Technisch rapport; TR-165

Ministerie van Verkeer en Waterstaat

Koninklijk Nederlands
Meteorologisch Instituut

Neurale netwerken versus lineaire regressie

een onderzoek naar de waarde van neurale netwerken in de meteorologische praktijk

R.M. Meuleman

De Bilt, 1994

Technisch rapport; TR-165

Verslag van afstudeeropdracht - Universiteit van Amsterdam.

De Bilt, 1994

Postbus 201
3730 AE De Bilt
Wilhelminalaan 10
Telefoon 030-206 911
Telefax 030-210 407

UDC: 551.509.3
551.509.51
681.324
ISBN: 90-369-2057-4
ISSN: 0169-1708

Voorwoord

Dit rapport is zowel een technisch rapport voor het Koninklijk Nederlands Meteorologisch Instituut, als een verslag van mijn afstudeeropdracht voor de Universiteit van Amsterdam. Hierbij bedank ik drs. S. Kruizinga, mijn stagebegeleider bij het Koninklijk Nederlands Meteorologisch Instituut, voor het begeleiden met mijn afstudeeropdracht, en iedereen die mij het mogelijk heeft gemaakt om mijn afstudeeropdracht te voltooien en dit rapport te maken. Verder bedank ik dr. A.J. van Es, mijn afstudeerdocent aan de Universiteit van Amsterdam, voor het begeleiden met mijn afstudeeropdracht.

Inhoudsopgave

1. INLEIDING

De meteoroloog heeft voor het opstellen van weersverwachtingen diverse gegevens beschikbaar. Voorbeelden van beschikbare gegevens zijn:

- de output van meerdere atmosferische modellen voor diverse verwachtingstijden, bijv. verwachtingen voor de volgende dag, verwachtingen voor over twee dagen, enz.

- recente waarnemingen die nog zijn binnengekomen nadat de programma's gebaseerd op atmosferische modellen zijn gestart met hun berekeningen.

Het is dan de taak van de meteoroloog om die gegevens te combineren tot één gedetailleerde verwachting die logisch aansluit bij de recente waarnemingen. Als extra moeilijkheid komt daar nog bij dat de output van de atmosferische modellen veelal niet die elementen kan bevatten waar de gebruiker belangstelling voor heeft. De meteoroloog dient dus ook nog te interpreteren. Dit proces is voor een groot deel subjectief maar bevat ook objectieve componenten.

Kort na de ontwikkeling van de numerieke atmosferische modellen is men in de Verenigde Staten gestart met het ontwikkelen van objectieve statistische hulpmiddelen voor de interpretatie van modeloutput. In 1972 werd door Glahn en Lowrie de zogenaamde Model Output Statistics (MOS) ontwikkeld (Glahn 1972) en later de LAMP methode (Glahn 1980) waarin ook recente waarnemingen worden meegenomen. Deze methoden vormen dus een objectieve ondersteuning van het subjectieve proces dat de meteoroloog uitvoert. De resultaten van deze methoden zijn in veel gevallen concurrerend met de resultaten die de meteoroloog bereikt. Ook nu worden nog steeds methoden ontwikkeld die gebaseerd zijn op deze benadering.

Er worden twee problemen gesignaleerd:

- MOS en LAMP zijn gebaseerd op de klassieke statistische technieken zoals lineaire regressie. Het is echter niet zeker of de relaties in de atmosfeer hiermee altijd beschreven kunnen worden.

- de atmosferische modellen zijn voortdurend in ontwikkeling en dus zijn de relaties tussen modeloutput en lokaal optredend weer aan verandering onderhevig. De statistische MOS relatie zou dus voortdurend opnieuw ontwikkeld moeten worden. Bij massale toepassing leidt dit tot logistieke problemen.

Voor beide problemen zouden neurale netwerken mogelijk een oplossing kunnen bieden. In principe kan een neuraal netwerk elke samenhang representeren. Daarnaast is het mogelijk om een neuraal netwerk zo op te zetten dat het voortdurend blijft leren en zich dus steeds aan kan passen aan de veranderende atmosferische modellen.

