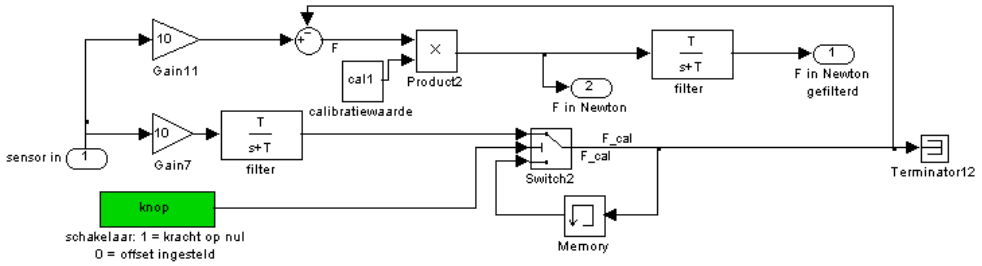
Offset resetten

Zoals beschreven in dit eindwerk is er een offset op de sensoren eens ze bevestigd zijn. Om deze offset te resetten is er binnen in het blok "krachten omrekenen" in Figuur B.4 een schema gemaakt dat getoond is in Figuur B.5.

In dit blok wordt tevens de kracht omgerekend naar de eenheid Newton, en ook gefilterd voor visualisatie-doeleinden. Om te kunnen tonen in het programma ControlDesk is het slechts nodig om een signaal een label te geven. Het signaal met het label "F" benoemen zal dus volstaan om deze data te kunnen "loggen". De filtering gebeurt met dezelfde laagdoorlaatfilter als gebruikt voor het koppel in Figuur B.4.



Figuur B.4: De inlezing en verwerking van krachten

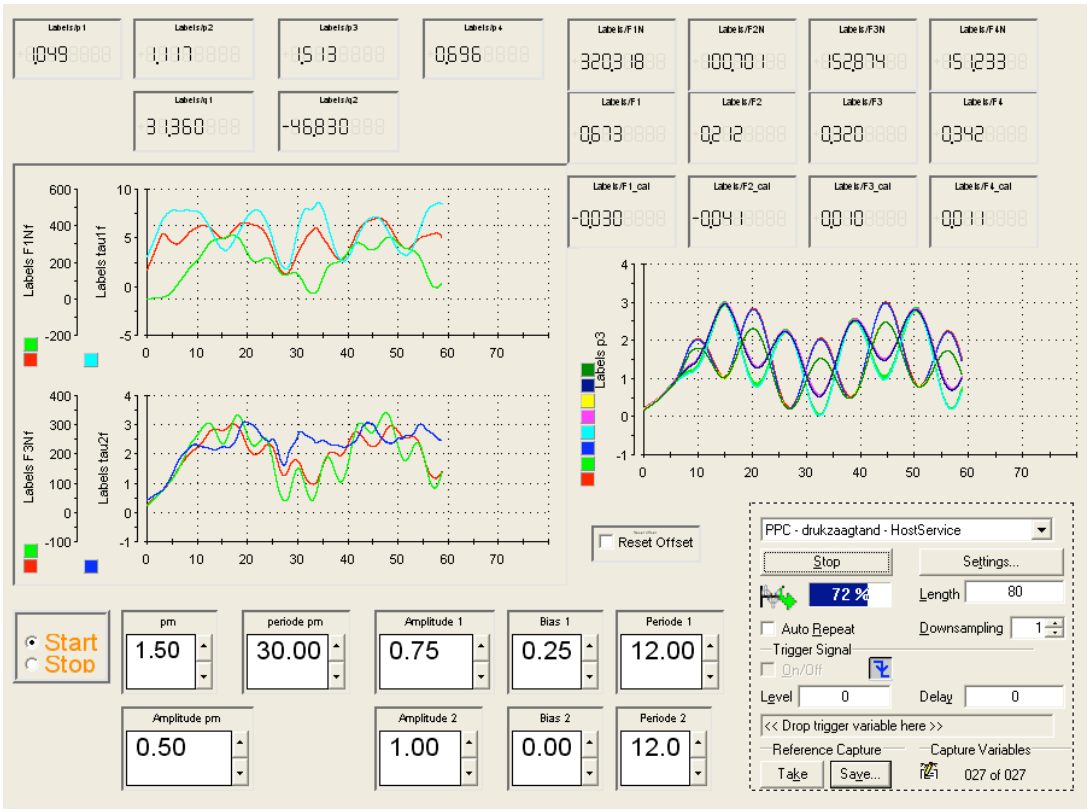


Figuur B.5: "Krachten omrekenen" in Figuur B.4

B.2 ControlDesk

B.2.1 Grafische interface

ControlDesk verzorgt de grafische interface. Een screenshot van deze interface is te zien in Figuur B.6.



