FAM en de kunst van het regels genereren

Jack van Dijk

26 augustus 1997

FAM en de kunst van het regels genereren

Doctoraal Scriptie
Bestuurlijke Informatica
Erasmus Universiteit Rotterdam
Jack van Dijk
Begeleider: dr. ir. J. van den Berg

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

Voorwoord

Na een periode van anderhalf jaar ligt hier dan mijn afstudeerscriptie voor de studie Bestuurlijke Informatica aan de Erasmus Universiteit Rotterdam. Anderhalf jaar lang met fuzzy logic bezig zijn. Anderhalf jaar lang je familie uitleggen waarom je nog niet afgestudeerd bent. Anderhalf jaar lang je werkgever vertellen dat die bijna af is.

In december 1995 hoorde ik van een studiegenoot dat Jan van den Berg iets met fuzzy logic had voor een afstudeerder. Zelf had ik net het keuzevak Fuzzy Wiskunde gevolgd en was wel geïnterreseerd om eens verder met dit onderwerp aan de slag te gaan. Een eerste gesprek met Jan had tot resultaat dat we allebei het wel interresant vonden om met fuzzy logic aan te slag te gaan. Daarmee stopte de omschrijving dan ook, want waar we concreet naar zouden gaan kijken wisten we eigenlijk nog niet.

De "vonst" van het artikel van Wang en Mendel gaf ons een idee voor een scriptie onderwerp. Dit heeft er nu onder andere toe geleid dat we een artikel geschreven hebben, die ook geaccepteerd is, en dat ik mijn scriptie heb kunnen schrijven.

Terugkijkend op de afgelopen anderhalf jaar zijn er zeker een aantal mensen die ik hier even wil bedanken. Ten eerste natuurlijk en boven alles Jan van den Berg. Ondanks dat hij het zelf heel druk had wist hij altijd wel tijd vrij te maken om de vorderingen te bekijken. Hier werd altijd ruim de tijd voor genomen en vaak gingen de gesprekken ook nog over heel andere onderwerpen.

Verder gaat dank uit naar Jilco en Lourens die er voor verantwoordelijk zijn dat het brouwsel wat ik voortbracht enigzins het Nederlands ging benaderen. Zoals het meestal gaat moesten deze controles nog even op het laaste moment gebeuren, wat dus met de gebruikelijke stress gepaard ging.

Natuurlijk kom ik er niet onderuit om de mensen van Ortec te bedanken voor het tolereren dat ik de datum van mijn afstuderen voor mij uit bleef schuiven. Na de aanvankelijke beloofde half jaar is het uiteindelijk anderhalf jaar geworden. En hier bovenop ga ik ze nog verlaten ook!

Mijn familie bedank ik voor het feit dat ze er niet al te veel een probleem van hebben gemaakt dat ik mij gedurende mijn studie, en het laatste jaar speciaal, niet altijd even sociaal heb opgesteld ten opzichte van het familiegebeuren. Tja, en nu krijgen we natuurlijk het probleem dan nadat ik zeven jaar lang heb moeten vertellen hoe mijn studie ook al weer heette, ik de volgende dertig jaar kan gaan uitleggen wat voor werk ik eigenlijk doe.

Als laatste gaat er een special dank uit naar Ome Piet die mij duidelijk heeft gestimuleerd om te gaan studeren. Meer precies heeft hij me zelfs gestimuleerd om Informatica te gaan studeren in plaats van Scheepsbouwkunde wat mijn aanvankelijke keus was. Achteraf gezien een juiste keuze!

Jack van Dijk
Rotterdam
25 Augustus 1997.

Hoofdstuk 1

Inleiding

> Fuzzy theory is wrong, wrong, and pernicious. What we need is more logical thinking, not less. The danger of fuzzy logic is that it will encourage the sort of imprecise thinking that has brought us so much trouble. Fuzzy logic is the cocaine of science.

Professor William Kahan
University of California at Berkeley

Dit hoofdstuk is gebaseerd op gegevens uit Fuzzy Thinking van Bart Kosko [Kosko, 1993].

1.1 Het verleden

Fuzzy logic[1], oftewel 'vage logica' is niet echt een term die doet vermoeden dat we hier met serieuze wetenschap te maken hebben. Het tegendeel blijkt echter waar te zijn. Als we de materie beter bekijken dan blijken we met een wiskundig gefundeerde wetenschap te maken te hebben. De vaagheid slaat niet zozeer op de wetenschap zelf als wel op de wereld die het probeert te beschrijven. Fuzzy logic breekt met het begrip uit de traditionele logica dat een bepaalde uitspraak volledig waar is of volledig on-waar. Fuzzy logic geeft de mogelijkheid dat iets tot een bepaalde gradatie waar is.

Fuzzy logic is een van de vele gevolgen van de acceptatie in de wetenschap dat het begrip onzekerheid een wezenlijk deel van de wetenschap uitmaakt [Klir, 1995]. 'Traditioneel' wilde men zich in de wiskunde alleen bezig houden met dingen die nauwkeurig gedefiniëerd kunnen worden. Dientengevolge werden verschijnselen als 'niet exact', 'niet specifiek', 'vaag' en 'inconsistent' als niet-wetenschappelijk en ongewenst afgedaan. Men is achter gekomen dat deze

[1] In deze scriptie wordt voor verschillende termen uit de fuzzy logica en de informatica de engelse variant gebruikt. Dit gebeurt enerzijds omdat deze in het dagelijks gebruik ook zo worden gebruikt (processor vs. centrale verwerkingseenheid), anderzijds om 'kromme' vertalingen te voorkomen. Daar waar zowel de Engelse als de Nederlandse term veel worden gebruikt binnen het vakgebied zullen ze door elkaar gebruikt kunnen worden (set vs. verzameling).

dingen niet noodzakelijk tegen je hoeven te werken maar ook een belangrijk gereedschap kunnen vormen voor het modelleren van de werkelijkheid. Daar waar de wiskunden aanvankelijk op de 2-waardige logica was gebaseerd, is nu gebleken dat we deze logica kunnen generaliseren tot wat we nu fuzzy logic noemen.

In het algemeen wordt een artikel van Lotfi A. Zadeh beschouwd als de eerste aanzet tot wat we nu fuzzy logic noemen. Sommige van de ideeën waren echter al 30 jaar daarvoor door Max Black geopperd. Het besef dat er meer is dan een simpele waar, niet-waar wereld bestaat echter al veel langer. In 500 voor Christus werd er al getwijfeld aan de juistheid van de klassieke logica die alleen maar 0/1 oplossingen gaf. Het was Boeddha, die in India leefde, die de traditionele waar/niet-waar wereld doorbrak en ideeën introduceerde dat dingen beiden konden zijn. Dit besef van 'iets grijs', fuzzy-achtigs kom je nog steeds tegen in allerlei oosterse culturen, van Lao-tse's Taoisme tot de moderne Zen in Japan.

Maar Zadeh komt de eer toe van het introduceren van fuzzy denken in de moderne wetenschap. Zadeh introduceerde een theorie met verzamelingen waarvan de grenzen niet exact zijn. Het lidmaatschap van zo'n verzameling is dan geen zaak meer van ja of nee, maar veeleer een zaak van voor welk percentage iets lid is van de verzameling. Maar ook Zadeh ondervond de tegenstand die velen voor hem al hadden ondervonden. Sommigen vonden het fuzzy denken nog steeds onwetenschappelijk. Anderen beweerden dat fuzzy logic niets meer was dan kansberekening in een ander jasje. De laatste 'beschuldiging' lijkt op het eerste gezicht inderdaad aannemelijk. We zullen daarom met een voorbeeld het (subtiele) verschil tussen modelleren met de kansberekening en fuzzy logic proberen uit te leggen.

We beginnen het vergelijk met een vraag. Kun je een cirkel tekenen? Het antwoord hierop is nee. Niemand heeft ooit een cirkel kunnen tekenen. We zijn nooit verder gekomen dan een benadering van dat wat in de wiskunden onder een cirkel wordt verstaan. Als je maar nauwkeurig genoeg naar een cirkel gaat kijken dan zal je altijd nog onvolkomenheden blijven zien. Stel, we proberen een cirkel op een stuk papier te tekenen. Vervolgens worden daarover de volgende uitspraken gedaan. Kansberekening: de getekende figuur is waarschijnlijk een cirkel. Fuzzy: De figuur is een fuzzy cirkel. Welke is nu nauwkeuriger? De kansberekening legt zich vast op een kenmerk die moeilijk te vinden is in de figuur. Waar is namelijk het kansaspect? De figuur is statisch en ligt vast. We kunnen meten in hoeverre de figuur afwijkt van de kenmerken van een cirkel, maar dit geeft nog steeds geen uitspraak over de kans. Hoe meer informatie we hebben over een feit, hoe minder we het feit aan de waarschijnlijkheid of het geluk zullen wijten. Complete informatie laat weinig ruimte over voor waarschijnlijkheid. Feitelijk hadden we al vastgesteld dat niemand een cirkel kan tekenen. De kans dat het getekende figuur een cirkel is, is dus al per definitie gelijk aan nul. Het figuur zal echter wel associaties met een cirkel oproepen en kan dus wel als een fuzzy cirkel gedefiniëerd worden. Tot op zekere hoogte voldoet die wel aan de kenmerken van een cirkel, het zal echter nooit een 'echte' cirkel worden, voorzover we die al kunnen definiëren. Het is dus een fuzzy cirkel, een cirkel tot een bepaalde graad.

Betekent dit dan dat de kansberekening geen recht van bestaan meer heeft? Zeer beslist niet! Niemand zal het in zijn hoofd halen om te zeggen wanneer hij met een dobbelsteen gooit: "Ik ga een fuzzy zes gooien". Om vervolgens als hij vier heeft gegooid te zeggen: "Ik heb een fuzzy zes, met de graad twee derde gegooid!". Het gaat erom, om het juiste gereedschap voor het juiste probleem te gebruiken.

1.2 Het heden

Fuzzy logica wordt al veelvuldig gebruikt. Vooral Japan loopt hierin voorop. In dit land zijn er al duizenden patenten verleend op toepassingen van fuzzy logic. Van de patenten op fuzzy logic gebied uitgegeven in Amerika, is ongeveer 75% in Japanse handen. De volgende tabel toont enkele voorbeelden van hoe fuzzy logic wordt toegepast.

Produkt	*Bedrijf*	*Functie van Fuzzy logic*
Anti-blokkeer remmen	Nissan	Bediening van de remmen gebaseerd op autosnelheid, versnelling, wielsnelheid en acceleratie.
Chemische mixer	Fuji Electric	Mixt chemicaliën op basis van fabriek omstandigheden.
Vaatwasser	Matsushita	Stelt de tijd, spoel en was instellingen in op basis van de hoeveelheid vaat en de hoeveelheid vuil op de vaat.
Wasdroger	Matsushita	Bepaalt de droogtijd en het programma op basis van de hoeveelheid was en het soort materiaal van de kleding.
Televisie	Goldstar Sony Hitachi	Stelt de schermkleuren en contrast in voor ieder beeld en volume, gebaseerd op de omgeving.

Tabel 1.1: Commerciële produkten op basis van fuzzy logic

Fuzzy logic is intussen een volwaardige wetenschap geworden. Enkele jaren geleden kon het nog gebeuren dat een artikel voor een AI-congres werd geweigerd omdat de term fuzzy in de titel voorkwam (Na wijziging van de titel werd het artikel alsnog geaccepteerd!)[Kosko, 1992b]. Tegenwoordig is fuzzy logic een hot topic geworden. Dit blijkt onder andere uit het feit dat er tijdschriften en (wetenschappelijke) boeken verschijnen met dit onderwerp en dat er conferenties rondom dit onderwerp worden georganiseerd.

1.3 Doel

In deze scriptie richten we ons op het bouwen van een Fuzzy Associative Memory. Een FAM is een verzameling fuzzy regels die een fuzzy expert-systeem in staat stelt zijn beslissingen te nemen. Omdat de fuzzy regels het volledige referentiekader vormen voor het fuzzy expert-systeem is het van groot belang dat deze regels goed zijn of, in ieder geval, zo goed mogelijk zijn. Vanwege de nadruk op het genereren van regels in deze scriptie en de overeenkomsten van de gedachten uit de fuzzy logic met het Zen-denken kwam ik op de titel “FAM en de kunst van het regels genereren”. Hierbij heb ik mij voor deze titel laten inspireren door het

boek van Robert Pirsig, "ZEN and the art of motorcycle maintenance" [Pirsig, 1974].

Expert-systemen zijn systemen die gegeven een bepaalde input een bijbehorende output genereren binnen een zeker afgebakend kennisgebied. Daarmee kunnen ze dus een vervanging, of aanvulling, voor een expert op dat kennisgebied vormen. De kennis van het systeem moet op een of andere manier in het systeem worden vastgelegd. Dit zou bijvoorbeeld een bestand met historische data kunnen zijn.

Bij expert-systemen waar gebruik wordt gemaakt van fuzzy logica wordt de kennis opgeslagen in een FAM. De FAM bestaat uit fuzzy regels die verbanden tussen input en output beschrijven.

Als basis voor het bouwen van de regels maken we gebruik van het algoritme van Wang-Mendel voor het genereren van FAM-regels [Wang, 1991]. Dit is een algoritme om uit numerieke data een FAM te genereren. Dit algoritme blijkt nog een bepaalde input van een expert nodig te hebben. Deze expert moet kennis hebben over het te automatiseren kennisgebied en van fuzzy systems. Deze combinatie kan moeilijk te vinden zijn.

De beslissingen van de expert bestaan uit het kiezen van een aantal gebieden waarin het domein van een variabele wordt verdeeld. In deze scriptie wordt een methode ontwikkeld die het mogelijk maakt om deze keuze automatisch te bepalen. Een direct gevolg hiervan is dat het algoritme gemakkelijker praktisch toepasbaar wordt. Een persoon die van het algoritme gebruik wil maken hoeft nu niet direct meer een deskundige op het gebied van fuzzy logic te zijn en kan het algoritme als een tool gebruiken.

Samenvattend kunnen we het doel van deze scriptie omschrijven als het praktisch toepasbaar maken van het Wang-Mendel algoritme. Hierbij heeft speciaal de aandacht dat we over het algemeen last hebben van ruis in data. De methode die we zullen voorstellen zal hier goed mee om moeten kunnen gaan.

1.4 Methodologie

Gegeven ons bovenstaand omschreven doel zijn de experimenten die we zullen doen dan ook gericht om te komen tot een versie van het Wang-Mendel algoritme die eenvoudig in de praktijk toepasbaar is zonder dat men direkt zeer gespecialiseerde experts nodig heeft.

Een van de problemen bij het algoritme is: hoe het omgaat met ruis. Het bleek al gauw dat hier weinig over bekend was. Op een gegeven moment onstond het idee om 'cross validation'[2] te gebruiken bij het genereren van de regels. We wilden deze binnen de neurale netwerken veel gebruikte methode bij het Wang-Mendel algoritme gebruiken om zo een ruisbestendige methode voor het genereren van een FAM te krijgen.

De hypothese is dat het toepassen van cross validation bij het genereren van regels voor een Fuzzy Associative Memory een methode zal opleveren die aan de ene kant eenvoudig toe te passen is en aan de andere kant kan omgaan met ruis in data. Na toetsing van de methode met

[2] Voor deze en de andere terminologie in deze paragraaf geldt dat ze later in de scriptie zullen worden verduidelijkt.

'cross-validation' is het Extended Wang-Mendel algoritme ontstaan. Vervolgens hebben het EWM-algoritme getoetst aan een bekend controle probleem, namelijk dat van de omgekeerde slinger.

De bevindingen betreffende deze hypothese en de bijbehorende resultaten hebben reeds geleid tot een artikel [Berg, 1997] welke geaccepteerd is voor publicatie bij de ICONIP '97.

1.5 Opbouw scriptie

De scriptie heeft de volgende indeling.

In hoofdstuk 2 gaan we in op de vereiste basiskennis ten aanzien van fuzzy logic. Hierbij komt onder andere aan bod hoe de verschuiving van de klassieke logica naar de fuzzy logica heeft plaatsgevonden. In aansluiting daarop volgt een bespreking van de begrippen 'linguïstische variabelen' en 'fuzzy propositie'. Als laatste gaan we in op 'approximate reasoning'.

Hoofdstuk 3 gaat in op Expert-Systemen. Besproken wordt de algemene werking van een expert-systeem en de implementatie hiervan met behulp van neurale netwerken of fuzzy logic. Daarna volgt een bespreking van de voor- en nadelen van het gebruik van fuzzy logic voor het bouwen van expert-systemen.

Het hoofdstuk 4 wordt volledig gebruikt om het Wang-Mendel algoritme uit te leggen. De werking van het algoritme wordt toegelicht door aan de hand van een voorbeeld het algoritme te doorlopen. Met behulp van een klein experiment laten we vervolgens zien hoe bepaalde keuzes die je maakt je uitkomsten beïnvloeden.

In hoofdstuk 5 bespreken we een voorstel voor uitbreiding van het Wang-Mendel-algoritme. De keuze voor deze uitbreiding wordt duidelijk gemaakt aan de hand van een experiment. Hieruit volgt een aangepast Wang-Mendel-algoritme, het Extended Wang-Mendel-algoritme. Deze wordt vervolgens getoetst met behulp van een bekend controle probleem uit de fuzzy logic; de omgekeerde slinger.

In het laatste hoofdstuk zetten we alles nog eens op een rij en bespreken we de behaalde resultaten. Tevens geven we enkele suggesties voor verder onderzoek.

Hoofdstuk 2

Basisbegrippen

> So far as the laws of mathematics refer to reality, they are not certain. And so far as they are certain, they do not refer to reality.

Albert Einstein
Geometry and Experience

Tenzij anders vermeld is alle theorie uit dit hoofdstuk afkomstig uit [Klir, 1995]

2.1 Fuzzy Sets

Een van de vele veranderingen die de wetenschap en de wiskunde deze eeuw hebben doorgemaakt betreft de modellering van het begrip onzekerheid. De verandering bestaat uit een verschuiving in de opvattingen van een kijk waarin onzekerheid als onwenselijk wordt ervaren naar een waarin dit wordt geaccepteerd als iets waar we niet omheen kunnen. Naast diverse andere ontwikkelingen (b.v. in de waarschijnlijkheidsrekening en statistiek) zorgt de publicatie van een artikel van Lotfi A. Zadeh in 1965 [Zadeh, 1965] voor een belangrijke verschuiving in deze opvatting. Hoewel enkele ideeën in dit artikel al dertig jaar eerder door de filosoof Max Black werden gepubliceerd, wordt dit artikel toch gezien als het begin van fuzzy logica. Zadeh introduceerde in zijn artikel een theorie over fuzzy verzamelingen, verzamelingen waarvan de grenzen niet exact bepaald zijn. Het lidmaatschap van een dergelijke verzameling is geen kwestie van bevestiging of afkeuring maar kan allerlei waarden aannemen tussen deze twee uitersten.

2.1.1 De verschuiving van klassieke logica naar fuzzy logica

De significantie in het artikel van Zadeh was dat het niet alleen de waarschijnlijkheidsrekening, als enige methode voor het omgaan met onzekerheid aanviel, maar ook de fundering waarop deze is gebaseerd, namelijk de traditionele verzamelingsleer, waarbij óf $x \in A$ óf $x \notin A$. Als A een fuzzy verzameling is en x is een mogelijk lid van deze verzameling dan is de opmerking "x

is lid van A" niet noodzakelijk waar of onwaar, zoals in de twee-voudige logica. In fuzzy logica kan deze opmerking tot een bepaalde graad waar zijn. Het is in het algemeen gebruikelijk om deze mate van lidmaatschap uit te drukken in getallen die in het interval $[0, 1]$ liggen. De uiterste grenzen van dit interval vallen samen met de mogelijkheden die we in de traditionele logica hebben.