In dit afstudeeronderzoek worden twee neurale netwerken en hun theoretische achtergronden beschreven. Een aantal eigenschappen van neurale netwerken worden tegen deze achtergrond onderzocht:

- in het eerste gedeelte worden de neurale netwerken toegepast op een kunstmatig lineair regressie probleem waarvan de onderliggende regressie exact bekend is en waarvoor leer- en testdata gegenereerd kunnen worden.

- in het tweede gedeelte worden deze neurale netwerken toegepast op praktijk data uit de meteorologische praktijk met een overwegend lineair verband en een zwakke interactie tussen twee predictoren.

- in het derde gedeelte wordt het lerend vermogen onderzocht.

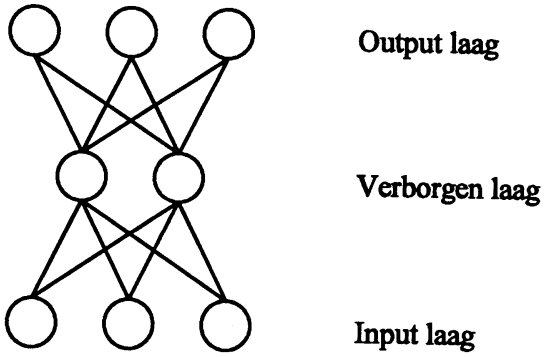
Dit onderzoek concentreert zich met name op de eigenschappen van neurale netwerken in vergelijking met de klassieke statistiek.

2. NEURALE NETWERKEN

Een neuraal netwerk is een netwerk van knooppunten (nodes, processing elements, units of neurons) die met elkaar verbonden zijn door middel van verbindingen met gewichten (synapsen). Elk knooppunt (behalve de knooppunten die als input gebruikt worden) krijgt als input een gewogen som van de outputs van andere knooppunten. De output is gelijk aan de waarde van de transferfunktie, berekend in die gewogen som.

Er bestaan supervised en unsupervised learning netwerken. Unsupervised learning netwerken, of self-organising networks (ART, Kohonen network), hebben alleen inputdata nodig. Deze netwerken worden gebruikt voor clustering, vector quantisatie en dimensie reductie.

Supervised learning netwerken hebben tijdens de leerfase ook outputdata nodig, waarmee het verschil tussen deze output en de door het netwerk berekende output berekend kan worden. Het verschil wordt geminimaliseerd door het netwerk aan te passen. Supervised learning netwerken bestaan uit feed-forward en recurrente netwerken. Bij recurrente netwerken (Hopfield network, Boltzmann machines) kunnen alle knooppunten met elkaar verbonden zijn.



Feedforward netwerk met verborgen laag

Een feed-forward netwerk (Adaline, Madeline, Perceptron) is een netwerk dat uit verschillende lagen bestaat. Er is een inputlaag, een outputlaag en eventueel verborgen lagen. De knooppunten in de inputlaag komen overeen met de input variabelen en de knooppunten in de outputlaag met de output variabelen. Voor beschrijvingen van deze methoden zie Cheng en Titterington (1994).

Om het verschil tussen de gewenste en de berekende output zo klein mogelijk te maken worden de gewichten aangepast. Dit kan op verschillende manieren. Bij General Regression Neural Network wordt de gewenste output al in de input laag gebruikt. Bij back-propagation wordt de gewenste output vergeleken met de berekende output en aan de hand hiervan worden de gewichten aangepast. Dit kan op verschillende manieren.

2.1 Regressie problemen

Als we een vergelijking zoeken tussen verschillende variabelen kunnen we lineaire regressie toepassen. Als er niet-lineaire verbanden zijn, zullen we grote fouten maken. Er zijn twee belangrijke methoden om een vergelijking te schatten. Niet-parametrische methoden worden voor situaties gebruikt als we niet weten met welke vergelijking we te maken hebben. Deze methode wordt in 2.2.2 behandeld. Niet-lineaire regressie is voor situaties als we wel de vergelijking weten, maar niet de bijbehorende parameters. Deze methode wordt in 2.3.2 behandeld.

2.2 General Regression Neural Network en de Nadaraya-Watson schatter

Hier volgt een beschrijving van GRNN en daarna de hier op lijkende Nadaraya-Watson schatter.