Figuur B.6: De grafische interface via ControlDesk

In deze figuur zijn aan de bovenkant links numerieke waarden te zien van de drukken in de spieren en de gemeten relatieve hoeken q_1 en q_2. Rechtsboven zijn 3 rijen numerieke waarden die overeenstemmen met respectievelijk de kracht in elke spier in Newton, de sensorwaarden voor de kracht, en de waarden voor de offset in sensorwaarden. Met de knop in het midden "Reset Offset" is het mogelijk deze offset te resetten.

Links zijn opgemeten krachten gevisualiseerd in Newton voor elke spier van de softarm. Bovenste figuur visualiseert de eerste link, onderste figuur de tweede link. De krachten zijn de gefilterde versies (zie paragraaf B.1.2). Voor elke link zijn in deze figuren ook de koppels getoond die in de scharnieren van elk gelid werken. Deze worden berekend via de krachtmetingen en kennis van de ontwerpparameters van de softarm.

De rechterfiguur laat toe om vele andere variabelen te bekijken. Hier is nu de druk bekeken in elke spier in functie van de tijd. Het is eenvoudig om andere variabelen te bekijken.

De Start/Stop knop onderaan links dient om de druk die gestuurd wordt af te zetten. Dit is de schakelaar die in de simulinkschema's voor de druksturing getoond is. Deze knop activeert de overgang.

Naast deze knop zijn alle instelbare parameters gegeven om de druksturing te manipuleren. p_m stuurt de gemiddelde relatieve druk in bar. "periode p_m" stuurt de periode van een sinusgolf die op deze gemiddelde druk gesuperponeerd wordt met een amplitude gelijk aan "Amplitude p_m".
Om Δp_1 en Δp_2 te sturen worden respectievelijk de eerste en tweede rij gebruikt naast het gedeelte voor de gemiddelde druk p_m.

Het omstippelde gedeelte is het captatiesysteem. Door op de stopknop hierin te drukken zal het loggen gestopt worden en stopt de visualisatie. Er kan ingesteld worden hoeveel seconden moet gelogd worden (hier 80 s). Er kan ook aan Downsampling gedaan worden voor het loggen.
De knop "Save" laat toe om de gelogde data te bewaren in een ".mat bestand".

B.3 Matlab

In Matlab zijn enkele programma's geschreven die de verwerking toelaten om ".csv bestanden" te creëren. Hier volgt een kort overzicht van de geschreven programma's die dit bewerkstelligen.

Dataconvert Dit programma roept eerst het programma *Dataload* aan, dan *Dataprocess* en vervolgens *Datasave*.

Dataload Deze geeft een dialoogvenster dat vraagt naar de locatie van het ".mat bestand" en bewaart de data hiervan in de workspace.

Dataprocess Dit programma wordt aangeroepen nadat in *Dataconvert* de variabele "aantal" een waarde heeft gekregen die overeenstemt met het gewenst aantal punten dat *Mathematica* zal gebruiken voor de identificatieprocedure. *Dataprocess* gebruikt de data gehaald uit *Dataload*. Deze data is echter in "structures" bewaard. Het is een ingewikkelde zaak om hier de waardes voor F_1, q_1 en dergelijke uit te halen, of alles wat men maar wenst. Dit gebeurt door na te kijken wat de labels zijn die meegegeven zijn in de *structures*. Op deze manier is men altijd zeker dat de data hier uitgehaald wordt wel degelijk overeenstemt met de data die men verwacht. Zo maakt het niet uit in welke volgorde ControlDesk de data bewaart in het ".mat bestand".

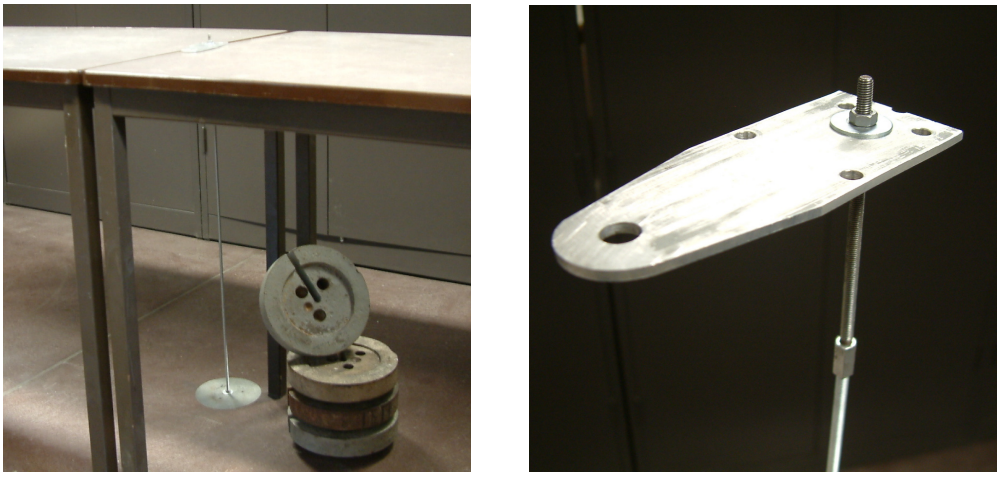
Datasave zal een dialoogvenster openen met de vraag waar het ".csv bestand moet bewaard worden en zal dit dan ook uitvoeren.