De bovenstaande beschrijving kunnen we op de volgende, meer formele, manier weergeven:

Definitie 2.1 (Indicatorfunctie) *Gegeven een verzameling* X , *niet leeg, te noemen het* **universum**; *onder een* **indicatorfunctie** *(characteristic function)* χ *verstaan we een functie* $\chi : \mathrm{X} \to \{0, 1\}$ *met domein* $D_\chi = \mathrm{X}$.

Elke indicatorfunctie χ op X bepaalt een deelverzameling $A_{(\chi)}$ van X.
En omgekeerd: elke deelverzameling A van X bepaalt een indicatorfunctie $\chi_{(A)}$ op X.

$$A_{(\chi)} = \{x \in \mathrm{X} \mid \chi(x) = 1\} \tag{2.1}$$

$$\forall\ x \in \mathrm{X} : \chi_{(A)}(x) = \begin{cases} 1 & \text{als } x \in A \\ 0 & \text{als } x \notin A \end{cases} \tag{2.2}$$

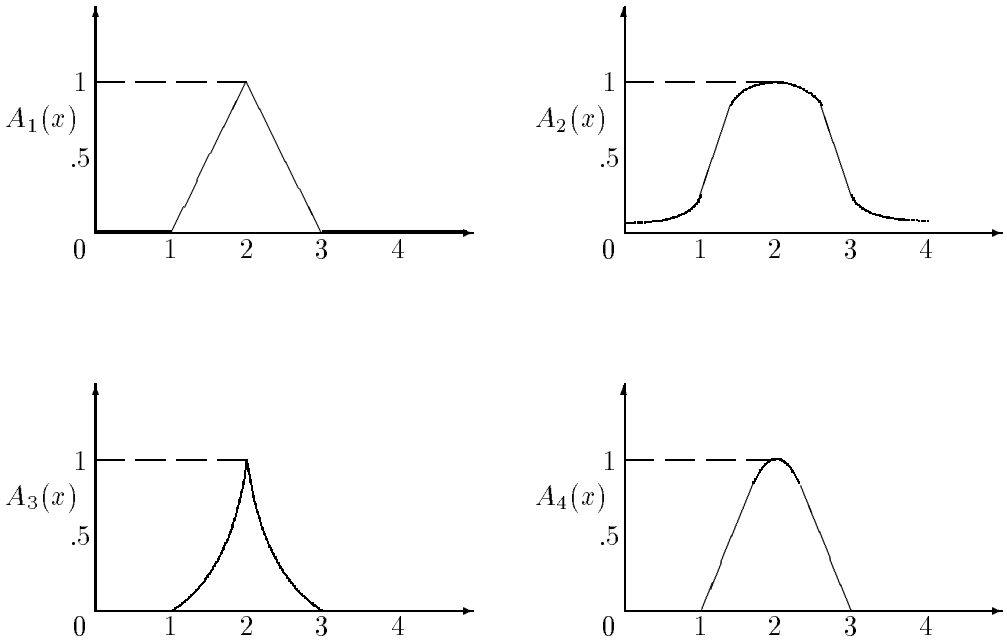
Er geldt: de indicatorfunctie $\chi_{(A_\chi)} = \chi$ en $A(\chi_A) = A$. Er bestaat dus een één-één-duidige bi-totale relatie tussen de collectie indicatorfuncties op X enerzijds en de collectie deelverzamelingen op X anderzijds.

Notatie-vereenvoudiging: geef de indicatorfunctie aan met dezelfde letter als de deelverzameling.

$$\forall\ x \in \mathrm{X} : A(x) = \chi_{(A)}(x) = \begin{cases} 1 & \text{voor } x \in A \\ 0 & \text{voor } x \notin A \end{cases} \tag{2.3}$$

Definitie 2.2 (Lidmaatschapsfunctie) *Gegeven een niet-lege verzameling* X, *het universum; onder een* **lidmaatschapsfunctie** *(memberfunction)* μ *op* X *verstaan we een functie* $\mu : \mathrm{X} \to [0, 1]$ *met domein* $D_\mu = \mathrm{X}$

Een fuzzy verzameling wordt exact bepaald door zijn lidmaatschapsfunctie. Als we nu een fuzzy verzameling van getallen 'in de buurt van 2' definiëren, dan zouden we in figuur 2.1 kunnen zien hoe we verschillende lidmaatschapsfuncties voor deze verzameling kunnen definiëren.



Figuur 2.1: Lidmaatschapsfuncties voor de fuzzy verzameling "Getallen dicht bij 2".

We zien dat de lidmaatschapsfuncties van vorm verschillen, maar toch een aantal gemeenschappelijke eigenschappen hebben, namelijk:

1. $A_i(2) = 1$ en $A_i(x) < 1$ voor alle $x \neq 2$;
2. A_i is symmetrisch rond $x = 2$, dus $A_i(2 + x) = A_i(2 - x)$ voor $x \in \mathbb{R}$;
3. $A_i(x)$ daalt monotoon van 1 naar 0 als $|2 - x|$ stijgt.

Het zal in het algemeen van de "omstandigheden" afhangen welke vorm het beste werkt voor je lidmaatschapsfunctie.

2.1.2 Operatoren

Voor fuzzy verzamelingen zijn dezelfde operatoren beschikbaar als voor klassieke verzamelingen, zoals doorsnede, vereniging en dergelijke. In tegenstelling tot klassieke verzamelingen zijn bij fuzzy verzamelingen de verschillende combinaties niet aftelbaar. Dit betekent dat er in het algemeen aan een fuzzy operator uitdrukking wordt gegeven met behulp van een functie, in tegenstelling tot een tabel met de verschillende mogelijkheden in de klassieke logica.

De standaard complement, $\bar{A}$, van een fuzzy verzameling A voor het universum X is voor alle

$x \in$ X als volgt gedefiniëerd:

$$\bar{A}(x) = 1 - A(x) \tag{2.4}$$

Elementen van X waarvoor geldt $A(x) = \bar{A}(x) = 0,5$ worden evenwichtspunten genoemd.

Gegeven twee fuzzy verzamelingen, A en B, worden hun doorsnede en hun vereniging standaard als volgt gedefiniëerd:

$$(A \cap B)(x) = \min[A(x), B(x)] \tag{2.5}$$
$$(A \cup B)(x) = \max[A(x), B(x)] \tag{2.6}$$

In de klassieke logica gelden voor operatoren met de standaard complement, de vereniging en de doorsnede de eigenschappen zoals vermeld in tabel 2.1. In te zien valt dat al deze operatoren, op de wet van contradictie en de wet van 'law of excluded middle' na, ook voor de fuzzy operatoren geldt.

Involution	$\overline{\overline{A}} = A$
Communativiteit	$A \cup B = B \cup A$
	$A \cap B = B \cap A$
Associativiteit	$(A \cup B) \cup C = A \cup (B \cup C)$
	$(A \cap B) \cap C = A \cap (B \cap C)$
Distributiviteit	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
Idempotence	$A \cup A = A$
	$A \cap A = A$
Absorption	$A \cup (A \cap B) = A$
	$A \cap (A \cup B) = A$
Absorption by X en $\emptyset$	$A \cup \text{X} = \text{X}$
	$A \cap \emptyset = \emptyset$
Identity	$A \cup \emptyset = A$
	$A \cap \text{X} = A$
Law of contradiction	$A \cap \overline{A} = \emptyset$
Law of excluded middle	$A \cup \overline{A} = \text{X}$
De Morgan's wetten	$\overline{A \cap B} = \overline{A} \cup \overline{B}$
	$\overline{A \cup B} = \overline{A} \cap \overline{B}$

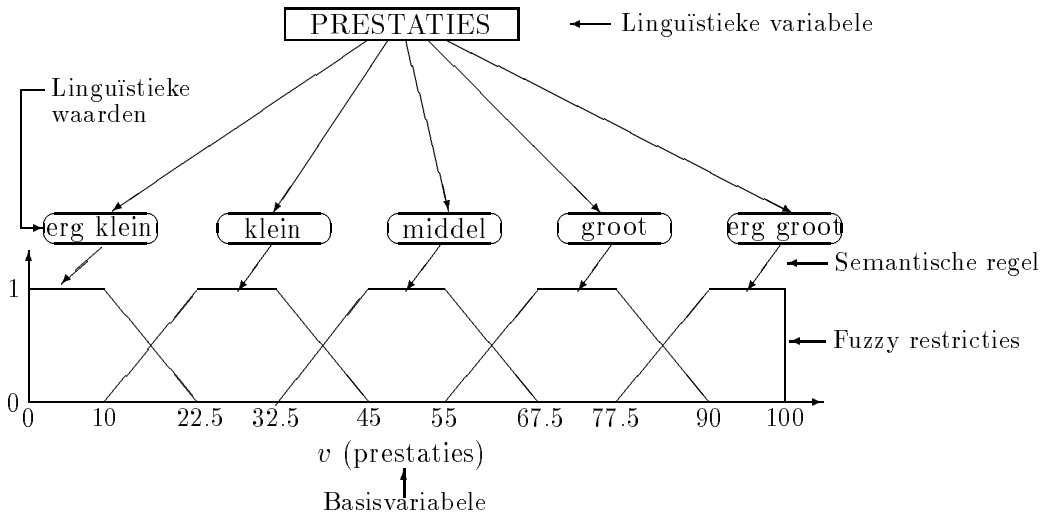
Tabel 2.1: Fundamentele eigenschappen van standaard operatoren van fuzzy sets

2.2 Linguïstische variabelen

Fuzzy getallen spelen een fundamentele rol in het formuleren van kwantitatieve fuzzy variabelen. Fuzzy getallen zijn variabelen waarvan de staat wordt beschreven door fuzzy nummers. Dit terwijl de fuzzy nummers linguïstische beschrijvingen geven zoals, *erg klein, klein, gemiddeld* enzovoorts. Als deze worden gebruikt in een bepaalde context dan worden de resulterende constructies liguïstieke variabelen genoemd.

Iedere linguïstische variabele is gedefiniëerd door een basisvariabele, waarvan de mogelijk waarden reële getallen zijn met een bepaald bereik. De basisvariabele is bijvoorbeeld de waarde van de temperatuur, snelheid e.d. In een linguïstische variabele drukken de linquïstische termen de verschillende waarden voor de basisvariabele uit. De linguïstische variabelen worden vervolgens in fuzzy getallen uitgedrukt.

Meer formeel wordt een linguïstische variabele gedefiniëerd als een set van vijf variabelen (v, T, X, g, m). Deze variabelen kunnen als volgt worden geïnterpreteerd. v is de naam van de variabele. T is de verzameling van linguïstische termen van v die slaan op een basisvariabele waarvan de waarden een bereik hebben binnen het universum X. g is de syntactische regel (grammatica) voor het genereren van linguïstische termen en m is een semantische regel die aan iedere linguïstische term $t \in T$ zijn betekenis, $m(t)$, verbindt. Deze betekenis is een fuzzy verzameling in X ($m : T \rightarrow \mathcal{F}$). In figuur 2.2 is een voorbeeld te zien van een linguïstische variabele.



Figuur 2.2: Een voorbeeld van een linguïstische variabele

2.3 Fuzzy proposities

Fuzzy logic is een generalisatie van de klassieke logica, dus hebben we ook fuzzy proposities. Terwijl in de klassieke logica een propositie waar of onwaar moet zijn, is in fuzzy logica de vraag of een propositie waar of onwaar een kwestie van graad.

Er zijn vier types fuzzy proposities te onderscheiden:

1. ongeconditioneerde en ongekwalificeerde proposities;
2. ongeconditioneerde en gekwalificeerde proposities;
3. geconditioneerde en ongekwalificeerde proposities;
4. geconditioneerde en gekwalificeerde proposities.

We zullen ieder nu apart beschrijven.

2.3.1 Ongeconditioneerde en ongekwalificeerde proposities

Deze vorm van fuzzy proposities, p, kan worden beschreven door:

$$p : \mathcal{V} \text{ is } F \tag{2.7}$$

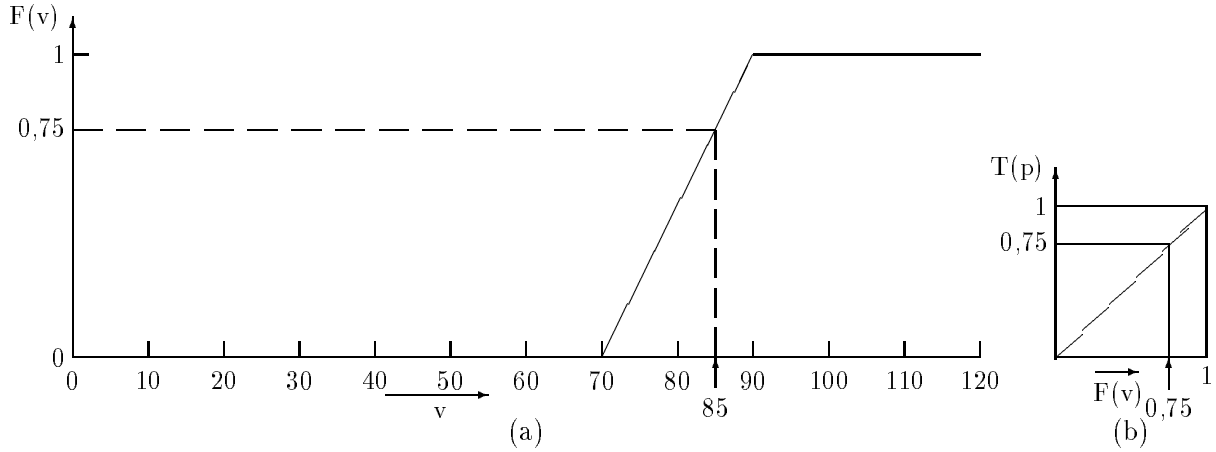
$\mathcal{V}$ is een variabele die waarden v uit universum V aanneemt. F is een fuzzy verzameling op V die een fuzzy predicaat voorstelt zoals groot, duur, laag, normaal, enzovoort. Gegeven een bepaalde waarde voor $\mathcal{V}$ (stel v), behoort deze waarde tot F met de lidmaatschapsgraad $F(v)$. Deze lidmaatschapsgraad wordt geïntepreteerd als de graad van waar zijn, $T(p)$, van propositie p. Dus:

$$T(p) = F(v) \tag{2.8}$$

voor iedere gegeven waarde v voor variabele V in propositie p.

Stel $\mathcal{V}$ is de temperatuur op een bepaalde plaats op aarde (gemeten in graden Fahrenheit) en laat de lidmaatschapsfunctie het predicaat *hoog* weergegeven, zie figuur 2.3a. Dan wordt de fuzzy propositie uitgedrukt door:

$$p : temperature(\mathcal{V}) \text{ is } hoog(F) \tag{2.9}$$



Figuur 2.3: Componenten van de fuzzy propositie p: Temperatuur ($\mathcal{V}$) is hoog (F)

De graad van waarheid, $T(p)$, hangt af van de feitelijke waarde van de temperatuur en van de gegeven definitie voor het predicaat *hoog*. De graad van waarheid wordt uitgedrukt door de lidmaatschapsfunctie T in figuur 2.3b, welke uitdrukking geeft aan formule 2.8. Dus als $v = 85$ dan geldt $F(85) = 0,75$ en $T(p) = 0,75$.

T vormt hier een brug tussen fuzzy verzamelingen en fuzzy proposities.

In bepaalde fuzzy proposities worden waarden voor variable $\mathcal{V}$ in (2.7) verbonden aan individuele elementen in een bepaalde verzameling I. Variabele $\mathcal{V}$ wordt dan een functie, $\mathcal{V} : I \rightarrow V$, waarbij $\mathcal{V}(i)$ de waarde is van $\mathcal{V}$ voor individu i in V. Formule 2.7 wordt dan:

$$p : \mathcal{V}(i) \text{ is } F, \text{ met } i \in I \tag{2.10}$$

2.3.2 Ongeconditioneerde en gekwalificeerde proposities

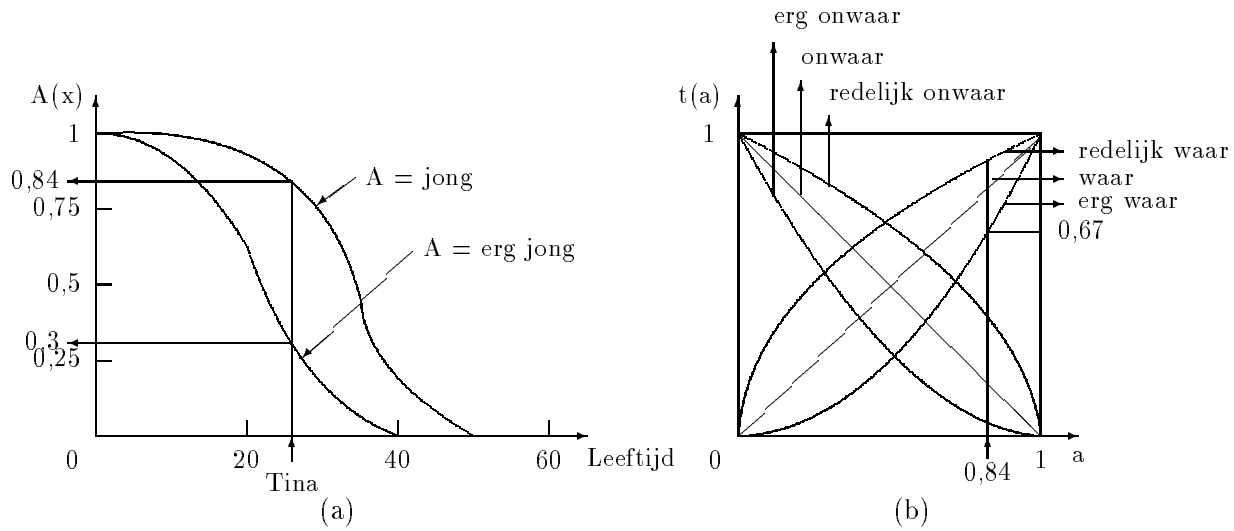
Proposities p van dit type worden gekarakteriseerd door één van de volgende vormen:

$$p : \mathcal{V} \text{ is } F \text{ is } S \tag{2.11}$$

óf

$$p : \text{Pro}\{\mathcal{V} \text{ is } F\} \text{ is } P \tag{2.12}$$

$\mathcal{V}$ en F hebben dezelfde betekenis als in formule 2.7. Pro$\{\mathcal{V}$ is $F\}$ is de kans op de fuzzy gebeurtenis "$\mathcal{V}$ is F", S is een fuzzy waarheidsindicator, en P is een fuzzy kans indicator. Ook hier kan $\mathcal{V}$ vervangen worden door $\mathcal{V}(i)$, met dezelfde betekenis als in formule 2.10. Formule 2.11 wordt wel waarheidgekwalificeerd genoemd en formule 2.12 kans gekwalificeerd. Zowel S als P worden gerepresenteerd door een fuzzy verzameling op $[0, 1]$.



Figuur 2.4: Waarheidswaarden van een fuzzy propositie

Een voorbeeld van een waarheid gekwalificeerde propositie is "T is jong is erg waar", waarbij het predicaat *jong* en de waarheidsindicator *erg waar* gerepresenteerd worden door de fuzzy verzamelingen zoals die te zien zijn in figuur 2.4. Als we aannemen dat Tina 26 jaar oud is, dan behoort ze tot de verzameling die het predicaat *jong* representeert met een lidmaatschapswaarde 0,84. Aldus hoort de propositie tot de verzameling proposities die erg waar zijn met de lidmaatschapswaarde 0,67. Dit heeft tot gevolg dat de graad van waarheid van de waarheid gekwalificeerde propositie ook 0,67 is.

In het algemeen wordt de graad van waarheid van iedere waarheid gekwalificeerde propositie p gegeven voor iedere $v \in V$ door de formule:

$$T(p) = S(F(v)) \tag{2.13}$$

Als we de lidmaatschapsfunctie $G(v) = S(F(v))$, voor $v \in V$, als een eenvoudig predicaat kunnen we iedere waarheid gekwalificeerde propositie met de vorm (2.11) intepreteren als de ongekwalificeerde propositie "$\mathcal{V}$ is G".