2.2.1 General Regression Neural Network

Dit type neuraal netwerk is voorgesteld door Donald Specht (1991) en wordt o.a. voor regressie toegepast. GRNN gebruikt een formule om de voorwaardelijke verwachting van Y bij gegeven X uit te rekenen. Voor een stochastische variabele Y en een stochastische vector $X=(X_1,\ldots,X_N)$ geldt

$$m(X) = E(Y|X) = \frac{\int_{-\infty}^{\infty} y\, f(X,y)\, dy}{\int_{-\infty}^{\infty} f(X,y)\, dy} \qquad (1)$$

,met f(x,y) de simultane dichtheid is van X en Y. Voor de afleiding zie appendix A.

Zowel X als Y kunnen vectoren zijn. Als Y uit meer dan één component bestaat, dan wordt de voorwaardelijke verwachting van Y componentsgewijs berekend. Deze formule bevat de simultane kansverdeling van X en Y. Deze simultane kansverdeling wordt aan de hand van een leerset geschat door Parzen schatting. Parzen schatting is een niet-parametrische schattingsmethode die een regressiefunctie schat door vele normale kansdichtheden op te bouwen en te normeren tot een kansverdeling. Zo'n normale kansdichtheid noemt men bij deze methode een Parzen window.

De Parzen schatter om f(x,y) te benaderen is

$$\hat{f}(x,y) = \frac{1}{(2\pi\sigma^2)^{(N+1)/2}} \frac{1}{T} \sum_{t=1}^{T} e^{-\frac{D_t^2}{2\sigma^2}} e^{-\frac{(y-Y^{(t)})^2}{2\sigma^2}} \qquad (2)$$

hier is $Y^{(t)} \in R^1$, waarbij $\{(X^{(t)}, Y^{(t)}) : t=1,\ldots T\}$ een leerset is

en $D_t(x) = |x - X^{(t)}| = \sqrt{\sum_{i=1}^{N} (x_i - X_i^{(t)})^2}$ de Euclidische afstand tussen het centrum $X^{(t)}$ en de vector x is en N is de dimensie van de input. σ is de spreidingsparameter die aan de volgende voorwaarden moet voldoen

$$\lim_{T\to\infty} \sigma(T) = 0 \quad \text{en} \quad \lim_{T\to\infty} T\sigma^N(T) = \infty$$

Aan deze voorwaarden is voldaan als $\sigma = \frac{S}{T^{E/N}}$,

waarbij T de omvang van de leerset is en N de dimensie van de input. Voor S en E moeten we zelf een keuze maken, met E tussen 0 en 1 inclusief 0. De σ is zo geconstrueerd dat bij verandering van het aantal inputvectoren de σ automatisch aangepast wordt.

Om onnodig veel Parzen windows te voorkomen, kunnen we dichtbij elkaar liggende inputvectoren clusteren. De afstand die bepaalt wanneer twee waarnemingen bij elkaar genomen moeten worden stellen we in met de Radius of influence. In de gebruikte software zit ook een vervalfunctie die afhangt van een tijdconstante τ. Ook een drempelwaarde kan ingesteld worden, waardoor een Parzen window, die na een bepaalde tijd nauwelijks of niet meer gebruikt wordt, voor een nieuwe Parzen window met een andere kern gebruikt zal worden.

Na uitwerking van formule (1) en (2) verkrijgt men voor elke component van Y een schatting

$$\hat{m}_j(x) = \frac{\sum_{t=1}^{T} Y_j^{(t)} e^{-\frac{D_t^2}{2\sigma^2}}}{\sum_{t=1}^{T} e^{-\frac{D_t^2}{2\sigma^2}}} \qquad (3)$$

, voor j=1,...,M,
waarbij M de dimensie van de output $Y=(Y_1,...,Y_M)$ is.
Voor de afleiding van formule (3) zie appendix A.

Na clusteren wordt bovenstaande formule

$$\hat{m}_j(x) = \frac{\sum_{k=1}^{K} A_j^{(k)} e^{-\frac{D_k^2}{2\sigma^2}}}{\sum_{k=1}^{K} B^{(k)} e^{-\frac{D_k^2}{2\sigma^2}}}$$

, voor j=1,...,M ,
waarbij $A_j^{(k)} := \Sigma Y_j^{(t)}$ over alle vectoren in cluster k.
$B^{(k)} := \Sigma 1$ over alle vectoren in cluster k.
D_k de Euclidische afstand tussen het centrum van de cluster en de variabele x. K is het aantal clusters.