polyfitderiv Dit is een programma dat gebruikt wordt in *Dataprocess* dat numerieke afleiding zal uitvoeren op de hoeksnelheid $q_1 d$ en $q_2 d$ om de tweede afgeleide van q_1 en q_2 te zoeken. Dit wordt gebruik om het koppel te berekenen via het dynamisch model (Deel III). Er wordt gebruik gemaakt van *polyfitself* wat hetzelfde is als de matlabfunctie *polyfit* maar enkele vervelende warnings uitschakelt.

Bijlage C

Calibratieopstelling

Figuur C.1 toont de opstelling die gebruikt is om de calibratie van de krachtsensoren te verrichten. De gewichten zijn elk 10kg.



(a) volledige opstelling (b) Uitvergroting van ophangstuk

Figuur C.1: Calibratieopstelling voor krachtsensoren

Om de sensoren van het type LTH300 te calibreren moet de sensor bovenop het ophangstuk geplaatst worden en via een rondel en moer bovenop wordt deze dan tussen het ophangstuk en rondel geklemd wanneer er aan de draadstang getrokken wordt door de gewichten aan te brengen op de opstelling. Zo ondervindt de sensor een kracht die te berekenen is. Er mag geen moer aanwezig zijn aan de onderkant van het ophangstuk zodat er geen voorspanning wordt gezet op de krachtsensor.

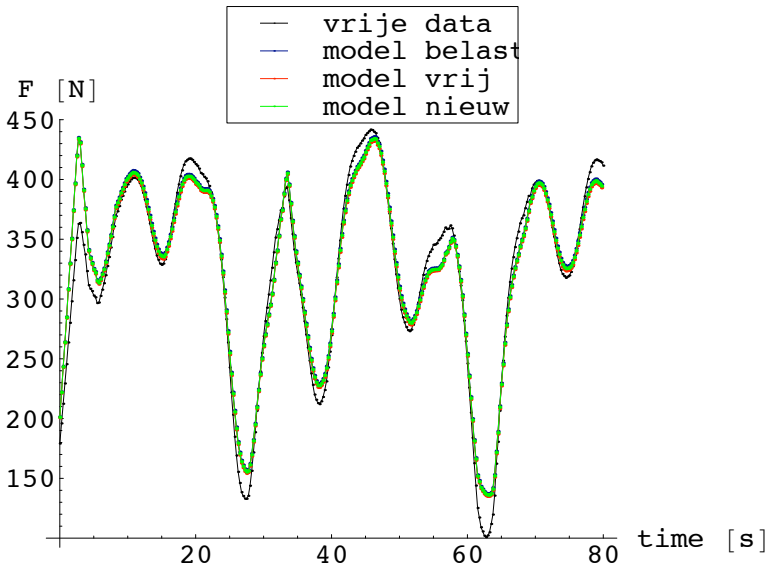
De sensor van het type LCM300 kan gemonteerd worden tussen twee draadstangen. Zoals in de uitvergroting van het ophangstuk kan gezien worden is de draadstang op te splitsen in twee stukken. Hiertussen kan deze sensor dan gemonteerd worden. Bij toevoegen van gewichten zal de sensor een trekkracht ondervinden die te berekenen is.