In te zien is dat ongekwalificeerde proposities speciale gevallen van waarheid gekwalificeerde proposities zijn, namelijk die waarin we aannemen dat de waarheid indicator S waar is. Zoals te zien is in de figuren 2.3b en 2.4b is de lidmaatschapsfunctie die deze indicator representeert de identiteitsfunctie. Dat wil zeggen $S(F(v)) = F(v)$ voor ongekwalificeerde proposities, dus mag S genegeerd worden om de zaak eenvoudiger te maken.

2.3.3 Geconditioneerde en ongekwalificeerde proposities

Geconditioneerde ongekwalificeerde proposities hebben de volgende vorm:

$$p: \text{ Als } \mathcal{X} \text{ is } A, \text{ dan } \mathcal{Y} \text{ is } B \tag{2.14}$$

Hierin zijn $\mathcal{X}, \mathcal{Y}$ variabelen waarvan de waarden respectievelijk in de verzamelingen X, Y liggen. A, B zijn fuzzy verzamelingen op respectievelijk X, Y. Deze proposities kunnen ook gezien worden als proposities van de vorm:

$$\langle \mathcal{X}, \mathcal{Y} \rangle \text{ is } R \tag{2.15}$$

R is hier een fuzzy verzameling op $X \times Y$ die wordt bepaald voor iedere $x \in X$ en $y \in Y$ door de formule:

$$R(x, y) = \mathcal{J}[A(x), B(y)] \tag{2.16}$$

waar $\mathcal{J}$ een binaire operator op $[0, 1]$ is waarbij deze een fuzzy implicatie representeert. De fuzzy implicaties zullen verder uitgelegd worden in 2.4.2.

2.3.4 Geconditioneerde en gekwalificeerde proposities

Dit type propositie kan worden uitgedrukt als:

$$p: \text{ Als } \mathcal{X} \text{ is } A, \text{ dan } \mathcal{Y} \text{ is } B \text{ is } S \tag{2.17}$$

óf

$$p: \text{Pro}\{\mathcal{X} \text{ is } A | \mathcal{Y} \text{ is } B\} \text{ is } P \tag{2.18}$$

$\text{Pro}\{\mathcal{X} \text{ is } A | \mathcal{Y} \text{ is } B\}$ is een geconditioneerde kans.

2.4 Approximate reasoning

'Approximate reasoning' is het redeneren op basis van fuzzy regels. Approximate reasoning stelt fuzzy systemen in staat hun werk te doen. Inhoudende op basis van een bepaalde input en een aantal regels gevolgtrekkingen doen en een bepaalde uitkomst geven.

2.4.1 Gevolgtrekkingen van geconditioneerde fuzzy proposities

In deze sectie zullen we de fuzzy generalisaties van drie klassieke afleidingsregels, *modus ponens*, *modus tollens* en *hypothetical syllogism* bespreken.

Beschouw variabelen $\mathcal{X}$ en $\mathcal{Y}$ die waarden aannemen uit respectievelijk de fuzzy verzamelingen X en Y. Neem aan dat voor alle $x \in X$ en $y \in Y$ de variabelen verbonden zijn door een functie $y = f(x)$. Dan, gegeven dat $\mathcal{X} = x$ kunnen we concluderen dat $\mathcal{Y} = f(x)$. Analoog kunnen we concluderen dat als we weten dat de waarde van $\mathcal{X}$ in een gegeven verzameling A ligt, dat de waarde van $\mathcal{Y}$ in de verzameling $B = \{y \in Y | y = f(x), x \in A\}$ ligt.

Neem nu aan dat de variabelen verbonden zijn door een willekeurige relatie op $X \times Y$, niet noodzakelijk een functie. Dan geldt dat gegeven $\mathcal{X} = u$ en een relatie R, we kunnen afleiden dat de waarde van $\mathcal{Y}$ in de verzameling $B = \{y \in Y | \langle x, y \rangle \in R\}$ ligt. Analoog, als we weten dat $\mathcal{X} \in A$, kunnen we afleiden dat $\mathcal{Y} \in B$, waarbij $B = \{y \in Y | \langle x, y \rangle \in R, x \in A\}$. In te zien is dat deze afleiding ook uitstekend in indicatorfuncties χ_A, χ_B, χ_R van de verzamelingen A, B, R is uit te drukken.

$$\chi_B(y) = \sup_{x \in X} \min[\chi_A(x), \chi_R(x, y)] \tag{2.19}$$

voor alle $y \in Y$.

Als we nog een stap verder gaan en aannemen dat R een fuzzy relatie is op $X \times Y$, en A', B' zijn fuzzy verzamelingen op X en Y. Dan, als R en A' gegeven zijn, kunnen we B' krijgen door de formule

$$B'(y) = \sup_{x \in X} \min[A'(x), R(x, y)] \tag{2.20}$$

voor alle $y \in Y$. Dit is een generalisatie van (2.19) door de indicatorfuncties te vervangen met de corresponderende lidmaatschapsfuncties. We kunnen deze formule ook in matrix vorm schrijven

$$\mathbf{B}' = \mathbf{A}' \circ \mathbf{R} \tag{2.21}$$

en wordt de *compositional rule of inference* genoemd.

Relatie R is verweven in een geconditioneerde fuzzy propositie p van de vorm

$$p: \text{ Als } \mathcal{X} \text{ is } A, \text{ dan } \mathcal{Y} \text{ is } B \tag{2.22}$$

en wordt bepaald voor alle $x \in X$ en alle $y \in Y$ door de formule

$$R(x, y) = \mathcal{J}[A(x), B(y)] \tag{2.23}$$

waarbij $\mathcal{J}$ een fuzzy implicatie is.

Als we de relatie R gebruiken van de gegeven propositie p in (2.23) en we een tweede propositie q hebben van de vorm

$$q : \mathcal{X} \text{ is } A' \tag{2.24}$$

dan mogen we concluderen dat $\mathcal{Y}$ is B' vanwege de compositie regel of inference (2.20). Deze procedure wordt de *gegeneraliseerde modus ponens* genoemd.

Als we propositie p als een regel zien en propositie q als een feit, dan wordt de gegeneraliseerde modus ponens uitgedrukt door het volgende schema:

Regel:	Als $\mathcal{X}$ is A, dan $\mathcal{Y}$ is B
Feit:	$\mathcal{X}$ is A'
Conclusie:	$\mathcal{Y}$ is B'

(2.25)

In dit schema wordt B' berekend met (2.20), en R in deze formule wordt bepaald door (2.23).

Een ander inference regel in de fuzzy logic, die *gegeneraliseerde modus tollens* heet, wordt uitgedrukt door het volgende schema:

Regel:	Als $\mathcal{X}$ is A, dan $\mathcal{Y}$ is B
Feit:	$\mathcal{X}$ is B'
Conclusie:	$\mathcal{X}$ is A'

(2.26)

In dit geval heeft de samengestelde regel van inference de vorm

$$A'(x) = \sup_{y \in Y} \min[B'(y), R(x, y)] \tag{2.27}$$

en R wordt in deze formule opnieuw bepaald door (2.23).

Als laatste zullen we nog de generalisatie van hypothetical syllogism bespreken. Deze is gebaseerd op twee geconditioneerde fuzzy proposities en wordt uitgedrukt door het volgende schema:

Regel 1:	Als $\mathcal{X}$ is A, dan $\mathcal{Y}$ is B
Regel 2:	Als $\mathcal{Y}$ is B, dan $\mathcal{Z}$ is C
Conclusie:	Als $\mathcal{X}$ is A, dan $\mathcal{Z}$ is C

(2.28)

In dit geval zijn $\mathcal{X}$, $\mathcal{Y}$ en $\mathcal{Z}$ variabelen die hun waarden aannemen uit respectievelijk de verzamelingen X, Y en Z. A, B en C zijn fuzzy verzamelingen op respectievelijk de verzamelingen X, Y en Z.

Voor iedere geconditioneerde fuzzy propositie in (2.28 is er een fuzzy relatie bepaald door (2.23). Deze relaties zijn bepaald voor iedere $x \in X$, $y \in Y$ en $z \in Z$ door de formules

$$R_1(x, y) \quad = \quad \mathcal{J}[A(x), B(y)] \tag{2.29}$$

$$R_2(y,z) = \mathcal{J}[B(y), C(z)] \tag{2.30}$$
$$R_3(x,z) = \mathcal{J}[A(x), C(z)] \tag{2.31}$$

Gegeven R_1, R_2 en R_3, verkregen door deze formules, zeggen we dat de gegeneraliseerde hypothetical syllogism geldt als

$$R_3(x,z) = \sup_{y \in Y} \min[R_1(x,y), R_2(y,z)] \tag{2.32}$$

die wederom de samengestelde inference regel uitdrukt. Deze formule kan ook weer geschreven worden in matrix vorm

$$\mathbf{R_3} = \mathbf{R_1} \circ \mathbf{R_2} \tag{2.33}$$

2.4.2 Fuzzy implicaties

Een fuzzy implicatie, $\mathcal{J}$, is een functie met de volgende vorm:

$$\mathcal{J} : [0,1] \times [0,1] \rightarrow [0,1] \tag{2.34}$$

welke voor iedere mogelijke waarheidwaarden a, b van gegeven fuzzy proposities respectievelijk p, q, de waarheidwaarde $\mathcal{J}(a,b)$ definiëert van de conditionele propositie "als p, dan q". Deze functie is bedoeld als een uitbreiding van de klassieke implicatie, $p \Rightarrow q$, van het domein $\{0,1\}$ naar het domein $[0,1]$ van waarheidswaarden in fuzzy logica.

In de klassieke logica, waar $a, b \in \{0,1\}$, kan $\mathcal{J}$ op verschillende manieren gedefinieerd worden. En hoewel al deze manieren in de klassieke logica equivalent zijn leveren ze in de fuzzy logica verschillende klassen van implicaties op. We zullen de meest voorkomende operatoren van fuzzy implicaties hier bespreken.

Een manier om $\mathcal{J}$ te definiëren in de klassieke logica is de logische formule:

$$\mathcal{J}(a,b) = \bar{a} \vee b \tag{2.35}$$

voor iedere $a, b \in \{0,1\}$. Als we deze formule voor het fuzzy geval moeten bekijken beschouwen we de complement en disjunctie als fuzzy complement en fuzzy vereniging. We krijgen dan de volgende uitdrukking voor $\mathcal{J}$ in fuzzy logica:

$$\mathcal{J}(a,b) = u(c(a), b) \tag{2.36}$$

voor alle $a, b \in [0,1]$, waarbij u en c staan voor de fuzzy vereniging (union) en fuzzy complement.

Een andere manier om $\mathcal{J}$ in de klassieke logica te definiëren is:

$$\mathcal{J}(a,b) = \max\{x \in \{0,1\} | a \wedge x \leq b\} \tag{2.37}$$

voor alle $a, b \in \{0,1\}$. Als we de conjunctie als een fuzzy doorsnede beschouwen, dan wordt $\mathcal{J}$ in de fuzzy logica gedefiniëerd als:

$$\mathcal{J}(a,b) = \sup\{x \in [0,1] | d(a,x) \leq b\} \tag{2.38}$$

voor alle $a, b \in [0,1]$, waarbij d staat voor de fuzzy doorsnede.

Terwijl de definities (2.35) en (2.37) voor implicaties in de klassieke logica equivalent zijn, zijn de fuzzy uitbreidingen (2.36) en (2.37) dat niet. Als we nog iets verder gaan, dan kunnen we (2.35) herschrijven als:

$$\mathcal{J}(a,b) = \bar{a} \vee (a \wedge b) \tag{2.39}$$

of

$$\mathcal{J}(a,b) = (\bar{a} \wedge \bar{b}) \vee b \tag{2.40}$$

De fuzzy uitbreidingen hiervan worden dan, respectievelijk:

$$\mathcal{J}(a,b) = u(c(a), d(a,b)), \tag{2.41}$$
$$\mathcal{J}(a,b) = u(d(c(a), c(b)), b) \tag{2.42}$$

waarin u, d, c aan de De Morgan's wetten moeten voldoen. Opnieuw geldt dat terwijl (2.35), (2.39) en (2.40) equivalent zijn in de klassieke logica, dat niet geldt voor hun fuzzy uitvoeringen (2.36), (2.41) en (2.42). De law of absorption of negation geldt namelijk in het algemeen niet voor fuzzy logica.

Uit (2.36) onstaat een familie van implicaties die de S-implicaties wordt genoemd. De volgende vier implicaties zijn de meest bekende voorbeelden uit de S-implicatie familie.

1. Als we de standaard fuzzy vereniging gebruiken, krijgen we de functie $\mathcal{J}_b$ gedefiniëerd voor alle $a, b \in [0,1]$ door de formule
   $$\mathcal{J}_b(a,b) = \max(1-a, b) \tag{2.43}$$
   Deze wordt de *Kleene-Dienes implicatie* genoemd.
2. Als we de algebraïsche som als fuzzy vereniging kiezen (d.w.z. $u(a,b) = a + b - ab$), krijgen we
   $$\mathcal{J}_r(a,b) = 1 - a + ab \tag{2.44}$$
   die de *Reichenbach implicatie* wordt genoemd.
3. De keuze voor de begrensde som, $u(a,b) = \min(1, a+b)$, resulteert in de functie
   $$\mathcal{J}_a(a,b) = min(1, 1-a+b) \tag{2.45}$$
   welke de *Lukasiewicz implicatie* wordt genoemd.

4. Als we de drastische fuzzy vereniging, u_{max}, kiezen, krijgen we

$$\mathcal{J}_{LS}(a,b) = \begin{cases} b & \text{als } a = 1 \\ 1-a & \text{als } b = 0 \\ 1 & elders \end{cases} \tag{2.46}$$

die de *grootste S-implicatie* is.

Op deze manier is er ook een familie implicaties uit (2.38) onstaan die de familie van *R-implicaties* vormt. De familie die onstaat uit (2.41) wordt de *QL-implicaties* genoemd.

Hoofdstuk 3

Expert systemen

> Proposition 1. The world is everything that is the case.
> Proposition 6.362. What can be described can happen too, and what is excluded by the law of causality cannot be described.
> Proposition 7. Whereof one cannot speak, one thereof must be silent.

Ludwig Wittgenstein
Tractatus Logico-Philosophicus

In dit hoofdstuk wordt een beschrijving gegeven van wat een expert-systeem is. Verder zullen we twee mogelijke implementaties van expert-systemen beschrijven. In de eerste plaats zullen we neurale netwerken beschrijven. De algemene kennis over de werking van neurale netwerken hebben we later nodig als we de aanpassingen gaan maken in het algoritme van Wang-Mendel. De ideeën die we daarbij gebruiken zijn afkomstig uit de theorie van neurale netwerken.

In de tweede plaats worden fuzzy systemen besproken omdat zij het uiteindelijke doel vormen van ons werk. Gegeven ons doel van een algoritme voor het maken van regels voor een fuzzy systeem, zullen we ook willen weten hoe deze fuzzy systemen werken.

3.1 Algemeen

Expert-systemen proberen oplossingen te geven voor problemen die normaal gesproken door experts opgelost worden [Rich, 1991]. Om dit te kunnen bewerkstelliggen, dient het expert-systeem te beschikken over een kennisbank[1]. Deze kennisbank moet bij voorkeur volledig zijn en in het algemeen efficiënt te gebruiken.

Een expert-systeem moet communiceren met zijn gebruiker. Als we willen dat deze interactie goed verloopt, dan dient het systeem aan de volgende kenmerken te voldoen:

[1] Oorspronkelijk werd er in expert-systemen geen gebruik gemaakt van een kennisbank. Het toevoegen van bepaalde 'context kennis' zorgt dat je de beste eigenschappen van parametrische modellen en parameter-vrije modellen samenvoegt [Bishop, 1995].

Redenering verklaren In veel gevallen zal een enkele uitkomst van een expert-systeem niet acceptabel zijn. De gebruiker zal willen weten waarom het systeem tot een bepaalde uitkomst is gekomen door middel van de afleidingen die hieraan ten grondslag hebben gelegen.

Nieuwe kennis vergaren Expert systemen staan of vallen met de juistheid van de kennisbank waarop zij gebaseerd zijn. In de praktijk zal de kennis van experts veranderen op basis van nieuwe gegevens en ervaringen die zijn verkregen. Een expert-systeem zal op een of andere manier dus ook zijn kennisbank moeten bijstellen. Dit kan gebeuren door interactie met een gebruiker, die expert is, of door het leren uit gegevens die worden aangeboden.

Hoe wordt een expert-systeem gebouwd? Eén manier is het interviewen van een expert en vervolgens op basis hiervan regels opstellen. Als dan het systeem gebouwd is moet het systeem iteratief verbeterd worden totdat het acceptabele resultaten geeft. Het is gebleken dat dit in het algemeen een tijdrovend en kostbaar proces is. We kunnen dus een winst creëren als we dit proces kunnen automatiseren.

Aanvankelijk berustte de werking van expert-systemen op het verwerken van symbolen. Dit bleef het dicht bij de gedachte om de computer als een brein in te zetten. De symbolen zijn middels regels in een dergelijk systeem met elkaar verbonden. Voorbeelden van programmeertalen die hieruit zijn voortgekomen zijn Lisp en Prolog. Lisp en Prolog verwerken symbolen en lijsten van symbolen. De klassieke logica met de daarbijhorende proposities en predicaten zijn in liggen ten grondslag aan hun werking.

Neurale netwerken en fuzzy systemen zijn beiden systemen die input-output functies benaderen [Kosko, 1992b]. Beide zijn lerende dynamische systemen. In tegenstelling tot statistische approximators benaderen ze een functie "zonder een wiskundig model"[2] over hoe de uitvoer van de invoer afhangt. Het zijn zogenaamde verdelingsvrije schatters (model-free estimators). Ze leren van 'ervaringen' bestaande uit numerieke en soms beschrijvende gegevens. Eén van de voordelen hiervan is dat een zelfde soort systeem voor vele toepassingen gebruikt kan worden.

In het bedrijfsleven hebben expert-systemen op het moment niet zo'n goede naam [Cox, 1995]. Dit is voornamelijk te wijten aan falende projecten in het verleden. Vaak werden te grote verwachtingen gewekt over de mogelijkheden van een expert-systeem. Vervolgens werd een expert-systeem gemaakt wat deze verwachtingen niet waar kon maken. Dit soort praktijken heeft niet bijgedragen aan de populariteit van expert-systemen. Een deel van het probleem ligt in de gebruikte gereedschappen om de expert-systemen te maken.

De kracht van expert-systemen zit hem veelal in hun snelheid. De tijd gaat zitten in het ontwerp en trainen van het systeem. Oftewel daar waar het tijd mag kosten! Hierna heb je een systeem met een zeer snelle responstijd.

[2] Dit is niet strikt juist. Bishop beschouwd neurale netwerken als een soort tussenvorm, semi-parametrisch genaamd, die wel bepaalde 'context kennis' hebben [Bishop, 1995].

3.2 Symbolen en getallen

Mensen blijken in het algemeen niet in staat te zijn om de mathematische regels op te stellen die ons gedrag verklaren. We kunnen bijvoorbeeld een violist vragen hoe hij viool speelt. Hij zal ons dat kunnen vertellen maar het zal niet in wiskunde formules uit te drukken zijn [Kosko, 1992b].

Levende wezens zijn daarentegen zeer goed in staat om patronen te herkennen. De meest eenvoudige patronen betreffen het kunnen vinden van voedsel, het ontwijken van vijanden en dergelijke. Om diverse redenen probeert de mens daarnaast toch een aantal regels op te stellen. Zo hebben we een aantal regels voor grammatica, algemene wetten en wetenschap. Maar ons natuurlijk taalgebruik en onze rechtspraak zijn meer door cultuur beïnvloedt dan door regels.

Zullen we dus het werk van een expert willen automatiseren dan zullen we op een of andere manier dit denken in patronen/symbolen in ons systeem moeten verwerken. Een dergelijke opzet met symbolen stelt ons in staat om snel kennis in regels vast te leggen. Maar dit voorkomt wel dat we gereedschappen uit de numerieke wiskunde kunnen gebruiken en het systeem met eenvoudige elektrische circuits kunnen bouwen.