Deze coëfficiënten zijn zo aangepast dat men eerdere data minder gewicht toekent dan recentere data. Dit werkt als volgt. Als de nieuwe vector ingedeeld wordt in cluster k dan berekenen we

$$A_j^{(k)}(t) = \frac{\tau - 1}{\tau} A_j^{(k)}(t-1) + \frac{1}{\tau} Y_j(t)$$

$$B^{(k)}(t) = \frac{\tau - 1}{\tau} B^{(k)}(t-1) + \frac{1}{\tau}$$

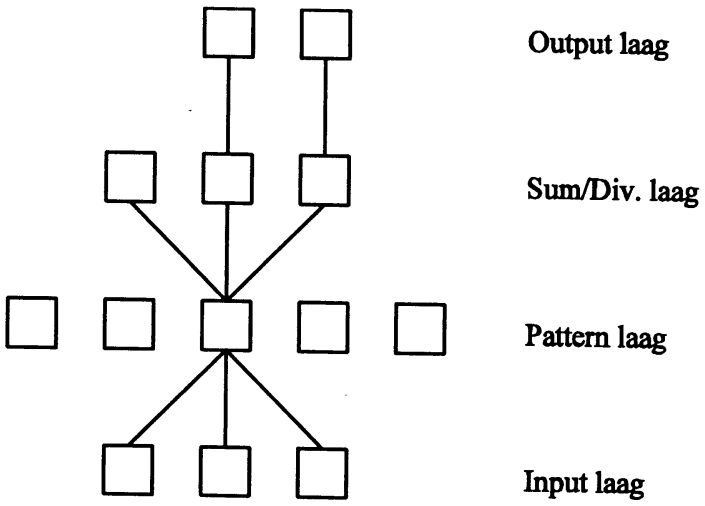
en als de nieuwe vector ingedeeld wordt in een andere cluster

$$A_j^{(k)}(t) = \frac{\tau - 1}{\tau} A_j^{(k)}(t-1)$$

$$B^{(k)}(t) = \frac{\tau - 1}{\tau} B^{(k)}(t-1)$$

Helaas is het niet mogelijk met τ onafhankelijk van T om de vervalfunctie uit te schakelen. Voor τ=1 telt alleen de laatste waarneming. Hoe groter τ wordt, hoe kleiner de bijdrage van de laatste waarneming wordt. Voor $\tau \to \infty$ telt de laatste waarneming nauwelijks meer mee. Voor τ gelijk aan het aantal data wat verwerkt is, levert dit het gewone gemiddelde van de gewogen input. De programmatuur suggereert dat τ onafhankelijk van de omvang T van de leerset gekozen kan worden. Bij vaste τ is de procedure echter niet consistent. Om wel consistentie te verkrijgen is het noodzakelijk om voor grotere omvang van de leerset een grotere τ te gebruiken. Voor het bewijs zie appendix B.

De architectuur



GRNN, alleen de verbindingen met een knooppunt in de Pattern laag zijn gegeven.

Een GRNN bestaat uit een Input-, een Pattern-, een Summation and Division- en een Outputlaag. De knooppunten van de Input-laag komen overeen met de afhankelijke variabelen en de knooppunten van de Outputlaag met de te voorspellen variabelen. De knooppunten van de Patternlaag komen overeen met de Parzen windows. De Summation and Divisionlaag heeft een knooppunt meer dan de Output laag.

De gewichten tussen de eerste twee lagen worden opgeslagen als centrum $X^{(k)}$ van de k-de kern. De somfunctie is de eerder genoemde D_k. De transferfunctie is $\frac{1}{\sigma^N} e^{-\frac{D_k^2}{2\sigma^2}}$.