Bijlage D

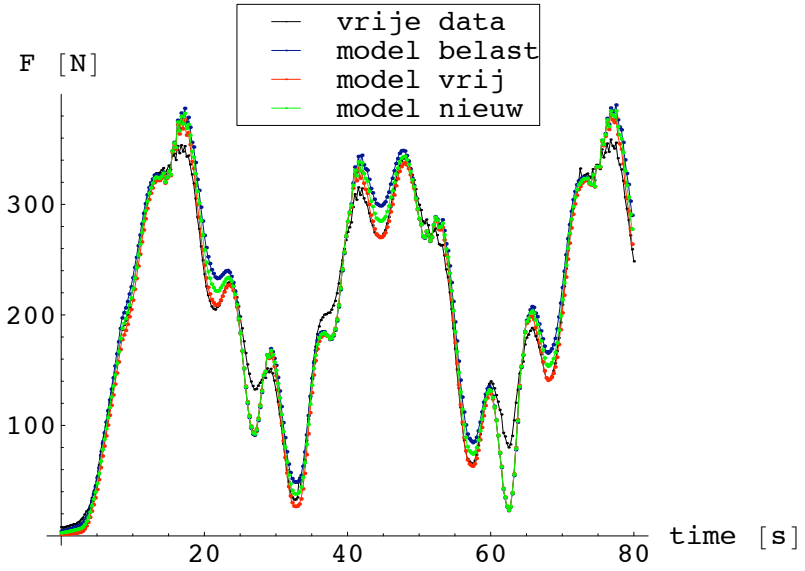
Validatie identificatie spieren

De figuren in deze bijlage tonen de validatie van de geupdate modellen gevonden door gebruik te maken van belaste data en dit model te testen op vrije data. Er is tevens een gemiddelde waarde genomen van de parameters gevonden met vrije en belaste data. Het is namelijk dit gemiddelde dat in dit eindwerk aangeraden is waarde te nemen voor de parameters van de spierfuncties.

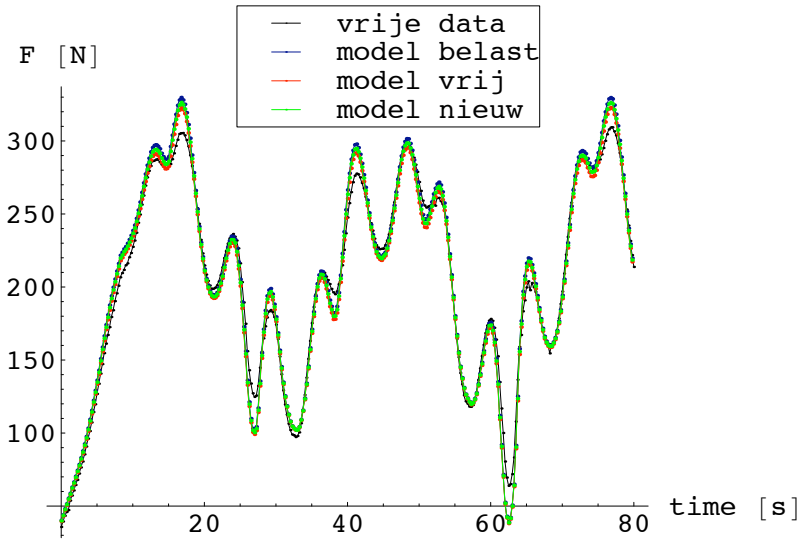
In volgende figuren is de gemiddelde waarde in het groen aangeduid met "model nieuw".



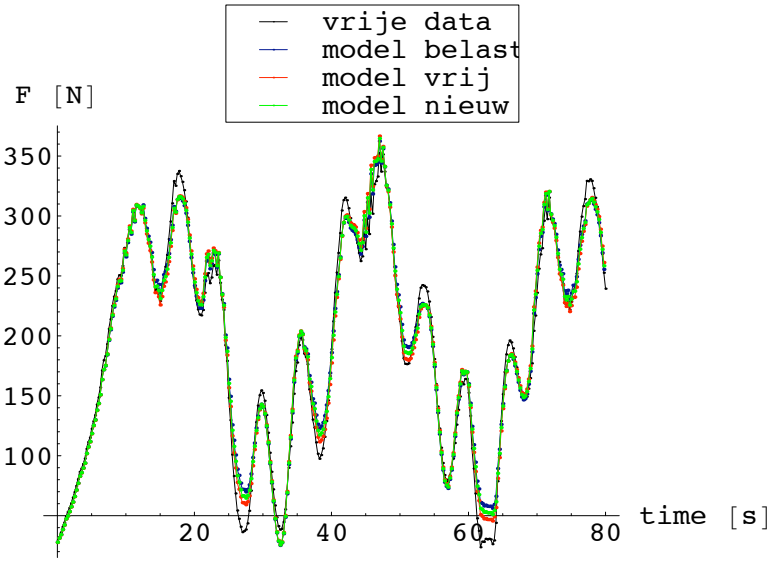
Figuur D.1: Validatie van model gevonden via belaste data op vrije data voor spier 1 samen met het gemiddelde model



Figuur D.2: Validatie van model gevonden via belaste data op vrije data voor spier 2 samen met het gemiddelde model



Figuur D.3: Validatie van model gevonden via belaste data op vrije data voor spier 3 samen met het gemiddelde model



Figuur D.4: Validatie van model gevonden via belaste data op vrije data voor spier 4 samen met het gemiddelde model

Bijlage E

Matrices dynamisch model

$$H = \begin{bmatrix} I_{z_1} + (d_{G_1}^2 + L_{G_1}^2)m_1 + I_{z_2} + (L_1^2 + d_{G_2}^2 + L_{G_2}^2)m_2 + 2L_1 m_2 (L_{G_2}\cos q_2 + d_{G_2}\sin q_2) & I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2 + L_1 L_{G_2} m_2 \cos q_2 + d_{G_2} L_1 m_2 \sin q_2 \\ I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2 + L_1 L_{G_2} m_2 \cos q_2 + d_{G_2} L_1 m_2 \sin q_2 & I_{z_2} + (d_{G_2}^2 + L_{G_2}^2)m_2 \end{bmatrix} \quad \text{(E.1)}$$