Figuur 3.1 laat verschillende mogelijke expert-systemen zien als we kennis opdelen naar gestructureerd (met regels) en ongestructureerd en we de expert-systeem methode opdelen naar symbolisch en numeriek. Alle systemen zijn 'model-free estimators', want gebruikers hoeven niet aan te geven hoe de uitkomst wiskundig afhangt van de invoer.

Kennis \ Systeemopzet	Symbolisch	Numeriek
Gestructureerd	AI expert-systeem	Fuzzy systemen
Ongestructureerd	——	Neuraal netwerk

Figuur 3.1: Indeling van 'model-free' systemen.

AI expert-systemen maken gebruik van gestructureerde kennis, als de maker er over kan beschikken, maar bewaard en bewerkt deze buiten de analytische en numerieke systeemopzet.

3.3 Neurale netwerken

De structuur van een neuraal netwerk is geïnspireerd door de architectuur van het menselijk brein. Dit is tot op heden maar ten dele gelukt. Je zou de volgende metafoor kunnen gebruiken. De huidige neurale netwerken verhouden zich tot het menselijk brein als een vliegtuig tot een vogel. Het vliegt, maar daar is dan ook alles mee gezegd.

Een neuraal netwerk heeft de volgende overeenkomsten met het brein:

- massieve parallelle architectuur bestaande uit een groot aantal relatief simpele units = neurons = neurodes = processing elements
- computing is niet algoritmisch
- geheugen is associatief en distributief
- lerend (toepassingen worden geleerd, niet geprogrammeerd)
- robuustheid t.a.v. ruis, verminking
- snel

Net als het menselijk brein kan een neuraal netwerk dingen herkennen die we niet exact kunnen definiëren. Wie zou er namelijk een gezicht kunnen beschrijven op zo'n manier dat deze altijd onmiskenbaar geïdentificeerd kan worden? We leren deze dingen gevoelsmatig door het aanschouwen van voorbeelden. We herkennen iets door het beschouwen van voorbeeld-gegevens. Net zoals een kind de kleur rood herkent van rode appels, rode auto's enzovoorts. Het kenmerk te kunnen herkennen zonder te definiëren, karakteriseert intelligent gedrag. Het stelt systemen in staat om te generaliseren.

Een neuraal netwerk slaat zijn gegevens op met gedistribueerde codering. Dit betekent dat gegevens niet op een bepaalde plaats in het netwerk staan, maar dat zij zijn verdeeld over het gehele netwerk. Dit heeft als voordeel dat een neuraal netwerk goed in staat is om partiële gegevens compleet te maken en om ruis uit gegevens te halen. Een ander voordeel is dat het netwerk hierdoor robuust wordt. Als een neuron uitvalt betekent dit niet dat bepaalde gegevens niet meer beschikbaar zijn. Hooguit kan dit betekenen dat bepaalde informatie niet meer volledig is.

Nadeel van deze manier van coderen is dat een storing in het netwerk kan optreden doordat verschillende gegevens dicht bij elkaar liggen. Gelijksoortige gegevens zouden dan kunnen gaan overlappen. Ook kunnen nieuwe gegevens oude gegevens wegdrukken of kunnen oude gegevens nieuwe gegevens verstoren. Als gevolg van deze storing is er een maximum aan gegevens wat je in een neuraal netwerk met gegeven aantal neuronen kan vastleggen zonder dat er degradatie van prestatie optreedt.

Neurale netwerken kunnen niet direct met gestructureerde kennis omgaan [Kosko, 1992b]. Neurale netwerken beelden een verzameling input/output waarden af op een black box netwerk. Tenzij we alle mogelijke input/output mogelijkheden uittesten weten we niet wat het neurale netwerk heeft geleerd. In het algemeen weten we ook niet wat het neurale netwerk 'vergeet' als we een nieuwe input/output combinatie aanbieden. We kunnen niet direct een regel als "Als de verkeersdrukte groot is een een bepaalde richting laat dan het verkeerslicht voor die richting langer op groen staan." in een neuraal netwerk stoppen. In plaats van dit moeten we

voldoende voorbeeld combinaties hebben van verkeersdrukte en de bijbehorende tijd dat het licht groen staat om het netwerk te trainen. Er is en wordt wel enig onderzoek op dit gebied gedaan, tot nu toe met weinig resultaat.

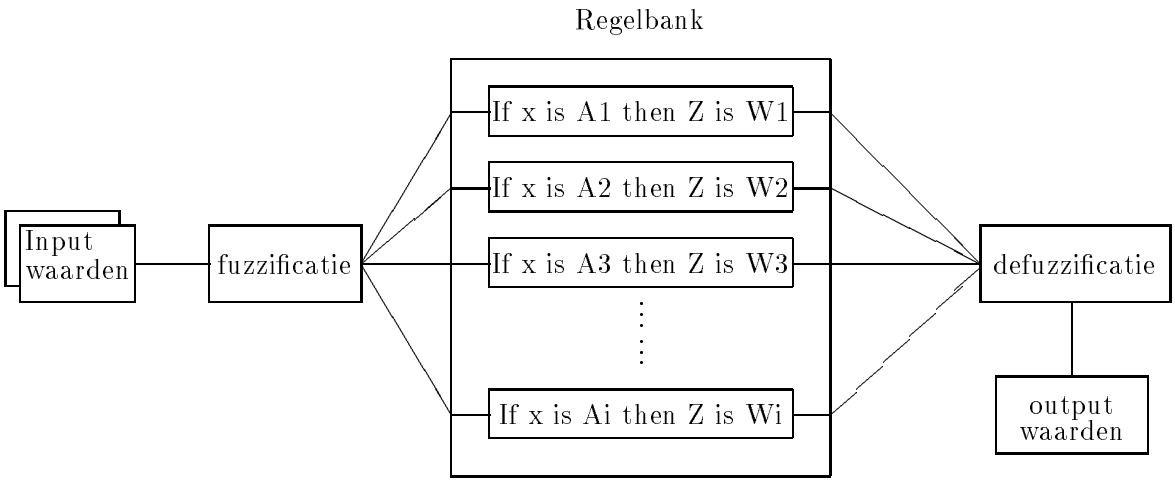
3.4 Fuzzy systemen

Fuzzy systemen combineren de hoge mate van flexibiliteit en kennisrepresentatie van de conventionele beslissingsondersteunende en expert-systemen met de kracht en het analytische vermogen van basis rekenmethoden [Cox, 1995]. Ze zijn als functie approximators bewezen toepasbaar en kunnen daarnaast, met een minimum aan regels, complexe, niet-lineaire, ruisbevattende systemen aan.

3.4.1 Werking van fuzzy systemen

Een fuzzy systeem ziet er globaal uit, zoals in figuur 3.2. Het systeem wordt gevormd door de volgende onderdelen:

- fuzzyficeerder
- regelbank
- defuzzyficeerder



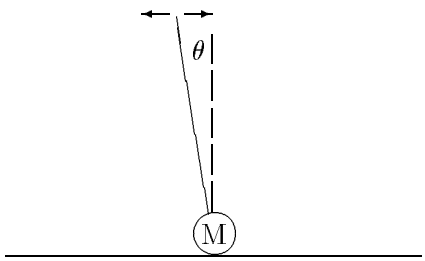
Figuur 3.2: Overzicht van een een fuzzy systeem

Deze onderdelen werken als volgt samen:

1. Een invoerwaarde of -waarden komt of komen het systeem binnen. De waarden moeten gefuzzyficeerd worden. Dit betekent, praktisch gezien, dat het fuzzy gebied of gebieden waarin de waarde valt moet worden bepaald.
2. Als het systeem voor de invoerwaarde(n) de bijbehorende gebieden hebben bepaald kan de regelbank geactiveerd worden. Afhankelijk van de gebieden waarin de invoer waarde(n) vallen zullen één of meer regels uit de regelbank geactiveerd worden.
3. Gegeven de geactiveerde regels, de invoerwaarden met de bijbehorende gebieden en de lidmaatschapswaarden in die gebieden moet er op een of andere manier één enkele uitvoerwaarde bepaald worden. Dit is het defuzzyficeren. Dit zou bijvoorbeeld kunnen gebeuren door de zo geheten centroïde defuzzyficatie methode [Kosko, 1992b] toe te passen.

3.4.2 Omgekeerde slinger

We zullen hier de werking van een fuzzy systeem toelichten aan de hand van een 'klassiek' voorbeeld voor fuzzy systemen, namelijk de omgekeerde slinger. De omgekeerde slinger moet in het midden staande worden gehouden door een elektromotor (M) aan de voet van de slinger (zie figuur 3.3). De slinger kan alleen naar links of rechts bewegen, niet naar voren of achteren. Dit type probleem is een typisch controle probleem, waar een mathematisch model geen oplossing bied [Kosko, 1992b]. Een mathematische oplossing blijkt namelijk bij de huidige stand der techniek niet adequaat te kunnen reageren.



Figuur 3.3: Schematische voorstelling van de omgekeerde slinger

Het fuzzy systeem bestaat uit twee invoerparameters en één uitvoerparameter. De eerste fuzzy invoervariabele is de hoek θ die de pendulum ten opzichte van de neutraalstand heeft. De middenstand heeft dus een hoek van 0. Positieve hoeken liggen aan de rechterkant van de middenstand en negatieve hoeken aan de linkerkant.

De tweede fuzzy invoer variabele is de verandering van de hoek, $\Delta\theta$. In de praktijk zullen we de verandering in de hoek approximeren door het verschil te nemen tussen de huidige hoek θ_t en de vorige meting van de hoek θ_{t-1}:

$$\Delta\theta_t = \theta_t - \theta_{t-1} \tag{3.1}$$

De uitvoer variabele is motorkracht, of de hoeksnelheid v_t. Deze snelheid kan positief of negatief zijn. Als de slinger naar rechts valt, dan zou de motorkracht negatief moeten zijn om te compenseren. Als de slinger naar links valt moet de motorkracht positief zijn. Als de slinger succesvol in het midden balanceert dan moet de motorkracht nul zijn.

Ieder van de variabelen kan positief, nul of negatief zijn. Stel voor dat we de grootte van een variabele uitdrukken in klein, middel en groot. Dit leidt tot zeven fuzzy verzamelingen:

NG: Negatief Groot
NM: Negatief Middel
NK: Negatief Klein
NU: Nul
PK: Positief Klein
PM: Positief Middel
PG: Positief Groot

Iedere regel in het Fuzzy Associative Memory voor de omgekeerde slinger wordt gevormd door een triple, zoals bijvoorbeeld (NM, NU; PM). Zo'n regel beschrijft hoe de motor kracht moet zijn, gegeven de hoek van de slinger en de verandering in de hoek van de slinger. Een FAM-regel combineert een hoek en verandering in de hoek van de slinger met een bepaalde motorkracht. De regel (NG, NU; PG) is als volgt te interpreteren:

> ALS de hoek van de slinger negatief is maar gemiddeld EN de verandering in de hoek is ongeveer nul, DAN moet de motorkracht positief zijn en gemiddeld.

Als we dit formeler opschrijven dan krijgen we:

> IF θ = NG AND $\Delta\theta$ = NU THEN v = PG

De FAM bank voor de slinger is een 7 bij 7 matrix met liguïstieke fuzzy verzameling elementen. De kolommen worden gevormd door de 7 fuzzy verzamelingen die de hoek, θ, uitdrukken. De rijen worden gevormd door de 7 fuzzy verzamelingen die uitdrukking geven aan de verandering in de hoek, $\Delta\theta$.

Iedere combinatie in de matrix kan als resultaat één van de 7 fuzzy verzamelingen voor de motorkracht hebben of heeft helemaal geen fuzzy verzameling als resultaat. Omdat een FAM regel een afbeelding of een functie is, is er voor iedere combinatie maximaal één resulterende fuzzy verzameling mogelijk. De 49 regels in de FAM bank vormen nu een deelverzameling van alle mogelijke combinaties ($7^3 = 343$) van regels. In de praktijk zullen deze 49 regels niet allemaal gedefiniëerd zijn.

Zullen we in de praktijk in het algemeen door middel van voorbeelddata de regels gaan samenstellen; bij een klein probleem als dit kan men ze nog grotendeels met gezond verstand afleiden. Ten eerste hebben we onze zogenaamde nulpunt. Als de slinger niet beweegt en de hoek nul is, dan doen we ook niets. Dit levert ons de steady state regel (NU, NU; NU) op.

Stel nu dat de slinger niet beweegt, dus $\Delta\theta = 0$. Als de slinger naar rechts afwijkt dan moet de motorkracht negatief zijn. Naarmate de slinger meer afwijkt van het nulpunt zal de motorkracht groter moeten zijn.

Stel nu dat de hoek van de slinger (θ) gelijk aan nul is, maar de slinger beweegt. Als de bewegingshoek negatief is, dan zal de slinger naar links gaan uitwijken. Dus hebben we een positieve motorkracht nodig om te compenseren. Wederom geldt: hoe groter de snelheid, hoe groter de motorkracht moet zijn.

Een positieve θ-waarde met een bijbehorende negatieve $\Delta\theta$ moet negatieve motorkrachten opleveren, omdat de slinger in de richting van het nulpunt beweegt. Het omgekeerde geldt dan ook: een negatieve θ met een positieve $\Delta\theta$ moet een positieve motorkracht opleveren.

$\Delta\theta$ \ θ	NG	NM	NK	NU	PK	PM	PG
NG				PG			
NM				PM			
NK				PK	NK		
NU	PG	PM	PK	NU	NK	NM	NG
PK			PK	NK			
PM				NM			
PG				NG			

Figuur 3.4: FAM bank voor de omgekeerde slinger

Op deze manier krijgen we de FAM zoals die in figuur 3.4 te zien is. De FAM bestaat uit 15 regels die in de praktijk uitstekend in staat zijn de slinger recht op te houden.

3.4.3 Voordelen van fuzzy systemen

Enkele redenen voor het toepassen van de fuzzy-set theorie in expert-systemen zijn [Zimmermann, 1989]:

- Voor de gebruikersinterface van een expert-systeem blijkt een communicatie op 'natuurlijke wijze' het meest geschikt te zijn. 'Natuurlijk' betekent hier: algemeen in de taal van de expert of van de eindgebruiker. De fuzzy-set theorie is in staat om 'natuurlijke' omschrijvingen als bijvoorbeeld lang, kort, veel en weinig te presenteren.
- De kennisbasis van expert-systemen is een verzameling van menselijke kennis. Aangezien menselijke kennis van nature vaag en onnauwkeurig is, bevat de kennisbasis van een expert-systeem een verzameling regels en feiten die voor een belangrijk deel noch absoluut zeker, noch absoluut consistent zijn. Het ligt daarmee voor de hand om dit gedeelte van de kennis niet door middel van 'scherpe' symbolen en concepten, maar door middel van fuzzy-sets weer te geven.

- Het gevolg van het werken met vage en onnauwkeurige kennis is dat het managen van onzekerheden een uiterst belangrijke rol speelt. Onzekerheid in de kennisbasis veroorzaakt onzekerheid in de conclusies en feiten. Dus moet de inferentiemachine van een expert-systeem in staat zijn om de omzetting van de onzekerheid vanaf de premissen tot en met de conclusies te analyseren en de resultaten op een begrijpelijke wijze aan de gebruiker mede te delen.

Voordelen van een fuzzy systeem waar het het ontwerp van het systeem betreft kunnen in de volgende drie punten worden samengevat [Cox, 1995]:

- Lagere applicatie-ontwikkelingskosten.
- Lagere applicatie-executiekosten.
- Lagere applicatie-onderhoudskosten.

In de praktijk is gebleken dat het grootste deel van de kosten van software engineering ligt in het onderhoud van applicaties [Vliet, 1991]. Tevens betekent een eenvoudiger ontwerp ook dat de kans op fouten in het ontwerp kleiner word, wat op zijn beurt weer een gunstige invloed op de onderhoudskosten heeft. Meer specifiek geldt, volgens Cox, in vergelijking met conventionele expert-systemen en beslissing ondersteunende systemen voor fuzzy systemen [Cox, 1995]:

- Ze zijn eenvoudiger te bouwen.
- Ze zijn eenvoudiger te begrijpen.
- Ze zijn eenvoudiger te verifiëren, te valideren en te tunen.
- Ze zijn stabieler.
- Ze zijn meer adaptief en robuust. Ze kunnen goed met ontbrekende regels overweg.
- Kunnen zeer niet-lineaire systemen approximeren.
- Kunnen adaptief zijn en zelf-regulerend.
- Er kan in korte tijd een prototype worden gemaakt.
- Hun gedrag kan worden verklaard.
- Ze kunnen omgaan met onzekere, onduidelijke en onnauwkeurige gegevens.
- Ze kunnen omgaan tegenstrijdige kennis van verschillende experts.
- Hebben een krachtiger redeneer systeem.
- Hebben minder regels nodig.

Een fuzzy informatie-systeem heeft significant hogere rekencapaciteiten dan met corresponderende symbolische expert-systemen. Dit komt onder meer door:

1. Parallelle verwerking.
2. Mogelijkheden om bewijzen voor en tegen een target te verzamelen.
3. Mogelijkheid om op een verzameling nivo te werken in plaats van op data nivo.
4. Mogelijkheid om met een hoge nauwkeurigheid met onduidelijke gegevens te werken.

Concluderend kunnen we stellen dat een fuzzy systeem compact en snel is, kennis op een hoog nivo representeert, fouttolerant is en goed met ontbrekende gegevens kan omgaan. Dit betekent praktisch dat:

- Minder ervaren kennis engineers complexere systemen kunnen bouwen.
- Minder tijd nodig is om kennis van experts te krijgen.
- Minder tijd nodig is om het ontwerp en prototype van een systeem te maken.
- Minder tijd nodig om fouten te vinden en op te lossen.
- Minder tijd nodig is voor de implementatie en het maken van eventuele uitbreidingen.
- Minder tijd nodig is voor het verifiëren en tunen van het model.

3.4.4 Nadelen van fuzzy systemen

Fuzzy systemen zijn niet het antwoord voor alle vragen. In onderstaande staan enkele vergelijkingen met andere systemen. Er worden enkele situaties geschetst waarin het verstandig is om geen fuzzy systeem te gebruiken. Tevens wordt er dan aangegeven wat vervolgens argumenten zouden kunnen zijn om toch te besluiten gebruik te maken van een fuzzy systeem [Cox, 1995].

Ten opzichte van lineaire systemen

Een fuzzy systeem is een heuristiek systeem. Dit wil zoveel zeggen dat een fuzzy systeem in het algemeen niet de meest optimale oplossing vind, maar wel een die goed bruikbaar is. Dit wil zeggen dat ze een proces approximeren. In het algemeen is dit geen beperking. Als we echter te maken hebben met een inzichtelijke lineaire functie, dan zal een fuzzy systeem, of ieder ander expert-systeem, kostbaarder zijn en minder nauwkeurig. Voor lineaire systemen met niet al te grote complexiteit geldt dus dat fuzzy systemen een tweede keus zijn. Een aantal punten die mee kunnen spelen om toch voor een fuzzy systeem te kiezen, daar waar een mathematisch model bekend is, zijn:

Het model werkt met fuzzy variabelen. Opvallend veel modellen maken gebruik van fuzzy variabelen. Model ontwerpers die niet bekend zijn met fuzzy logic omschrijven deze in mathematische termen. Deze ingewikkelde constructies kunnen in het algemeen eenvoudig en elegant met behulp van fuzzy variabelen beschreven worden.

Het model gebruikt data met ruis. Veel systemen hebben met een dataruimte te maken die ruis bevat. In lineaire modellen worden deze data in exacte functies gestopt, waardoor de ruis niet meer herkend wordt. Fuzzy systemen blijken uitermate geschikt om met data die ruis bevat om te gaan.

Het model wordt regelmatig herzien Als een model regelmatig moet worden herzien vanwege externe invloeden, dan zal in het algemeen gelden dat een fuzzy systeem meer geschikt is dan een lineair model. Een fuzzy systeem is gemakkelijker te onderhouden vanwege de robuustheid, eenvoudig onderhoud en fouttolerantie ten opzichte van hard gecodeerde lineaire modellen.

Het model is zelfverklarend. Een fuzzy systeem geeft ten opzichte van lineaire modellen een beter inzicht in de verbanden die bestaan in het model. Hierdoor zal een fuzzy systeem gemakkelijker te begrijpen en te verifiëren zijn.

De model ontwerper onderhoudt het model niet. Als de omgeving verandert en het model aanpassing behoeft, zal niet altijd de modelontwerper beschikbaar zijn. Een

fuzzy systeem zal in dat geval eenvoudiger te begrijpen en dus aan te passen zijn door de personen die met het onderhoud belast zijn.

Natuurlijke taalverwerking

Er is veel onderzoek gedaan naar de toepassing van fuzzy logic in de verwerking van natuurlijke taal. Ook naar het gebruik in database queries is gekeken om semantische dubbelzinnigheid te verminderen. In eerste instantie lijkt dit de ultieme toepassing van fuzzy logic. In de praktijk blijkt echter dat doordat natuurlijke taal inherent dubbelzinnig is, fuzzy logic niet te gebruiken is op dit gebied. In tegenstelling tot de dubbelzinnigheid in numerieke domeinen of continue model variabelen is de dubbelzinnigheid in de natuurlijke taal niet berekenbaar.

Wel kunnen we fuzzy logic gebruiken voor database en natuurlijke taal query systemen. We werken dan met relevance sets en frequenties. Als we Keywords In Context (KWIC) indices willen aanleggen voor grote lexicografische en tekst databases kunnen we deze termen aan fuzzy sets toewijzen om te bepalen hoe goed ze bij bepaalde concepten passen. Dit soort query verwerking wordt meta-analyse genoemd, omdat we redeneren over het materiaal, in tegenstelling tot de inhoud van het materiaal.

Het omgaan met ontbrekende informatie

In het algemeen wordt er van uit gegaan dat fuzzy logic met onnauwkeurige informatie omgaat. Die onnauwkeurige informatie is gebaseerd op 'gaten' in de onderliggende data. Fuzzy logic is echter geen tovenarij. Zonder een bepaald mechanisme om de gaten in de data op te vullen in een bepaald verband tot de kontekst, kan fuzzy logic zelf geen betrouwbare voorspelling doen over de missende informatie.

Data met ruis. Data die ruis bevat kan worden worden gebruikt door de aanname te maken dat data punten fuzzy nummers zijn. Fuzzy nummers representeren ruis of onzekerheid in de eigenlijke waarde van een data element. De data ontbreekt niet, maar ligt verspreid rond een centrale mogelijk waarde.

Ontbrekende data. Gegeven duidelijk onbekende referentiepunten, doet fuzzy logic weinig om een mogelijke domein ruimte te geven voor een datapunt. Hoewel fuzzy kwalificaties een waardevolle uitbreiding geven voor gevoeligheidsanalyse en optimalisatie, is het op zich geen methode voor het behandelen van een complexe verzameling van onbekenden. Als er genoeg historische data voorhanden is, kan een fuzzy model, gecombineerd met technieken voor tijdreeksanalyse, helpen bij het ontdekken van mogelijke waarden voor de ontbrekende model parameters. Fuzzy logic is berekend op het omgaan en uitdrukken van de onzekerheid en afwijkingen in de beschrijving van een parameter, niet met de afwezigheid van informatie over een parameter.

Modelleren van systemen met heel veel input- en outputvariabelen

Fuzzy systemen zijn gevoelig voor combinatorische explosie als het aantal input en output variabelen boven een bepaald aantal groeit. Dit heeft te maken met de complexiteit van het onderliggende model. Een model met 10 variabelen met ieder twee fuzzy sets (HOOG en LAAG) heeft een grove verdeling en dus een lage complexiteit. Als het zelfde model echter zo'n 8 fuzzy sets per variabele zou hebben, dan zou de complexiteit aanzienlijk hoger zijn. In eenvoudige fuzzy systemen groeit het aantal regels exponentieel met het aantal input en output variabelen. In het algemeen geldt dat voor een systeem met als gedrag:

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^p \tag{3.2}$$

waar er k fuzzy sets onder de variabelen liggen, er

$$k^{(n+p)-1} \tag{3.3}$$

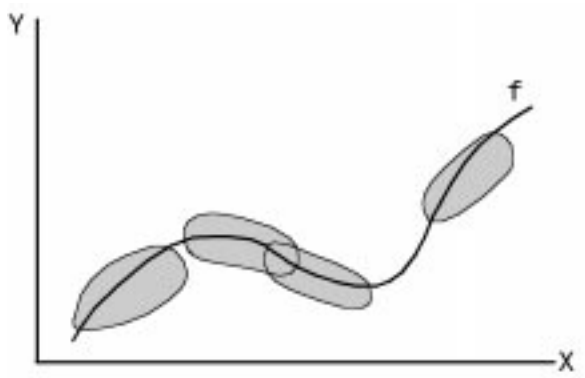
regels nodig zijn om het complete oppervlak van f af te dekken. Het exacte aantal regels hangt af van de soort functie, de vereiste nauwkeurigheid en de verdeling van de regels in de functie. Door Bishop wordt dit de 'curse of dimensionality' genoemd [Bishop, 1995]. Als de inputvariabele in meer regio's wordt verdeeld neemt het aantal mogelijke regels exponentieel toe. Omdat iedere cel minstens één trainingspunt moet bevatten om een regel op te leveren, houdt dit in dat de hoeveelheid trainingsdata ook exponentieel toe moet nemen. De hoeveelheid trainingsdata die je tot je beschikking hebt legt dus een bovengrens op aan het aantal regio's waarin je je variabelen kunt verdelen.

3.5 Fuzzy functieapproximatie

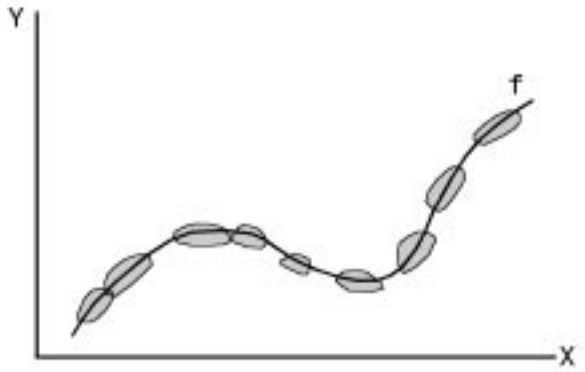
We kunnen de fuzzy wiskunde gebruiken om functies te approximeren. Dit is in feite wat er gebeurt bij toepassing van een fuzzy systeem. Een fuzzy expert-systeem kan opgevat worden als een functie-approximatie van een bepaalde input-output relatie.

3.5.1 Fuzzy gebieden en fuzzy regels

Fuzzy functieapproximatie [Kosko, 1992a] wordt gedaan door middel van het bedekken van de te approximeren functie met fuzzy gebieden. Voor een functie $f : X \rightarrow Y$ krijgen we fuzzy dan een verband tussen gebieden uit X en gebieden in Y. Initieel zullen deze gebieden relatief groot zijn en zullen dientengevolge een grote relatieve fout hebben. Naarmate je meer in staat bent om het aantal gebieden te vergroten en in grootte te laten afnemen zul je beter in staat zijn om de functie te approximeren. Vergelijk hiervoor de figuren 3.5 en 3.6. In de eerste figuur hebben we nog slechts enkele gebieden en deze zijn vrij groot. De functie wordt hier nog zeer grof geapproximeerd. In de tweede figuur hebben we al veel meer gebieden, die een stuk kleiner zijn, en deze zijn ook al veel beter in staat de te approximeren functie te volgen.



Figuur 3.5: Begin van approximatie



Figuur 3.6: Gevorderde approximatie

Een fuzzy regel is een beschrijving van een fuzzy gebied in het functiedomein. Een fuzzy regel geeft een relatie tussen fuzzy deelverzamelingen weer. Een fuzzy regel heeft de vorm: If A then B. Hierin is A de beschrijving van de invoerruimte en B de beschrijving van de uitvoerruimte.

Als we een inputwaarde hebben zullen we daar met onze functieapproximatie een bijbehorende uitvoerwaarde willen uitrekenen. Omdat de fuzzy regel een geheel gebied beschrijft, zal dit alleen niet genoeg uitsluitsel geven. Er moet dus een manier zijn om, gegeven de invoerwaarde en de fuzzy regel, een uitvoerwaarde te berekenen. Dit noemen we defuzzyficatie. Wat er gebeurt is dat de invoerwaarde en de fuzzy regel op een zodanige manier worden gecombineerd dat we een goede uitvoerwaarde krijgen. Gegeven een verband tussen een X-ruimte en een Y-ruimte werkt het systeem zoals in figuur 3.7 te zien is.

Figuur 3.7: Werking van een fuzzy regel met defuzzyficatie

3.5.2 Defuzzificatie

Er zijn meerdere manieren om onze fuzzy functie te defuzzificeren. Elk van deze mogelijkheden heeft zijn voor- en nadelen. We zullen nu enkele methoden van defuzzificatie bespreken.

Maximale lidmaatschap defuzzificatie

Dit is de meest eenvoudige vorm van defuzzificatie. Hierin wordt element y_{max} gekozen die de maximale lidmaatschap in de uitvoerverzameling B heeft [Kosko, 1992b]:

$$m_B(y_{max}) = \max m_B(y_j) \tag{3.4}$$

De 'maximum-likelihood' parameter schatting uit de kansberekening ligt ten grondslag aan deze defuzzyficatie methode. Bij deze methode bepaal je de waarschijnlijkheid dat een bepaalde uitkomst hoort bij een bepaalde schatter [Bain, 1989].

Een nadeel van deze methode is dat er niet een absoluut maximum hoeft te bestaan. Er zijn situaties waarbij je voor verzameling B een functie in de vorm van een driehoek met afgekapte top heb. Op dat moment heb je een lijn die het maximum is. Een mogelijke oplossing zou kunnen zijn om in zulke gevallen het midden van de lijn te nemen.

Een tweede nadeel is dat deze methode informatie negeert die in de vorm van verzameling B opgesloten ligt.

Centroïde defuzzificatie

Bij de centroïde defuzzificatie methode wordt de reële uitvoerwaarde bepaald door een gewogen convexe combinatie van de lidmaatschapswaarden met de waarden van de uitvoerruimtes te nemen [Kosko, 1992b]:

$$y = \frac{\sum_{j=1}^{p} y_j m_B(y_j)}{\sum_{j=1}^{p} m_B(y_j)} \tag{3.5}$$

y is hierin te vergelijken met de verwachte waarde van een kansfunctie.

De centroïde defuzzyficatie methode heeft niet de nadelen zoals de maximale lidmaatschap defuzzyficatie die heeft. De fuzzy centroïde is uniek en gebruikt alle informatie welke in de

uitvoerverzameling B opgesloten ligt. In veel gevallen zullen we echter de discrete sommatie moeten vervangen door een integraal over een continue ruimte.

Bij de fuzzy systemen zoals we die hier bespreken wordt in het algemeen de laatste defuzzificatie methode gebruikt.

Hoofdstuk 4

Het genereren van fuzzy regels

> In the beginner's mind there are many possibilities. In the expert's mind there are few.

Shunryu Suzuki
Zen Mind, Beginner's Mind

4.1 Regels genereren uit numerieke data

In dit hoofdstuk wordt een beschrijving gegeven van een algoritme voor het genereren van fuzzy regels uit een dataset [Wang, 1991]. Dit algoritme creëert uit een database van gegevens een fuzzy associative memory (FAM). Het FAM kan vervolgens in een fuzzy systeem gebruikt worden.

In het algoritme wordt gebruikt gemaakt van invoerdata met de volgende vorm:

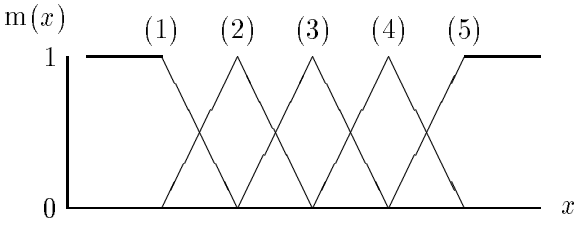
$$(x_1^{(1)}, x_2^{(1)}, y^{(1)}), (x_1^{(2)}, x_2^{(2)}, y^{(2)}), \ldots \tag{4.1}$$

Hier vormen x_1 en x_2 de verklarende waarden en is y de resulterende waarde (Economisch gezien zijn x_1 en x_2 de exogenen en is y de endogeen). Het doel van het algoritme is een functie-approximatie te vinden voor het volgende verband: $f(x_1, x_2) \rightarrow y$, gegeven de invoer-uitvoer paren uit 4.1. Hiertoe worden de volgende vijf stappen doorlopen.

4.1.1 Stap 1: Het verdelen van in- en uitvoerdomeinen in fuzzy regio's

Neem aan dat de domeinen van de variabelen x_1, x_2 en y er als volgt uitzien $[x_1^-, x_1^+]$, $[x_2^-, x_2^+]$ en $[y^-, y^+]$. Deze domeinen hebben de betekenis dat het te verwachten is dat de variabele in dit interval ligt. Dit betekent dus niet dat alle waarden voor de variabele in dit interval moeten liggen.

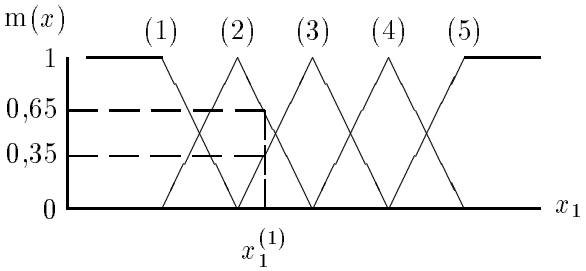
Elk van de variabelen wordt in $2N + 1$ (N kan voor iedere variabele verschillend zijn) regio's verdeeld. De lengte van ieder van de regio's behorende bij een variabele kan voor alle regio's gelijk zijn, maar mag ook verschillend zijn. Bij ieder van de regio's wordt een lidmaatschapsfunctie geconstrueerd. In het originele Wang-Mendel artikel heeft deze de vorm van een gelijkbenige driehoek. In de praktijk zijn ook andere vormen, zoals een belvorm, mogelijk. De top van de functie ligt in het midden van de bijbehorende regio's. De uiteinden van de benen liggen in het midden van de aangrenzende regio's. Dit geeft een beeld zoals in figuur 4.1.



Figuur 4.1: Regio's en lidmaatschapsfuncties van een variabele

4.1.2 Stap 2: Genereren van fuzzy regels uit de gegeven dataparen

Als de fuzzy regio's en hun bijbehorende lidmaatschapsfuncties bekend zijn kunnen we de lidmaatschapswaarden van onze data gaan bepalen. Bij ieder van de variabelen, $x_1^{(i)}$, $x_2^{(i)}$ en $y^{(i)}$, worden de lidmaatschapswaarden in de verschillende regio's bepaald.



Figuur 4.2: Lidmaatschapswaarde van x_1

Stel, we hebben voor $x_1^{(1)}$ zoals in de situatie van figuur 4.2. al zijn regio's bepaald. Voor de regio's (1), (4) en (5) heeft $x_1^{(1)}$ de lidmaatschapswaarde 0. Voor regio (2) heeft $x_1^{(1)}$ de lidmaatschapswaarde 0,65 en voor regio (3) de waarde 0,35. Door te maximaliseren over de lidmaatschapswaarden wordt er een regio aan de variabele toegewezen. In dit geval wordt regio (2) aan $x_1^{(1)}$ verbonden.

Stel dat $x_2^{(1)}$ in de bij die variabele regio's behorende regio (4) zijn maximale lidmaatschapswaarde van 0,7 bereikt. Als $y^{(1)}$ zijn maximale lidmaatschapswaarde van 0,5 in regio (1)

heeft. Dan volgt de volgende fuzzy regel voor deze invoer-uitvoer data:

$$\text{IF } x_1 \text{ is (2) AND } x_2 \text{ is (4), THEN } y \text{ is (1)} \tag{4.2}$$

Op deze manier ontstaat voor ieder invoer-uitvoer datapaar een fuzzy regel. De gegenereerde regels zijn "AND" regels omdat we met numerieke data te maken hebben en er geldt dat de uitvoer alleen bij die combinatie van invoer behoort.

4.1.3 Stap 3: Een gewicht aan iedere regel toewijzen

Het resultaat van stap 2 is dat er evenveel regels als invoer-uitvoer paren zijn. Het is heel goed mogelijk dat er regels tussen zitten die tegenstrijdig met elkaar zijn. Dit wordt als volgt opgelost. Aan iedere regel wordt een gewicht toegekend. Van een groep met conflicterende regels wordt alleen die regel bewaard die het hoogste gewicht heeft. Op deze manier worden de conflicten verwijderd en tevens het aantal regels sterk verminderd.

We wijzen op de volgende manier een gewicht aan een bepaalde regel toe. Uitgaande van de regel "IF x_1 is A AND x_2 is B, THEN y is C" wordt het gewicht van de regel, aangeduid met *D(Regel)*, gedefinieerd als:

$$D(\text{Regel}) = \mathrm{m}_A(x_1)\,\mathrm{m}_B(x_2)\,\mathrm{m}_C(y) \tag{4.3}$$

Regel 1 heeft dan het volgende gewicht:

$$\begin{aligned} D(\text{Regel 1}) &= \mathrm{m}_{(2)}(x_1)\,\mathrm{m}_{(4)}(x_2)\,\mathrm{m}_{(1)}(y) \\ &= 0,65 \times 0,7 \times 0,5 \\ &= 0,2275 \end{aligned} \tag{4.4}$$

In de praktijk zullen we vaak van te voren al enige informatie over het belang van de data hebben. We hebben bijvoorbeeld een expert laten aangeven welke dataparen belangrijk zijn en welke van minder belang zijn. Een expert kan bijvoorbeeld aan ieder datapaar een gewicht toekennen.

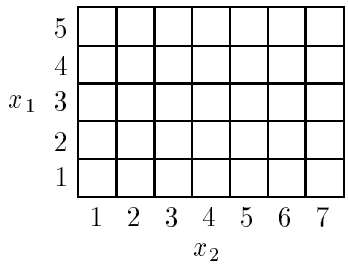
Stel dat het datapaar $(x_1^{(1)}, x_2^{(1)}, y^{(1)})$ als gewicht $m^{(1)}$ door de expert toegewezen heeft gekregen. Het totale gewicht van de regel wordt dan als volgt gedefinieerd:

$$D(\text{Regel 1}) = \mathrm{m}_{(2)}(x_1)\,\mathrm{m}_{(4)}(x_2)\,\mathrm{m}_{(1)}(y)\,\mathrm{m}^{(1)} \tag{4.5}$$

Dus het gewicht van een regel wordt bepaald door gewichten van iedere variabele afzonderlijk en het gewicht van het datapaar als geheel. De door de expert gegeven gewichten stellen ons zo in staat om afwijkende data uit het proces te filteren. Als er geen gewichten door een expert aan de dataparen worden toegewezen worden ze gelijk aan 1 verondersteld.

4.1.4 Stap 4: Een gecombineerde FAM bank maken

Als het domein van x_1 in 5 regio's is ingedeeld en x_2 in 7 regio's is ingedeeld, dan krijgt men een FAM zoals in figuur 4.3. Ieder van de $5 \times 7 = 35$ vakjes stelt een mogelijke regel voor.



Figuur 4.3: Schematische weergave van een FAM

De FAM wordt als volgt gevuld. De na stap 3 overgebleven regels worden op hun plaats in de FAM gezet. Als er in een hokje meer dan 1 regel komt dan wordt daarvan de regel met het hoogste gewicht gekozen. Een "AND"-regel vult één, maar een "OR"-regel vult een heel kruis in de FAM.