Voor de coriolis- en centrifugaalmatrix C :

$$C = \begin{bmatrix} L_1 m_2 \dot{q}_2 (d_{G_2}\cos q_2 - L_{G_2}\sin q_2) & L_1 m_2 \dot{q}_1 (d_{G_2}\cos q_2 - L_{G_2}\sin q_2) + L_1 m_2 \dot{q}_2 (d_{G_2}\cos q_2 - L_{G_2}\sin q_2) \\ L_1 m_2 \dot{q}_1 (-d_{G_2}\cos q_2 + L_{G_2}\sin q_2) & 0 \end{bmatrix} \quad \text{(E.2)}$$

Voor de gravitatiematrix G :

$$G = \begin{bmatrix} g\Big((L_{G_1} m_1 + L_1 m_2)\cos q_1 + L_{G_2} m_2 \cos(q_1 + q_2) + d_{G_1} m_1 \sin q_1 + d_{G_2} m_2 \sin(q_1 + q_2)\Big) \\ g\, m_2 \Big(L_{G_2}\cos(q_1 + q_2) + d_{G_2}\sin(q_1 + q_2)\Big) \end{bmatrix} \quad \text{(E.3)}$$

$$K = \begin{bmatrix} \ddot{q}_1 & \ddot{q}_1 + \ddot{q}_2 & g L_1 \cos q_1 + L_1^2 \ddot{q}_1 & g\cos(q_1 + q_2) + 2L_1 \cos q_2 \,|\, : \ddot{q}_1 + L_1 \cos q_2 \,|\, : \ddot{q}_2 - 2L_1 \dot{q}_1 \dot{q}_2 \sin q_2 - L_1 \dot{q}_2^{\,2} \sin q_2 & 2L_1 \cos q_2\, \dot{q}_1 \dot{q}_2 + L_1 \cos q_2\, \dot{q}_2^{\,2} + 2L_1 \dot{q}_1 \sin q_2 + L_1 \ddot{q}_2 \sin q_2 + g\sin(q_1 + q_2) & g\cos q_1 & g\sin q_1 \\ 0 & \ddot{q}_1 + \ddot{q}_2 & 0 & g\cos(q_1 + q_2) + L_1 \cos q_2\, \ddot{q}_1 + L_1 \dot{q}_1^{\,2} \sin q_2 & -L_1 \cos q_2\, \dot{q}_1^{\,2} + L_1 \ddot{q}_1 \sin q_2 + g\sin(q_1 + q_2) & 0 & 0 \end{bmatrix} \quad \text{(E.4)}$$

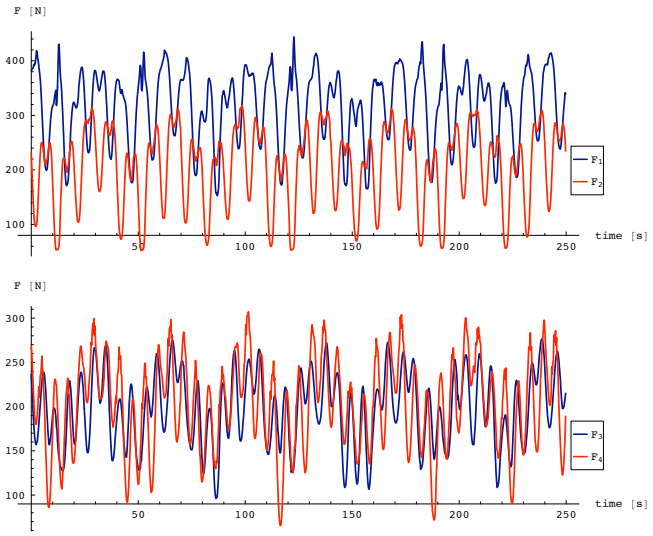
Bijlage F

Datareeksen bij Deel III

F.1 Datareeks 1

F.1.1 0 tot 250 s

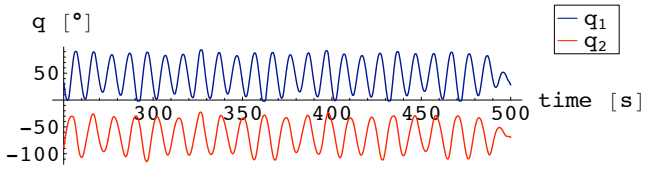
De krachten die door de krachtsensoren gemeten worden zijn voor deze reeks gegeven in Figuur F.1.