4.1.5 Stap 5: Een afbeelding bepalen op basis van de gecombineerde FAM bank

Om uitvoer y te bepalen bij gegeven invoervariabelen (x_1, x_2), volgen we de volgende defuzzificatie strategie. Ten eerste bepalen we voor de invoer het gewicht bij de i-de fuzzy regel door middel van de produktoperator. Op die manier krijgen we een gewicht m_O^i, behorende bij de uitvoer variabele gegeven de invoer (x_1, x_2).

$$\mathrm{m}_{O^i}^i = \mathrm{m}_{I_1^i}(x_1)\,\mathrm{m}_{I_2^i}(x_2) \tag{4.6}$$

O^i duidt hier de uitvoerregio van regel i aan en I_j^i duidt de invoerregio van regel i voor de j-de component aan. Dus voor regel 1 krijgen we:

$$\mathrm{m}_{(1)}^1 = \mathrm{m}_{(2)}(x_1)\,\mathrm{m}_{(4)}(x_2) \tag{4.7}$$

Vervolgens gebruiken we de volgende centroïde defuzzificatie formule om de uitvoervariabele te bepalen:

$$y = \frac{\sum_{i=1}^{K} \mathrm{m}_{O^i}^i\, \bar{y}^i}{\sum_{i=1}^{K} \mathrm{m}_{O^i}^i} \tag{4.8}$$

4.2 Een voorbeeld

Teneinde de werking van het algoritme duidelijk te maken, doen we het volgende. We kiezen een functie, op een bepaald domein. Vervolgens nemen we 20 punten het domein en berekenen hierbij de functiewaarden. Deze 20 punten en de bijbehorende functiewaarden vormen onze invoer voor het algoritme.

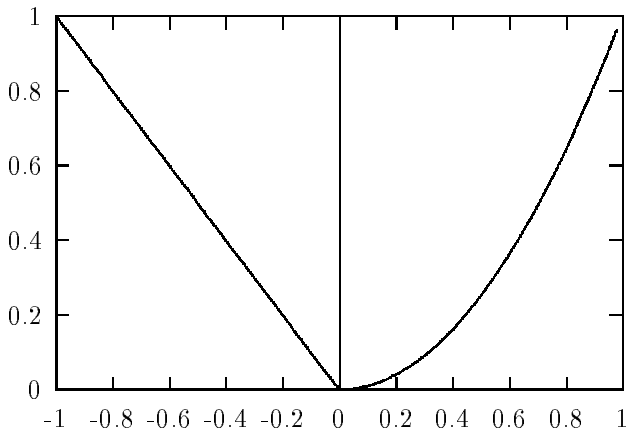
Met de FAM die op basis van deze 20 punten wordt gemaakt kunnen we dan bepalen hoe goed het algoritme de functie approximeert bij bepaalde combinatie's van gekozen regio's.

4.2.1 De functie

Als de te approximeren functie gebruiken we de volgende:

$$f(x) = \begin{cases} -x & \text{voor } x \leq 0 \\ x^2 & \text{voor } x \geq 0 \end{cases} \tag{4.9}$$

We gebruiken als domein $[-1, 1]$, dit levert ons het functieverloop op wat we in de volgende figuur zien.



Figuur 4.4: Verloop van de te approximeren functie

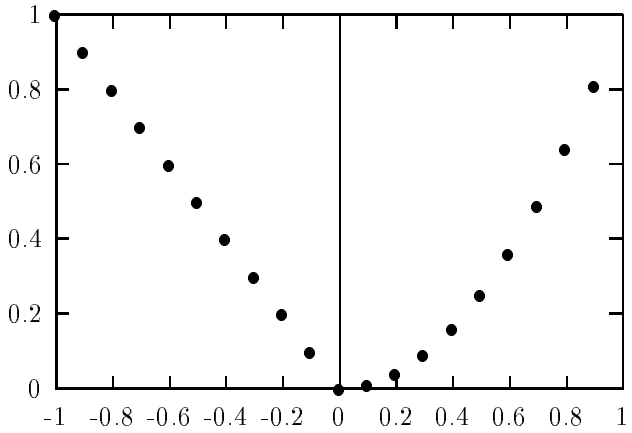
4.2.2 De punten

We kiezen 20 punten gelijkmatig verdeeld over het domein en berekenen hierbij de functiewaarden.

x	-1	-0,9	-0,8	-0,7	-0,6	-0,5	-0,4	-0,3	-0,2	-0,1
$f(x)$	1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1
x	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
$f(x)$	0	0,01	0,04	0,09	0,16	0,25	0,36	0,49	0,64	0,81

Tabel 4.1: Trainingsset voor het algoritme

Dit levert de volgende figuur op.



Figuur 4.5: Trainingsdata voor het algoritme

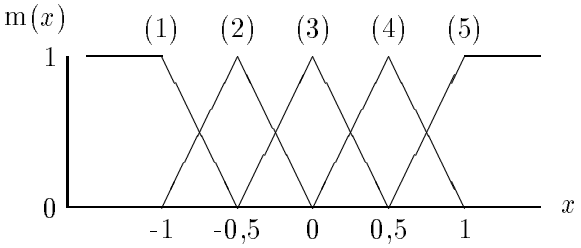
4.2.3 Het Algoritme

We gaan nu met onze trainingsset het algoritme doorlopen.

Stap 1: Verdelen in fuzzy functies

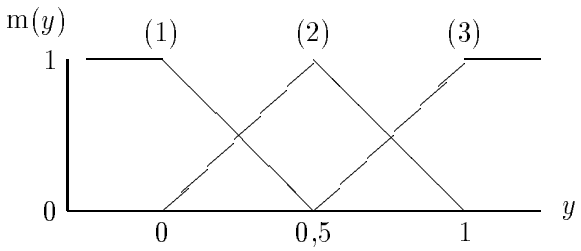
De eerste stap is om het domein en bereik van de functie in fuzzy functies te verdelen. We zullen het domein van de functie in 5 functies en het bereik van de functie in 3 functies verdelen. Deze waarden zijn volstrekt willekeurig gekozen en het is dan ook niet te zeggen of deze waarden de beste fit zullen opleveren.

Als we het domein van $[-1, 1]$ in 5 functies indelen, dan krijgen we de volgende situatie.



Figuur 4.6: Verdeling van het domein in 5 functies

Het bereik van de functie, gegeven het domein is [0,1]. De verdeling van het bereik in 3 functies ziet er dan als volgt uit.



Figuur 4.7: Verdeling van het bereik in 3 functies

Stap 2: Het maken van de fuzzy regels

We hebben onze fuzzy functies voor x en y gemaakt en kunnen onze fuzzy regels gaan maken met onze trainingsset.

Het eerste element uit onze trainingsset is (-1,1). Voor $x = 1$ krijgen we de volgende resultaten uit de fuzzy functies:

$$\begin{aligned} \mathrm{m}_x^{(1)} &= 1 \\ \mathrm{m}_x^{(2)} &= 0 \\ \mathrm{m}_x^{(3)} &= 0 \\ \mathrm{m}_x^{(4)} &= 0 \\ \mathrm{m}_x^{(5)} &= 0 \end{aligned}$$

Voor $y = 1$ krijgen we de functiewaarden:

$$\begin{aligned} \mathrm{m}_y^{(1)} &= 0 \\ \mathrm{m}_y^{(2)} &= 0 \end{aligned}$$

$$\mathrm{m}_y^{(3)} = 1$$

Hieruit volgt dan de fuzzy regel:

IF x is (1), THEN y is (3) (4.10)

Het tweede element uit de trainingsset is (-0,9, 0,9). Deze geeft de functiewaarden voor x:

$$\begin{aligned}
\mathrm{m}_x^{(1)} &= 0,8 \\
\mathrm{m}_x^{(2)} &= 0,2 \\
\mathrm{m}_x^{(3)} &= 0 \\
\mathrm{m}_x^{(4)} &= 0 \\
\mathrm{m}_x^{(5)} &= 0
\end{aligned}$$

En voor y:

$$\begin{aligned}
\mathrm{m}_y^{(1)} &= 0 \\
\mathrm{m}_y^{(2)} &= 0,2 \\
\mathrm{m}_y^{(3)} &= 0,8
\end{aligned}$$

Deze levert ook de regel:

IF x is (1), THEN y is (3) (4.11)

op.

Op dezelfde manier kunnen we de waarden bepalen voor de rest van onze trainingsset. We krijgen dan de resultaten uit tabel 4.2.

x	-1	-0,9	-0,8	-0,7	-0,6	-0,5	-0,4	-0,3	-0,2	-0,1
MAX i	(1)	(1)	(1)	(2)	(2)	(2)	(2)	(2)	(3)	(3)
m_x^i	1,0	0,8	0,6	0,6	0,8	1,0	0,8	0,6	0,6	0,8
y	1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1
MAX j	(3)	(3)	(3)	(2)	(2)	(2)	(2)	(2)	(1)	(1)
m_y^j	1,00	0,80	0,60	0,60	0,80	1,00	0,80	0,60	0,60	0,80
x	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
MAX i	(3)	(3)	(3)	(4)	(4)	(4)	(4)	(4)	(5)	(5)
m_x^i	1,0	0,8	0,6	0,6	0,8	1,0	0,8	0,6	0,6	0,8
y	0	0,01	0,04	0,09	0,16	0,25	0,36	0,49	0,64	0,81
MAX j	(1)	(1)	(1)	(1)	(1)	(2)	(2)	(2)	(2)	(3)
m_y^j	1,00	0,98	0,92	0,82	0,68	0,50	0,72	0,98	0,72	0,62

Tabel 4.2: Functiewaarden van de trainingsset

Stap 3: Gewichten berekenen

Als alle functiewaarden bekend zijn kunnen de gewichten van de regels berekend worden. Dit gebeurt door de functiewaarden van iedere regel met elkaar te vermenigvuldigen. Het resultaat staat in tabel 4.3

x	-1	-0,9	-0,8	-0,7	-0,6	-0,5	-0,4	-0,3	-0,2	-0,1
MAX i	(1)	(1)	(1)	(2)	(2)	(2)	(2)	(2)	(3)	(3)
m_x^i	1,0	0,8	0,6	0,6	0,8	1,0	0,8	0,6	0,6	0,8
y	1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1
MAX j	(3)	(3)	(3)	(2)	(2)	(2)	(2)	(2)	(1)	(1)
m_y^j	1,00	0,80	0,60	0,60	0,80	1,00	0,80	0,60	0,60	0,80
D	1,00	0,64	0,36	0,36	0,64	1,00	0,64	0,36	0,36	0,64
x	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
MAX i	(3)	(3)	(3)	(4)	(4)	(4)	(4)	(4)	(5)	(5)
m_x^i	1,0	0,8	0,6	0,6	0,8	1,0	0,8	0,6	0,6	0,8
y	0	0,01	0,04	0,09	0,16	0,25	0,36	0,49	0,64	0,81
MAX j	(1)	(1)	(1)	(1)	(1)	(2)	(2)	(2)	(2)	(3)
m_y^j	1,00	0,98	0,92	0,82	0,68	0,50	0,72	0,98	0,72	0,62
D	1,00	0,784	0,552	0,492	0,544	0,50	0,576	0,588	0,432	0,496

Tabel 4.3: Functiewaarden en gewichten van de trainingsset

Alle punten uit de trainingsset zijn even belangrijk. Het zijn tenslotte allemaal functiewaarden. In de gewichten voor ieder datapaar zitten dan ook geen verschillen en worden alle 1 verondersteld. De berekende gewichten zijn dus ook onze uiteindelijke bij de dataparen behorende gewichten.

Stap 4: Het samenstellen van de FAM

Als laatste moeten we nog de FAM gaan samenstellen. Als we naar onze data kijken, zien we dat de dataparen (-1,1), (-0,9 ,0,9) en (-0,8, 0,8) alle drie dezelfde regel opleveren, namelijk:

$$\text{IF } x \text{ is } (1), \text{ THEN } y \text{ is } (3) \tag{4.12}$$

Deze regel komt dus direct in de FAM.

Uit de dataparen (-0,7, 0,7), (-0,6, 0,6), (-0,5, 0,5), (-0,4, 0,4) en (-0,3, 0,3) volgt de volgende regel voor de FAM:

$$\text{IF } x \text{ is } (2), \text{ THEN } y \text{ is } (2) \tag{4.13}$$

De dataparen (-0,2, 0,2), (-0,1, 0,1), (0, 0), (0,1, 0,01) en (0,2, 0,04) leveren de volgende regel op:

$$\text{IF } x \text{ is } (3), \text{ THEN } y \text{ is } (1) \tag{4.14}$$

De dataparen (0,3, 0,09), (0,4, 0,16), (0,5, 0,25), (0,6, 0,36) en (0,7, 0,49) leveren twee mogelijke regels op:

$$\begin{array}{l}\text{IF } x \text{ is } (4), \text{ THEN } y \text{ is } (1)\\ \text{IF } x \text{ is } (4), \text{ THEN } y \text{ is } (2)\end{array} \tag{4.15}$$

De tweede regel heeft het hoogste gewicht, namelijk 0,588. De tweede regel zal dus in de FAM komen en de eerste zal komen te vervallen.

De laatste twee dataparen, (0,8, 0,64) en (0,9, 0,81), leveren ook twee mogelijke regels op:

$$\begin{array}{l}\text{IF } x \text{ is } (5), \text{ THEN } y \text{ is } (2)\\ \text{IF } x \text{ is } (5), \text{ THEN } y \text{ is } (3)\end{array} \tag{4.16}$$

Hier heeft de tweede regel het hoogste gewicht van 0,496.

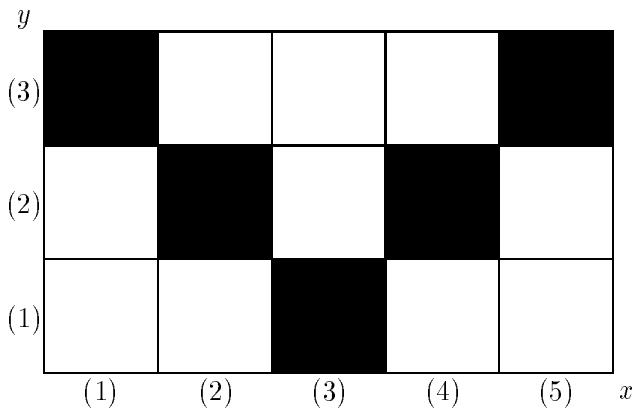
We hebben nu al onze regels gehad en hieruit is de volgende gecombineerde FAM ontstaan:

Regel
IF x is (1), THEN y is (3)
IF x is (2), THEN y is (2)
IF x is (3), THEN y is (1)
IF x is (4), THEN y is (2)
IF x is (5), THEN y is (3)

Tabel 4.4: Gecombineerde FAM

4.2.4 Resultaten

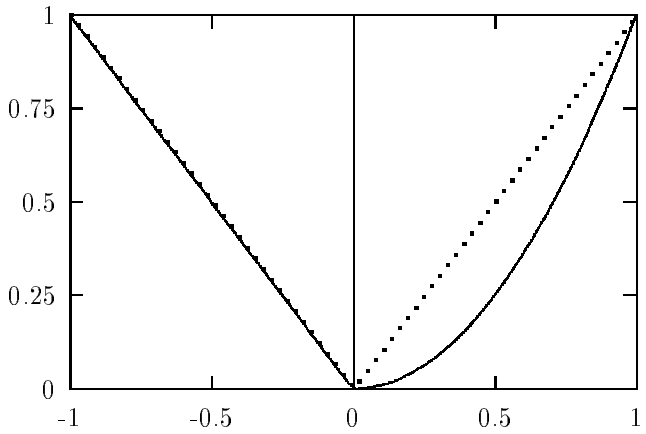
Als we de gebieden die deze regels uit de FAM beschrijven in een tekening zwart maken, dan zien we al het globale verloop van de functie verschijnen (zie figuur 4.8).



Figuur 4.8: Grafische representatie van de FAM

Waar we werkelijk in geïnteresseerd zijn, is hoe goed de FAM werkt. Daartoe pakken we 100 functiepunten en berekenen met behulp van de formule uit stap 5 van het Wang-Mendel-algoritme de bijbehorende waarden. Deze kunnen we vervolgens afzetten tegen de werkelijke functiewaarden.

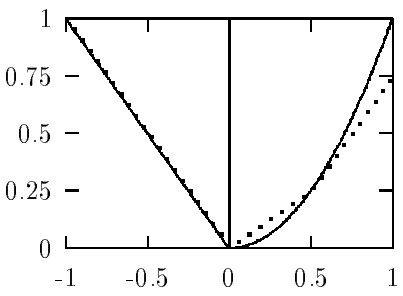
In de volgende figuur zien we onze eigenlijke functie getekend en daarnaast de geapproximeerde functie.



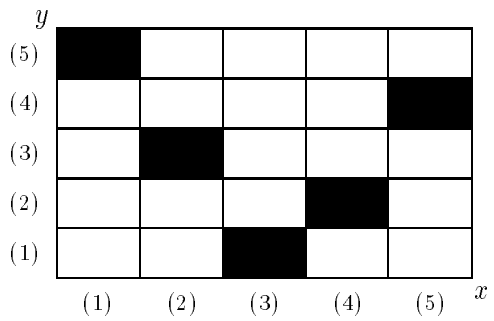
Figuur 4.9: Geapproximeerde functie bij verdeling van x in 5 en y in 3 fuzzy functies.

We zien dat de approximatie op het gedeelte [0,1] nog niet echt goed gaat. De oplossing om een betere approximatie te krijgen is om je domein in meer stukjes te verdelen. Oftewel, we moeten in de eerste stap van het Wang-Mendel-Algoritme het domein van x en y in meer fuzzy functies verdelen. Zoals we reeds in paragraaf 3.5 hebben gezien geldt dat als je je domein en bereik in meer gebieden opdeelt, je in principe beter kan approximeren[1].

We hebben het Wang-Mendel-Algoritme nog enkele keren toegepast met verschillende aantallen regio's waarin x en y verdeeld werden. Als trainingsset zijn telkens dezelfde 20 punten gebruikt uit tabel 4.1. Voor de controle zijn ook weer dezelfde honderd punten gebruikt die zijn gebruikt voor het maken van figuur 4.9. Figuren 4.10 tot en met 4.21 laten de resultaten van deze experimenten zien. Naast iedere approximatie staat ook telkens de gebruikte FAM afgebeeld.

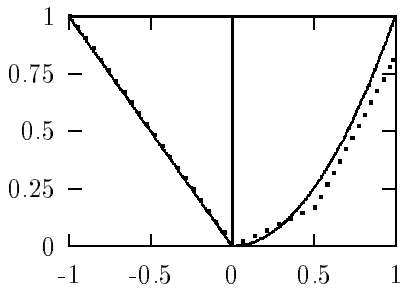


Figuur 4.10: Geapproximeerde functie bij verdeling van x in 5 en y in 5 fuzzy functies.

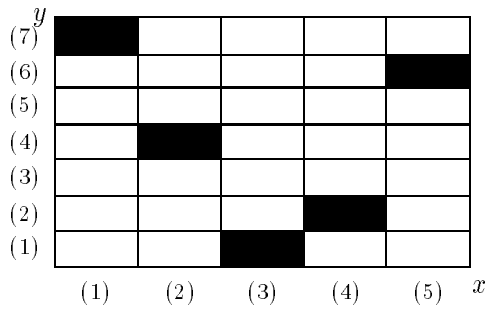


Figuur 4.11: FAM bij verdeling van x in 5 en y in 5 fuzzy functies.

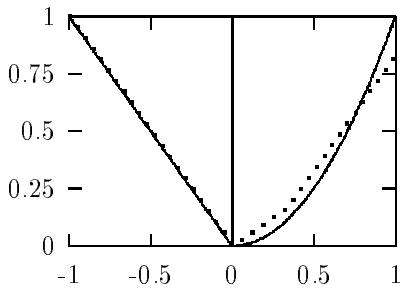
[1] Je hebt hier te maken met een representatieprobleem. Of je ook daadwerkelijk beter approximeert is een aparte kwestie.