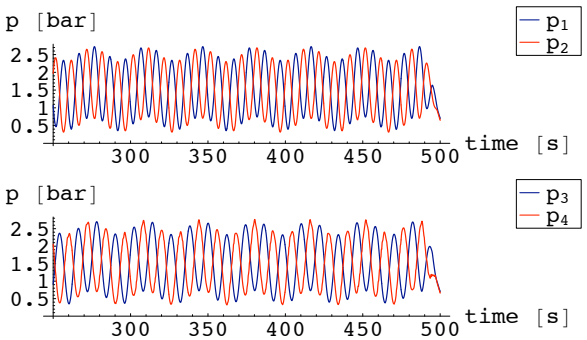
Figuur F.1: Krachtmetingen voor de eerste 250 s van datareeks 1

F.1.2 250 tot 500 s

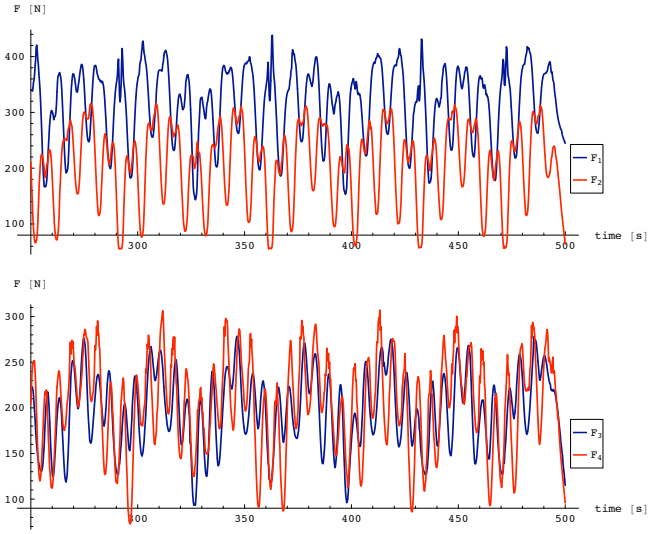
De figuren die seconde 250 tot 500 tonen voor datareeks 1 zijn hier getoond. De hoeken worden getoond in Figuur F.2. De drukken worden getoond in Figuur F.3. De krachten worden gegeven in Figuur F.4.



Figuur F.2: q_1 en q_2 voor de laatste 250 s van datareeks 1



Figuur F.3: Gestuurde drukken voor de laatste 250 s van datareeks 1

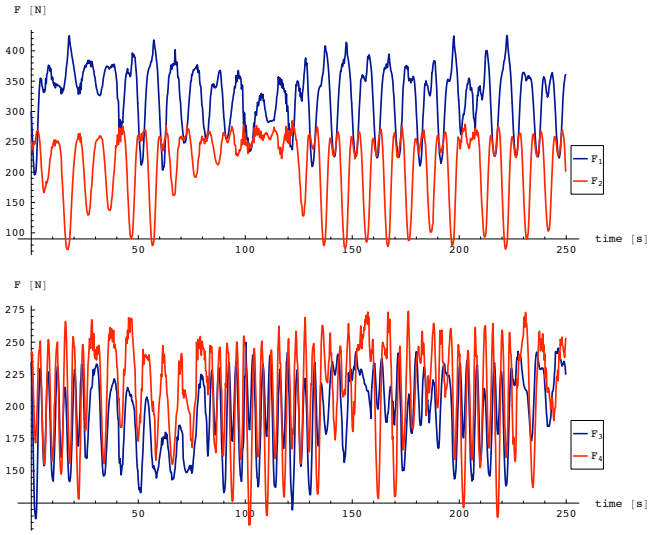


Figuur F.4: Krachtmetingen voor de laatste 250 s van datareeks 1

F.2 Datareeks 2

F.2.1 0 tot 250 s

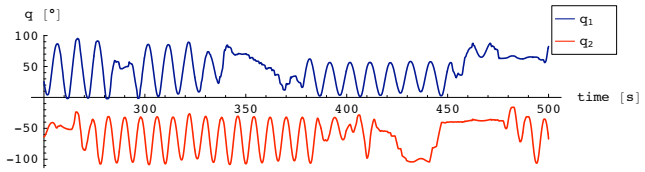
De krachten die door de krachtsensoren gemeten worden zijn voor deze reeks gegeven in Figuur F.5.



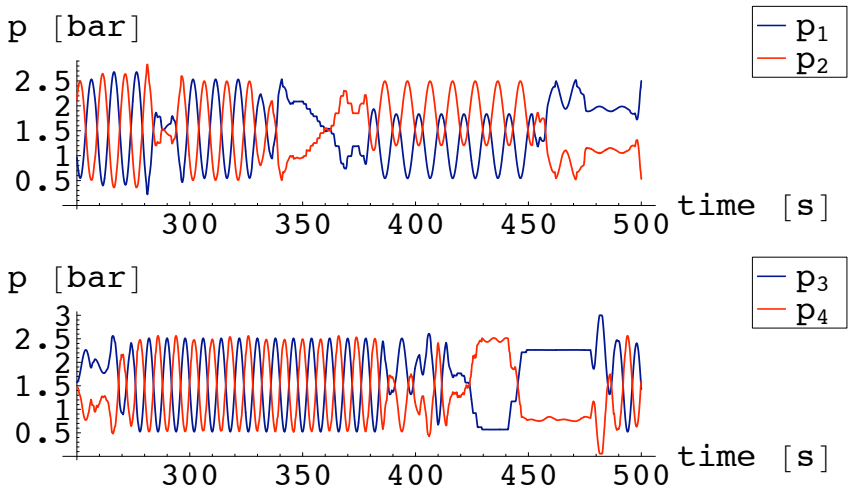
Figuur F.5: Krachtmetingen voor de eerste 250 s van datareeks 2

F.2.2 250 tot 500 s

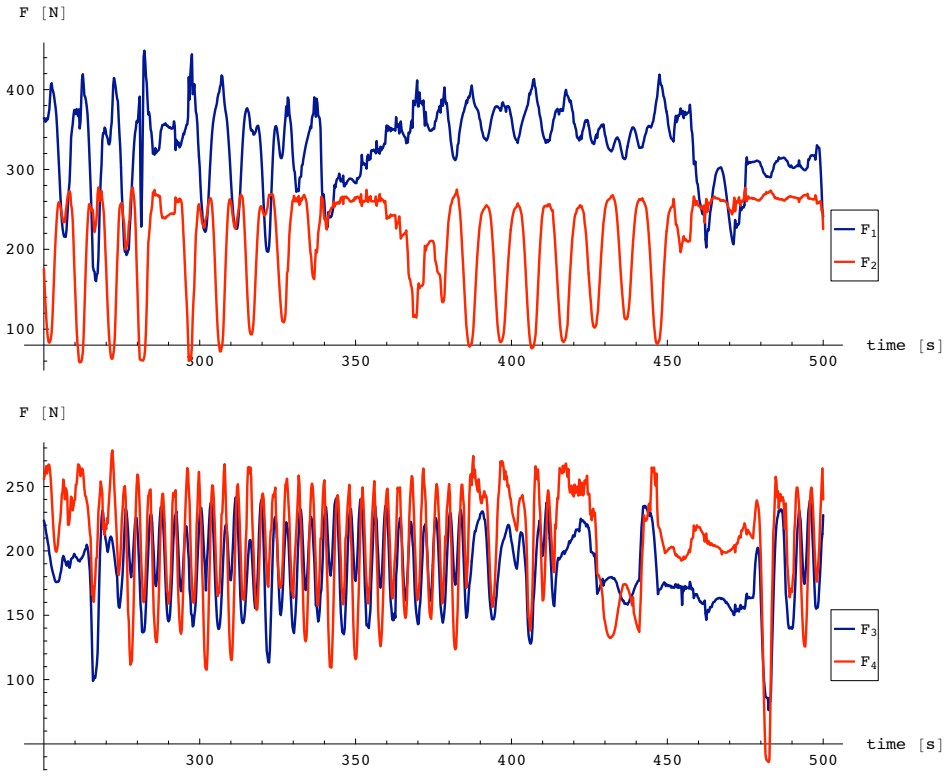
De figuren die seconde 250 tot 500 tonen voor datareeks 2 zijn hier getoond. De hoeken worden getoond in Figuur F.6. De drukken worden getoond in Figuur F.7. De krachten worden gegeven in Figuur F.8.



Figuur F.6: q_1 en q_2 voor de laatste 250 s van datareeks 2



Figuur F.7: Gestuurde drukken voor de laatste 250 s van datareeks 2



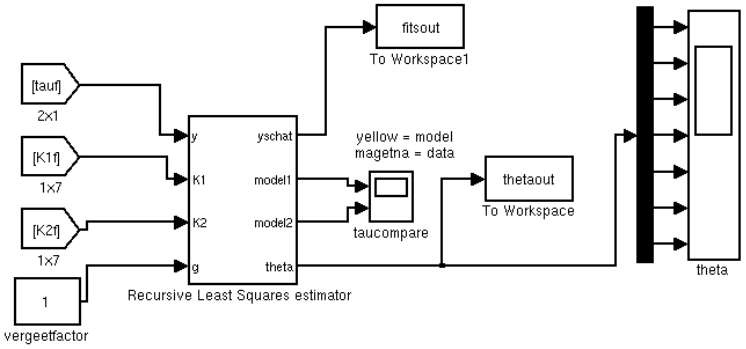
Figuur F.8: Krachtmetingen voor de laatste 250 s van datareeks 2