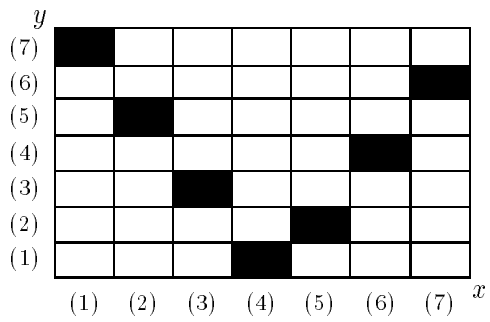
Figuur 4.12: Geapproximeerde functie bij verdeling van x in 5 en y in 7 fuzzy functies.



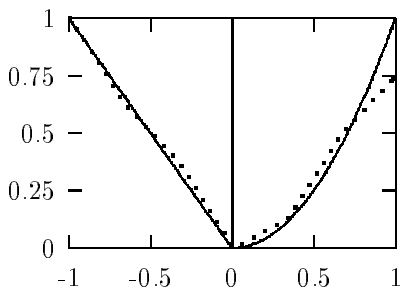
Figuur 4.13: FAM bij verdeling van x in 5 en y in 7 fuzzy functies.



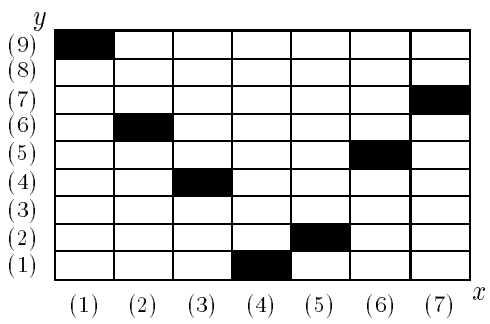
Figuur 4.14: Geapproximeerde functie bij verdeling van x in 7 en y in 7 fuzzy functies.



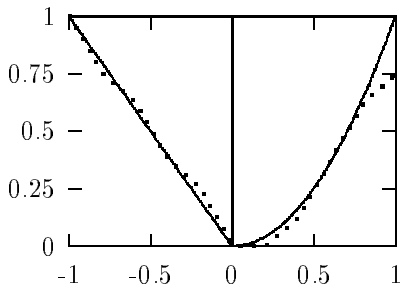
Figuur 4.15: FAM bij verdeling van x in 7 en y in 7 fuzzy functies.



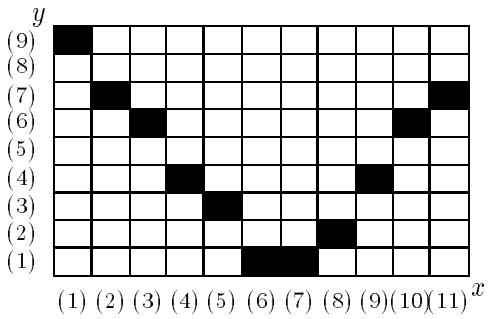
Figuur 4.16: Geapproximeerde functie bij verdeling van x in 7 en y in 9 fuzzy functies.



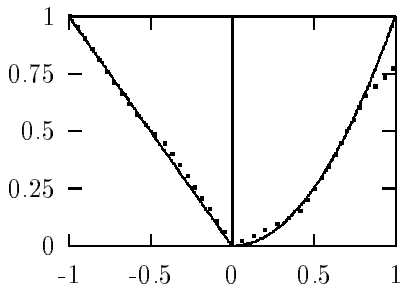
Figuur 4.17: FAM bij verdeling van x in 7 en y in 9 fuzzy functies.



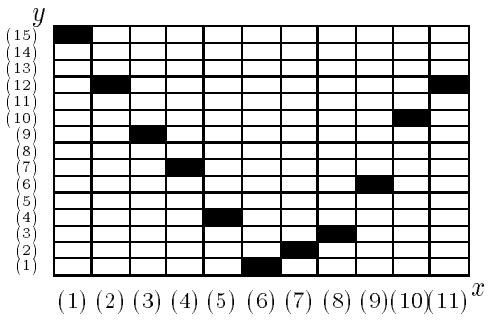
Figuur 4.18: Geapproximeerde functie bij verdeling van x in 11 en y in 9 fuzzy functies.



Figuur 4.19: FAM bij verdeling van x in 11 en y in 9 fuzzy functies.



Figuur 4.20: Geapproximeerde functie bij verdeling van x in 11 en y in 15 fuzzy functies.



Figuur 4.21: FAM bij verdeling van x in 11 en y in 15 fuzzy functies.

Zoals te zien is wordt er beter geapproximeerd naarmate de domeinen in meer stukjes zijn verdeeld. We zien tevens dat we aan de rechterkant van de grafiek de grootste afwijking houden. Dit wordt voornamelijk verklaard doordat het grootste punt uit onze trainingsset (0,9, 0,81) was. Vervolgens worden er bij de controle ook nog punten tussen 0,9 en 1 berekend. Gegeven het feit dat er op deze regio niet is getraind gaat het systeem extrapoleren vanuit de vorige regio. Dit kan echter tot gevolg hebben dat we op het interval (0,9, 1) een grote afwijking krijgen.

Wat we ook zien is dat op een gegeven moment de approximatie van de rechte lijn tussen (-1,1) en (0,0) afwijkingen krijgt. Als we naar de verdeling in x en y regio's kijken is dit echter wel te verklaren. De rechte lijn loopt van het maximum van y bij $x = -1$ naar het minimum van y bij $x = 0$. Je mag dus verwachten dat bij de regio van x waar -1 in valt de maximale regio van y hoort en dat bij de regio van $x = 0$ de minimale regio van y hoort. Gegeven het feit dat je een rechte lijn approximeert, mag je ook verwachten dat, wanneer je vanuit de regio waarin $x = -1$ valt door de tussenliggende regio's naar de regio van $x = 0$ gaat, dat dan de bijbehorende y-regio steeds evenredig daalt en deze van de maximale regio in de minimale regio voor y terechtkomt.

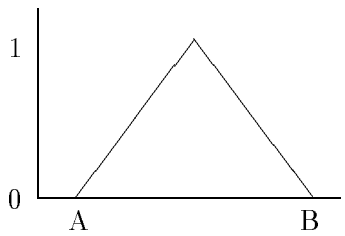
In dit laatste punt zit precies het probleem. Als we naar de figuren kijken, zien we dat het

in de figuren 4.13 en 4.15 precies gaat zoals we het verwachten. In figuur 4.17 zien we het al misgaan, het aantal x en y regio's staan geen verhouding toe, zodat het verloop in de y-regio's steeds evenredig daalt. Er vindt op een zeker moment een verschuiving plaats. De afwijking ten gevolge van deze verschuiving heeft direct gevolgen voor de nauwkeurigheid van de approximatie. In de figuren 4.19 en 4.21 zien we dit probleem ook optreden. Wel is het zo dat naarmate er meer regio's zijn de afwijking, veroorzaakt door zo'n verschuiving, minder groot is.

4.3 De approximatie

In de vorige paragraaf controleerden we de gegenereerde FAM's door 100 punten aan het systeem aan te bieden en hier de uitkomst van te bepalen. Het is echter zo dat je exact kan bepalen wat de uitvoer van het systeem is gegeven een bepaalde invoer.

Iedere regel in de FAM is een combinatie van invoerregio's x_i en bijbehorende resultaatregio's y_j. Bij regio $[A, B]$ behorend bij input x_i bestaat een fuzzy functie met een vorm als in figuur 4.22.



Figuur 4.22: Fuzzy functie

Deze fuzzy functie is het resultaat van de volgende functie:

$$\mathrm{m}(x_i) = \begin{cases} \frac{2}{B-A}(x_i - A) & \text{voor } A \leq x_i \leq \frac{A+B}{2} \\ 1 - \frac{2}{B-A}(x_i - \frac{A+B}{2}) & \text{voor } \frac{A+B}{2} \leq x_i \leq B \end{cases} \tag{4.17}$$

Deze functie is lineair. Als we nu de bijbehorende y gaan berekenen dan krijgen we de combinatie van een lineaire functie met een constante (het midden van de y-regio) wat weer een lineaire functie oplevert. Als we naar het Wang-Mendel algoritme kijken, dan zien we dat een x_i altijd in twee regio's zal vallen, omdat alle regio's elkaar voor 50% overlappen. Op die manier krijg je een tweede lineaire functie. Deze twee lineaire functies worden vervolgens gesommeerd wat als resultaat een lineaire functie zal opleveren. Voor $i > 1$ (meerdere inputwaarden) krijgen we ook een som van lineaire functies die in totaal ook weer een lineaire functie opleveren.

Uiteindelijk wordt iedere y_i dus opgebouwd uit een verzameling van lineaire functies welke

exact zijn af te leiden uit de functies voor de x regio's. Dit is een direct gevolg van het feit dat we voor de x regio's lineaire functies gebruiken, als we hier een niet-lineaire functies definiëren dan zal dit ook voor y_i een niet-lineaire functie opleveren.

Om dit duidelijk te maken zullen we nu met behulp van een voorbeeld het geval behandelen waarbij x in 5 regio's en y in 7 regio's werd verdeeld. Deze verdeling leverde vijf fuzzy regels op:

1. IF x is (1) THEN y is (7)
2. IF x is (2) THEN y is (4)
3. IF x is (3) THEN y is (1)
4. IF x is (4) THEN y is (2)
5. IF x is (5) THEN y is (6)

Dit wil zeggen als we met een waarde voor x in de eerste regio zitten, gaan we daarbij een waarde voor y in de zevende regio van y berekenen. En als we voor x in regio 2 zitten dan krijgen we een y-waarde in regio 4, enzovoorts.

Dit levert 4 verschillende stukken uit het domein van x op waarvoor een y functie kan worden berekend.

Interval	Werkende regels
$[-1,-0,5]$	1 en 2
$[-0,5,0]$	2 en 3
$[0,0,5]$	3 en 4
$[0,5,1]$	4 en 5

Tabel 4.5: Intervallen en bijbehorende regels op dat interval

Op het interval $[-1,-0,5]$ hebben we te maken met het dalende gedeelte van de fuzzy functie (1) en het stijgende gedeelte van de functie (2). Het dalende gedeelte van de fuzzy functie (1) is als volgt uit te drukken:

$$\mathrm{m}(x) = 1 - 2\ (x+1) \tag{4.18}$$

En het stijgende gedeelte van de fuzzy functie (2) heeft de volgende vorm:

$$\mathrm{m}(x) = 2\ (x+1) \tag{4.19}$$

Het resultaat van fuzzy functie (1) moet met het midden van fuzzy functie behorende bij de y-regio (7) worden vermenigvuldigd en het resultaat van fuzzy functie (2) met het midden van de functie behorende bij de y-regio (4). Het midden van de functie behorende bij de y-regio

(7) ligt bij 1 en dat van de functie van y-regio (4) bij $\frac{1}{2}$. Middels de centroïde defuzzyficatie methode kunnen we nu het resultaat voor y bepalen.

$$y = \frac{\sum_{i=1}^{2} \mathrm{m}_{O^i}^i \, \bar{y}^i}{\sum_{i=1}^{K} \mathrm{m}_{O^i}^i} \tag{4.20}$$

$$= \frac{(1 - 2\ (x+1)) \times 1 + (2\ (x+1)) \times \frac{1}{2}}{1 - 2\ (x+1) + 2\ (x+1)} \tag{4.21}$$

$$= (1 - 2\ (x+1)) \times 1 + (2\ (x+1)) \times \frac{1}{2} \tag{4.22}$$

$$= -x \tag{4.23}$$

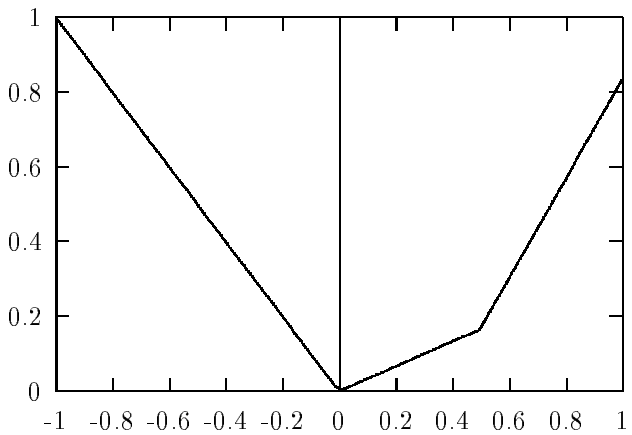
Op een analoge manier kunnen we nu de functies voor de andere drie gebieden bepalen. Dit geeft het volgende resultaat:

$$y = -x \text{ voor } 0,5 \leq x \leq 0 \tag{4.24}$$

$$y = \frac{1}{3}x \text{ voor } 0 \leq x \leq 0,5 \tag{4.25}$$

$$y = \frac{4}{3}x - \frac{1}{2} \text{ voor } 0,5 \leq x \leq 1 \tag{4.26}$$

Als we deze functies tekenen, zien we inderdaad dezelfde figuur ontstaan als bij de controle met de 100 punten. De steekproef met de 100 punten blijkt dus overbodig, omdat we bij een fuzzy systeem met lineaire driehoekige fuzzy functies exact de approximatie kunnen achterhalen.



Figuur 4.23: Berekende approximatie

4.4 Samenvattend

Een voor de hand liggende conclusie zou zijn, dat je in de praktijk maar een groot genoeg aantal fuzzy functies moet kiezen om een goede benadering te krijgen. De werkelijkheid is echter meestal minder eenvoudig. Ten eerste heb je in de praktijk niet de echte functie ter beschikking om je approximatie te controleren. Als dat zo zou zijn zouden we überhaupt al niet gaan approximeren. Ten tweede speelt in de praktijk het probleem van ruis. Ruis geeft afwijkingen in je data die je eigenlijk niet in het model wilt meenemen omdat het geen permanent karakter heeft. Hierdoor krijg het model slechte generalisatie eigenschappen. Wat gebeurt er nu als je je aantal fuzzy functies telkens maar gaat verhogen? Dan gaat het systeem op den duur de data zo nauwkeurig approximeren dat het ook ruis gaat betrekken in het systeem.

Er is dus blijkbaar een bepaald moment wanneer je moet stoppen met het verhogen van het aantal fuzzy functies om de data te beschrijven. In het volgende hoofdstuk zullen we een methode proberen te ontwikkelen om het juiste aantal fuzzy functies te bepalen.

Hoofdstuk 5

Het Extended Wang Mendel Algoritme

> Shuzan (926-992 A.D.) once held up his bamboo stick to an assembly of his disciples and declared: "Call this a stick and you assert. Call it not a stick and you negate. Now, do not assert or negate, and what would you call it? Speak! Speak!" One of the disciples came out of the ranks, took the stick away from the master, and breaking it in two, exclaimed, "What is this?"

Daisetz Teitaro Suzuki
An Introduction to Zen Buddhism

In het vorige hoofdstuk hebben we gezien dat het succes van een fuzzy system staat of valt met het op een juiste manier destilleren van de regels uit de de gegeven data. In dit hoofdstuk zullen we een methode voorstellen die ons in staat stelt op een geautomatiseerde manier de juiste verdeling in regio's te bepalen en zo de juiste regels af te leiden.

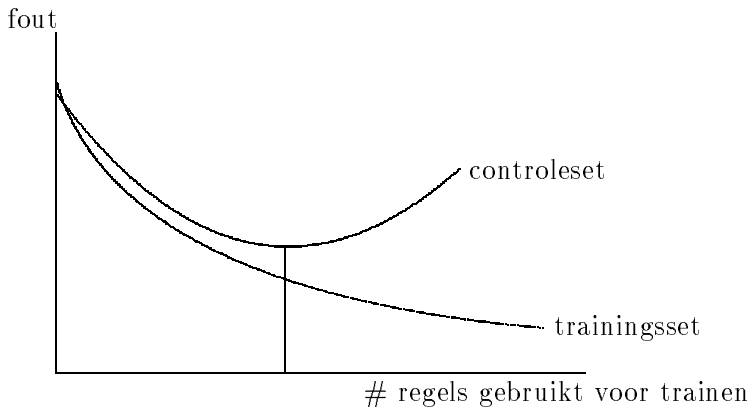
5.1 De ontwerp-paradox

Het maken van de regels voor een fuzzy systeem zadelt ons op met een tegenstrijdigheid. Ten eerste zouden we de domeinen van de variabelen in zo veel mogelijk regionen willen verdelen, want we hebben bij de fuzzy functie approximatie gezien dat onze approximatie dan steeds beter wordt. Aan de andere kant betekent dat wanneer onze approximatie steeds beter wordt bij de gegeven data, we ook de ruis die onvermijdelijk in deze data zit in ons model gaan incorpereren. Dit laatste zal in het algemeen als zeer onwenselijk worden ervaren.

We moeten een methode zien te vinden die hier mee om kan gaan. De eis die we stellen is dat we een zo goed mogelijke approximatie krijgen, maar dat we ruis in onze data buiten beschouwing laten. Wat we eigenlijk willen is het aantal regionen zo groot maken dat we nog net niet de ruis gaan fitten. In feite speelt dit probleem bij alle systemen die je baseert op historische data.

5.2 Cross validation

Bij het maken van neurale netwerken speelt het bovenstaande probleem ook een rol. Een van de methoden die men hier gebruikt is cross validation. Dit gaat als volgt in zijn werk [Fu, 1994]. Men deelt dan de data (input met daarbij behorende output) op in een trainingsset en in een controleset. Men gaat het neurale netwerk trainen met de trainingsset. Vervolgens controleert men de uitkomst van het netwerk met de trainingsset. Wat er nu gebeurt is dat naarmate men meer data uit de trainingsset aan het netwerk aanbiedt de fout voor de trainingsset steeds kleiner zal worden. De fout op de trainingsdata wordt in het begin ook steeds kleiner. Tot op een bepaalde hoogte waarna de fout weer begint toe te nemen. Men krijgt dan de volgende figuur:



Figuur 5.1: Verloop van de fout bij Cross validation

Het feit dat de fout voor de controleset in het begin daalt is volgens verwachting. De data uit de trainingsset slaan tenslotte op hetzelfde systeem als de data in de controleset. Als de fout niet zou dalen dan zou dit betekenen dat het neurale netwerk niet leert! Het feit dat de fout voor de controleset op een gegeven moment weer begint te stijgen is te wijten aan ruis in de trainingsdata. Het neurale netwerk begint vanaf dat moment ruis te fitten in de trainingsdata. Deze ruis heeft geen betrekking op het systeem en heeft dus tot gevolg dat de fout op de controledata gaat stijgen. In het algemeen wordt het punt waarbij de fout op de controleset minimaal is dan ook gezien als het punt waar men moet stoppen met het trainen van het neurale netwerk. Dit noemt men 'early stopping' [Fu, 1994].

5.3 Cross validation bij Fuzzy Systems

Het idee van cross validation, zoals in de vorige paragraaf geschetst, gaan we ook toepassen op fuzzy systemen, zij het in licht gewijzigde vorm. Ten eerste gaat een fuzzy systeem niet gedurende het trainen beter presteren, zoals een neuraal netwerk. Het principe van het neurale netwerk zorgt er voor dat deze in het begin vrij ruw schat en door middel van meer trainen

gaat 'leren'. Bij een fuzzy systeem is een regel echter wel of niet aanwezig. Teneinde een fuzzy systeem te kunnen valideren zal die dus eerst volledig getraind moeten zijn. Het leren is hier een 'one step procedure'.

We kunnen een fuzzy systeem dus niet zolang trainen totdat de fout op de controleset minimaal is. Het trainen van een fuzzy systeem gebeurt in één stap, of het systeem is volledig getraind of volledig niet! Het trainen van een fuzzy systeem gebeurt voor gegeven verdeling van de variabelen in een aantal regio's. Het gaat ons om de keuze van die aantallen. Het resultaat van zo'n keuze is een compleet fuzzy systeem. Om dus een goede keuze voor het aantal regio's van de variabelen te maken, moeten we deze complete fuzzy systemen met elkaar vergelijken. Het vergelijken van deze complete fuzzy systemen gebeurt naar analogie van de cross validation. We zoeken dat systeem dat met een bepaalde combinatie van het aantal regio's voor de variabelen de laagste fout geeft op de controleset.

De nieuwe methode die we hierna als Extended Wang-Mendel zullen aanduiden, werkt als volgt:

1. *Opsplitsen dataset.* We splitsen onze dataset op in een trainingsset en een controleset.
2. *Starten.* We beginnen met het meest eenvoudige fuzzy systeem. Dat wil zeggen een systeem waarbij voor alle variabelen het aantal regio's 3 is.
3. *Wang-Mendel.* Hier wordt het algoritme van Wang-Mendel toegepast op de trainingsdata. Het resultaat van deze toestand is een volledig fuzzy systeem.
4. *Controleren.* Alle fuzzy systemen worden nu gecontroleerd met de controleset. Hieruit volgt een foutwaarde voor dit specifieke fuzzy systeem.
5. *Zoekrichting.* Als zoekrichting wordt nu het "omliggende"[1] systeem genomen met de laagste fout, mits deze fout kleiner is dan die van het huidige systeem.
6. *Stopcriterium.* Het zoeken stopt als alle "omliggende" systemen een grotere fout hebben dan het huidige systeem.

Ad 1. Bij het splitsen van de dataset moet men er op letten dat dit op zo'n manier gebeurt dat zowel de trainingsset als de controleset het volledige domein en bereik van het systeem beslaan. In het algemeen zal de trainingsset ook uit meer elementen bestaan dan de controleset.

Ad 4. Gegeven de uitkomst van stap 3, wordt nu de controleset als input aan het systeem gegeven. Iedere uitkomst van het systeem wordt vergeleken met de uitkomst volgens de controleset. Van deze uitkomsten wordt het verschil genomen en dit wordt gekwadrateerd. Als we alle uitkomsten van de controleset hebben bepaald, kunnen we de som der kwadraten maken voor deze combinatie van controleset en fuzzy systeem.

[1] Tot en met systemen met 3 variabelen is van het begrip "omliggende" nog wel een goede voorstelling te maken. Met 3 variabelen moet je namelijk in 3D denken. Met meer variabelen wordt dit begrip voor ons mensen wat moeilijker voor te stellen. Wiskundig gezien houdt dit echter in dat je alle mogelijke permutaties gaat bepalen van het huidige aantal regio's plus of min 2 regio's. Dus voor een systeem met twee variabelen en huidige aantal regio's van 5 en 7 krijgen we dan de volgende combinaties: $\{3,5\}, \{3,7\}, \{3,9\}$, {5, 5}, {5, 9}, {7, 5}, {7, 7} en $\{7,9\}$.

5.4 De praktijk

Teneinde er achter te komen of bovenstaand verhaal inderdaad ook werkt hebben we een experiment uitgevoerd. Daarvoor hebben we dezelfde functie gebruikt als in het vorige hoofdstuk:

$$f(x) = \begin{cases} -x & \text{voor } x \leq 0 \\ x^2 & \text{voor } x \geq 0 \end{cases} \tag{5.1}$$

Wederom is er als domein $[-1, 1]$ gebruikt.

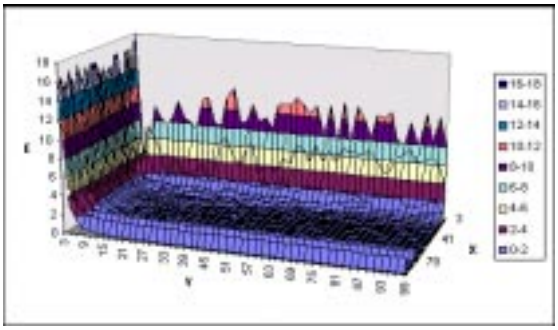
Er zijn vijfhonderd punten op het domein gekozen. Vervolgens moest er ruis gegenereerd worden. Deze ruis heeft een Normale verdeling. Daar ons alleen een random generator voor een Uniforme verdeling, ranf() [L'Ecuyer,], ter beschikking stond, hebben we op de volgende manier een pseudo-normale random generator gemaakt:

```
float RandomNormaal()
{
   for (int i=0; i < 10; i++)
      RandomNummer += ranf();

   RandomNummer -= 5.0;

   RandomNummer / 2.24;

   return RandomNummer;
}
```

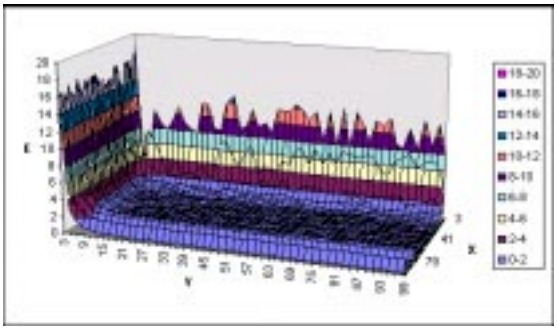
Het pseudo-normaal zijn van deze formule wordt uitgelegd in Appendix A. De ruis werd vervolgens met een factor c vermenigvuldigd om zodoende meer of minder ruis toe te voegen.

De resulterende 500 getallen hebben we opgedeeld in een trainingsset en een controleset, in een verhouding van 1:2. Met deze trainingsset en controleset hebben we de stappen zoals beschreven in de vorige paragraaf doorlopen. Extra informatie werd gevormd door bij de controleset ook de fout ten opzichte van de werkelijke waarde van y te berekenen. Dit gaf ons de mogelijkheid om vast te stellen of het systeem, dat we volgens het stappenplan zouden selecteren, ook op de originele data goed scoort.

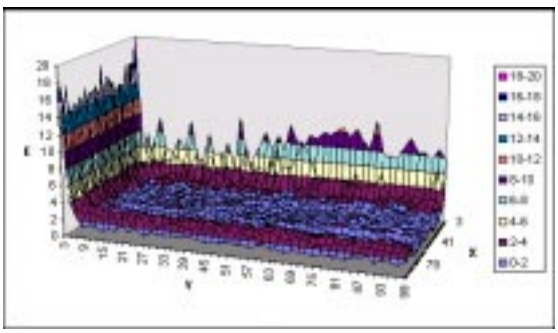
De overzichten van de fouten vinden we in de volgende 6 figuren:



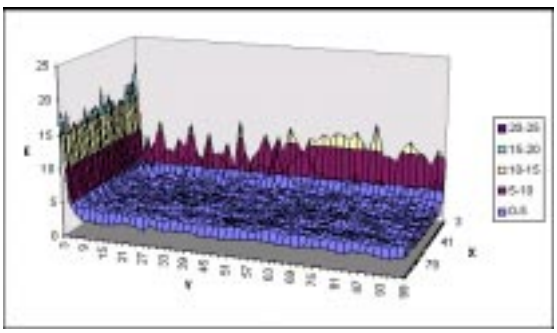
Figuur 5.2: Fout ten opzichte van exacte waarden, bij $c = 0,05$



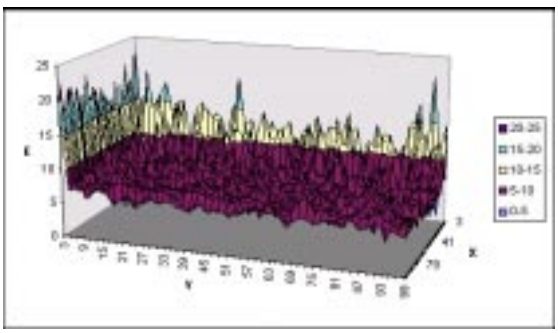
Figuur 5.3: Fout ten opzichte van controleset, bij $c = 0,05$



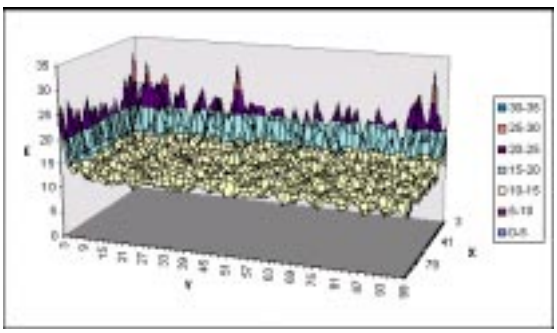
Figuur 5.4: Fout ten opzichte van exacte waarden, bij $c = 0,15$



Figuur 5.5: Fout ten opzichte van controleset, bij $c = 0,15$



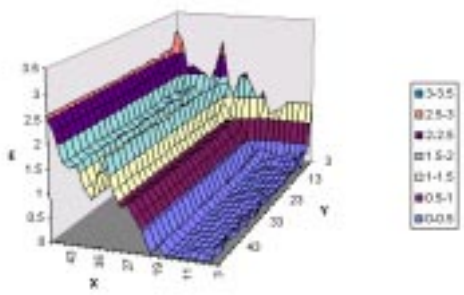
Figuur 5.6: Fout ten opzichte van exacte waarden, bij $c = 0,35$



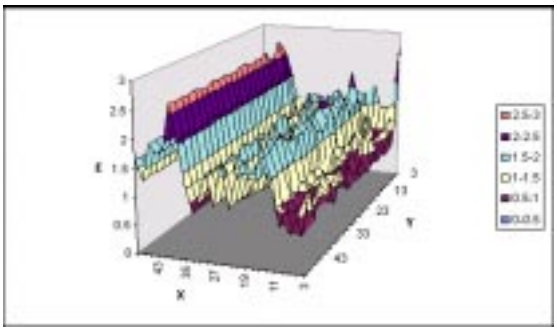
Figuur 5.7: Fout ten opzichte van controleset, bij $c = 0,35$

Wat in de vorige figuren opvalt is dat het verloop van de fout niet verliep zoals verwacht. In plaats van dat de fout op een gegeven moment weer steeg, kregen we een redelijk groot "platte vlak". Dit "plat vlak" lijkt verklaard te worden door de grote hoeveelheid data punten (500) die we hebben gebruikt. Ons maximaal aantal regio's was 100 en dus hebben we een behoorlijke puntenwolk waar doorheen de lijnen worden bepaald. We vermoeden dat het feit dat de vorm van onze fuzzy functies, driehoeken, die rechte lijnen in de approximatie opleveren, zie 4.3, bijgedragen hebben aan dit gedrag.

Om één en ander te verifiëren hebben we het experiment herhaald met 50 datapunten. De figuren 5.8 en 5.9 laten voor twee waarden van c de resultaten zien ten opzichte van de controleset. We zien dat nu inderdaad de fout op een gegeven moment weer gaat toenemen en dat dit eerder gebeurt naarmate er meer ruis in de data zit. Het maximale aantal regio's in deze figuren bedraagt 50. Gebruik van meer regio's heeft geen zin, omdat er dan zeker regio's zullen zijn waar geen regel voor gemaakt kan worden (er zijn maar 50 datapunten).



Figuur 5.8: Fout ten opzichte van controleset, bij $c = 0,05$



Figuur 5.9: Fout ten opzichte van controleset, bij $c = 0,35$

Het optreden van het zogenaamde "platte vlak" deed ons nogmaals nadenken over het stopcriterium van ons zoekalgoritme. We zeggen daar dat we stoppen zodra alle "omliggende" systemen een grotere fout hebben. Dit stopcriterium vergt de aanpassing dat we stoppen zodra alle "omliggende" systemen een fout hebben dat groter *of gelijk* is aan ons huidig systeem. Waarom zou je namelijk de variabele in meer regio's willen verdelen als dat geen lagere fout tot gevolg heeft. Het enige wat dit tot gevolg heeft is dat je last krijgt van 'the curse of dimensionality' [Bishop, 1995] die stelt dat met het toenemen van het aantal regio's de complexiteit van je systeem exponentiëel toeneemt.

5.5 Het Extended Wang-Mendel toegepast op een "real world" probleem

Als een 'grote' test hebben we het Extended Wang-Mendel algoritme losgelaten op het omgekeerde slinger probleem. In paragraaf 3.4.2 hebben we dit controle probleem reeds aan de

orde gebracht.

Teneinde aan testdata te komen hebben we de FAM zoals Kosko die samenstelt voor dit probleem geïmplementeerd [Kosko, 1992b], zie figuur 3.3. Met behulp van dit fuzzy systeem hebben we 100 datapunten gegenereerd. Deze 100 datapunten dienden vervolgens als input voor ons algoritme. De volgende stap was dan weer om ruis toe te voegen aan de data.

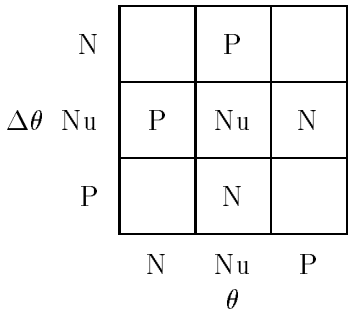
Als eerste werd het algoritme losgelaten op de dataset voordat er enige ruis was toegevoegd. Het EWM algoritme gaf in dit geval als beste keus voor de verdeling in regio's om alle variabelen in 3 regio's te verdelen. Als we de FAM van Kosko bekijken dan is dit resultaat heel goed te begrijpen. De verdeling binnen de FAM is zo dat de motorkracht in de uiterste gevallen maximaal is. In de nulstand is de motorkracht ook nul. Tussen de uiterste gevallen en nul daalt de motorkracht steeds evenredig. Eigenlijk vormt de functie van de motorkracht steeds een rechte lijn vanuit de uiterste gevallen tot in het nulpunt. Dit kan men al representeren met een regio rondom nul waar de motorkracht nul is en vervolgens in de beide uitersten een regio waar de motorkracht maximaal is, 3 regio's dus.

Vervolgens werd er ruis aan de data toegevoegd. Hierop nam het aantal regio's voor de uitvoer waarde toe tot 5. Hierna bleef het systeem stabiel met toenemende ruis. De volgende figuren laten de resulterende FAM's voor $c = 0$ en $c > 0$ zien. Als we een variabele in 3 regio's verdelen dan noemen we deze regio's:

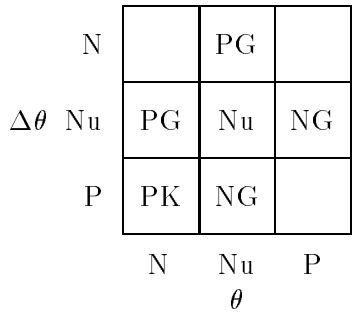
- Negatief (Ne)
- Nul (Nu)
- Groot (Gr).

Als een variabele in 5 regio's is verdeeld noemen we die:

- Negatief Groot (NG)
- Negatief Klein (NK)
- Nul (Nu)
- Positief Klein (PK)
- Positief Groot (PG)



Figuur 5.10: Resulterende FAM bij data zonder ruis, verdeling voor alle drie de variabelen in 3 regio's.



Figuur 5.11: Resulterende FAM bij data met ruis, verdeling van θ en $\Delta\theta$ in 3 regio's en de Motorkracht in 5 regio's.

5.6 Resultaten

Wat we zien is dat ons resulterend systeem eenvoudiger is dan het systeem wat Kosko [Kosko, 1992b] bedacht. We hebben ook al vastgesteld dat dit een logisch gevolg was van het feit dat Kosko zijn systeem feitelijk rechte lijnen vanuit de maxima's naar nul. Hierbij moeten we wel de kanttekening plaatsen dat dit alleen geldt als je de driehoek vorm gebruikt voor de fuzzy functies. Met de triangel vorm krijg je ene verzameling rechte lijnen voor de approximatie, zie 4.3. Als er zou worden gekozen voor een klokvorm voor de fuzzy systemen zou dit wel eens tot gevolg kunnen hebben dat het aantal regio's groter wordt in dit geval. De klokvorm zal gebogen lijnstukjes voor de approximatie tot gevolg hebben. Deze lijnstukken zullen in eerste instantie behoorlijk afwijken van de rechte lijnen die ze moeten approximeren. Als je nu het aantal regio's verhoogt krijg je kleinere lijnstukjes die dus ook kleinere boogjes vormen en de rechte lijnen beter benaderen.

Het Extended Wang Mendel algoritme geeft ons een algoritme dat vanuit numerieke data een Fuzzy Associative Memory genereert. Deze FAM vormt vervolgens de kennisbank voor een Fuzzy Expert Systeem. De enige vereiste voor het EWM algoritme is dat men beschikt over trainingsdata. Vervolgens is er dan in tegenstelling tot het Wang-Mendel-algoritme geen andere input voor het algoritme meer nodig. Automatisch wordt nu de meeste eenvoudige FAM, die voldoet voor het systeem, geproduceerd.

Bijlage A

Pseudo-normale verdeling

De gegevens in deze appendix zijn alle afkomstig uit [Bain, 1989].

```
float RandomNormaal()
{
   for (int i=0; i < 10; i++)
     RandomNummer += ranf();

   RandomNummer -= 5.0;

   RandomNummer / 2.24;

   return RandomNummer;
}
```

Het pseudo-normaal zijn van de functie RandomNormaal is als volgt uit te leggen. De functie ranf is een functie die uniform verdeeld is met als grenzen 0 en 1:

$$ranf \sim UNIF(a, b) = UNIF(0, 1) \tag{A.1}$$

Het gemmidelde van ranf is daarom gelijk aan:

$$E(ranf) = \frac{a+b}{2} = \frac{1}{2} \tag{A.2}$$

En de variantie van ranf gelijk is aan:

$$Var(ranf) = \frac{(b-a)^2}{12} = \frac{1}{12} \tag{A.3}$$

De centrale limiet stelling zegt dat als $X_1, \ldots, X_n$ trekkingen zijn uit een verdeling met gemiddelde μ en variantie $\sigma^2 < \infty$, dan is de gelimiteerde distributie van

$$Z_n = \frac{\sum_{i=1}^{n} X_i - n\mu}{\sqrt{n}\sigma} \tag{A.4}$$

standaard normaal, $Z_n \to Z \sim N(0,1)$ als $n \to \infty$.

Als we deze formule invullen voor ons geval dan krijgen we:

$$\frac{\sum_{i=1}^{10} X_i - 5}{\sqrt{10}0,5} \approx \frac{\sum_{i=1}^{10} X_i - 5}{2.24} \tag{A.5}$$

Welke pseudo-normaal is omdat $n \ll \infty$.

Bibliografie

[Bain, 1989] Bain, L.J. en Engelhardt, M., *Introduction to Probability and Mathematical Statistics*,p268,1989

[Berg, 1997] Berg, J. van den en Dijk, J. van, "How to create the simplest FAM-based Fuzzy System", *Proc. of ICONIP '97*, to be published in november, 1997

[Bishop, 1995] Bishop, C.M., *Neural networks for pattern recognition*,H1-H2,1995

[Cox, 1995] Cox, E.D. *Fuzzy logic for business and industry*,1995

[L'Ecuyer,] L'Ecuyer, P. en Cote, S., "Implementing a Random Number Package with Splitting Facilities", *ACM Transaction on Mathematical Software* 17:1 p98-111,

[Fu, 1994] Fu, L., *Neural networks in computer intelligence*,H13,1994

[Klir, 1995] Klir, G.J. en Yuan, B., *Fuzzy sets and fuzzy logic*, 1995

[Kosko, 1992a] Kosko, B., "Fuzzy systems as universal approximators", *Proc. of IEEE FUZZ-92*,Maart 1992

[Kosko, 1993] Kosko, B., *Fuzzy thinking*, 1993

[Kosko, 1992b] Kosko, B., *Neural networks and fuzzy systems*, 1992

[Pirsig, 1974] Pirsig, R.M., *Zen and the art of motorcycle maintenance*, 1974

[Rich, 1991] Rich, E. en Knight, K., *Artificial Intelligence*, H20, 1991

[Vliet, 1991] Vliet, J.C. van, *Software Engineering*, H11, 1991

[Wang, 1991] Wang, L.X. en Mendel, J.M., "Generating fuzzy rules by learning from examples", *IEEE International Symposium on Intelligence Control*, p263-268, 1991

[Zimmermann, 1989] Zimmermann, H.J., *The interface between artificial intelligence and operations research in fuzzy environment*, 1989

[Zadeh, 1965] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, p338-353, 1965

Index