Bijlage G

Simulink schema's online schatter

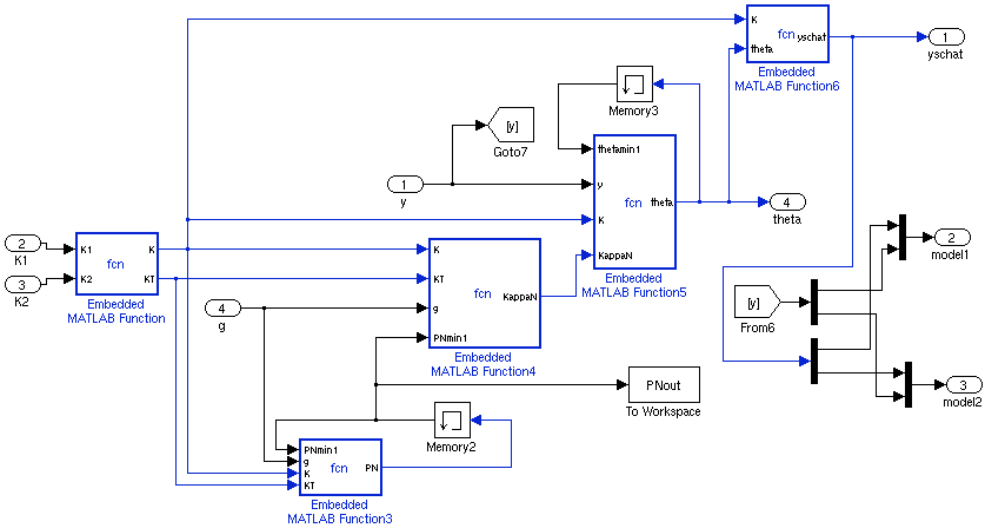
In Hoofdstuk 11 is een online identificatie methode beschreven die geïmplementeerd is in Simulink. De Simulink blokdiagramma's die bij deze schatter horen worden in deze bijlage gegeven.

Figuur G.1 toont hoe de schatter moet gebruikt worden in een simulatieschema. In deze figuur staat "Kf1" voor de eerste rij van de gefilterde observatiematrix $K_f$en "Kf2" voor de tweede rij. "tauf" staat voor de gefilterde vector van koppels. Het geschatte model wordt gegeven door "yschat". "model1" en "model2" tonen yschat en y in dezelfde grafiek. "theta" is een 7x1 vector die de geschatte parameters bevat.



Figuur G.1: De recursieve kleinste kwadraten schatter

Figuur G.2 toont het blok uit Figuur G.1 in detail. De symbolen die hierin gebruikt worden spreken voor zich wanneer gekeken word naar de formulaties van de schatter in paragraaf 11.2.



Figuur G.2: De recursieve kleinste kwadraten schatter in detail

Bibliografie

[1] B. Tondu A. Sanchez, V. Mahout. Nonlinear parametric identification of a mc-kibben artificial pneumatic muscle using flatness property of the system. *IEEE*, pages 70–74, 1998.

[2] M. Claes. Studie van de sturing van een pneumatisch aangedreven soepele manipulatorarm in direct contact met een operator. Master's thesis, Vrije Universiteit Brussel, 1999.

[3] F. Daerden. *Conception and realization of pleated pneumatic artificial muscles and their use as compliant actuation elements.* PhD thesis, Vrije Universiteit Brussel, 1999.

[4] Weiping Li J.J.E. Slotine. *Applied nonlinear control.* Prentice Hall, 1991.

[5] J.G. Fontaine N.K. M'Sirdi, M. Guihard. Identification and control of pneumatic driven robot.

[6] D. Lefeber P. Kool. *Robotics and Multi-Body Mechanics I, II.* Vrije Universiteit Brussel, 2005.

[7] Xavier Bombois P.M.J. Van den Hof. *System Identification For Control.* Delft University of Technology, 2003.

[8] J. Schoukens. *An Introduction To System Identification.* Vrije Universiteit Brussel, 2001.

[9] D. De Tobel. Bouw en sturing van een pneumatisch aangedreven manipulatorarm in direct contact met een operator. Master's thesis, Vrije Universiteit Brussel, 2003.

[10] B. Verrelst. *A dynamic walking biped actuated by pleated pneumatic artificial muscles: Basic concepts and control issues.* PhD thesis, Vrije Universiteit Brussel, 2005.

[11] `http://en.wikipedia.org/wiki/QR_decomposition`.

[12] `http://en.wikipedia.org/wiki/Cholesky_decomposition`.

[13] `http://mathworld.wolfram.com/LUDecomposition.html`.

[14] `http://www.uwlax.edu/faculty/will/svd/`.