De gewichten tussen de Pattern- en de Summation and Division laag zijn de eerder genoemde $A_j^{(k)}$ en $B^{(k)}$ coëfficiënten. De somfunctie is de noemer van formule (3) voor het eerste knooppunt uit de Sum/Div laag en de teller van de formule voor de resterende knooppunten uit de Sum/Div laag, hun aantal is gelijk aan het aantal componenten van de output. De transferfunctie van de Sum/Div laag is de teller gedeeld door de noemer. Dit geheel levert formule (3).

2.2.2 Nadaraya-Watson schatter

Het GRNN netwerk is in feite, op details na, een neuraal netwerk versie van een bekende niet-parametrische regressie schatter, de Nadaraya-Watson schatter.

Om niet-parametrische vergelijkingen te schatten kunnen we de kernschatters gebruiken (Härdle 1990, 1991). Deze schatters kunnen gebaseerd zijn op verschillende kernfuncties: een normale kansdichtheid zoals gebruikt is in de vorige paragraaf of een uniforme, een driehoek, een parabolische functie, de zogenaamde Epanechnikov kern, enz.

De Nadaraya-Watson schatter wordt gegeven door

$$\hat{m}_h(x) = \frac{n^{-1}\sum_{i=1}^{n} K_h(x-X_i)\,Y_i}{n^{-1}\sum_{j=1}^{n} K_h(x-X_j)}$$

,met $K_h(x)=h^{-1}(x/h)$ de kernfunctie en h een parameter voor de bandgrootte welke overeen komt met de σ voor de normale verdeling en n de grootte van de steekproef. Deze schatter is afgeleid uit de voorwaardelijke verwachting van de simultane verdeling van de input en output. Zie appendix A voor een analoge afleiding voor de Gausische kernschatter.

De Nadaraya-Watson schatter is een consistente schatter als $h\to 0$ en $nh\to\infty$.
De gemiddelde kwadratische fout in een vast punt x is

$$MSE[\hat{m}_h(x)] = E(\hat{m}_h(x)-m(x))^2$$
$$= \frac{1}{nh}\frac{\sigma^2(x)}{f(x)}\|K\|_2^2 + \frac{h^4}{4}\left(m''(x)+2\frac{m'(x)f'(x)}{f(x)}\right)^2\mu_2^2(K)$$
$$+o(nh^{-1})+o(h^4) \quad (h\to 0, nh\to\infty)$$

,waarbij $\sigma^2(x)$ de voorwaardelijke variantie, $\|K\|_2^2$ de integraal van K^2 en μ_2 de verwachting van $\int x^2K(x)dx$ is.
De kleinst mogelijke gemiddelde kwadratische fout is van de orde $O(n^{-4/5})$ als we $h=c*n^{-1/5}$ nemen, voor een constante c die afhangt van K en m.

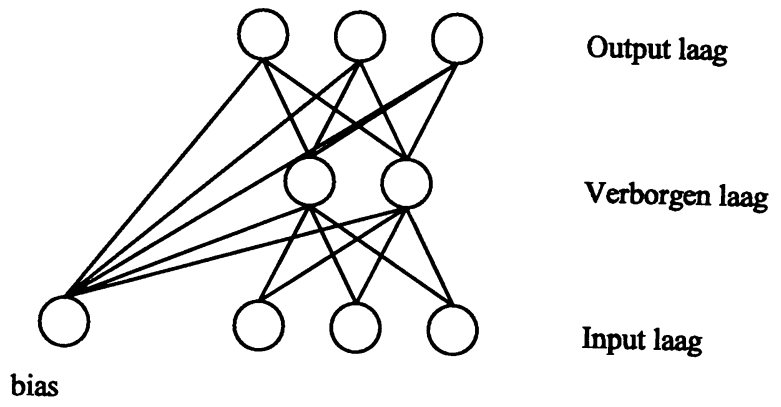
De WARPing methode (weighted averaging of rounded points) is een methode die waarden clustert (Härdle 1991). Met een kleiner aantal kernen wordt de regressie vergelijking benaderd. Deze methode is een soortgelijke methode als de clustermethode in de gebruikte software.

2.3 Back-propagation en niet-lineaire parametrische regressie

Een alternatieve methode gebruikt een feedforward neuraal netwerk met back-propagation en daarna volgt een beschrijving van niet-lineaire parametrische regressie.

2.3.1 Feedforward neuraal netwerk met back-propagation

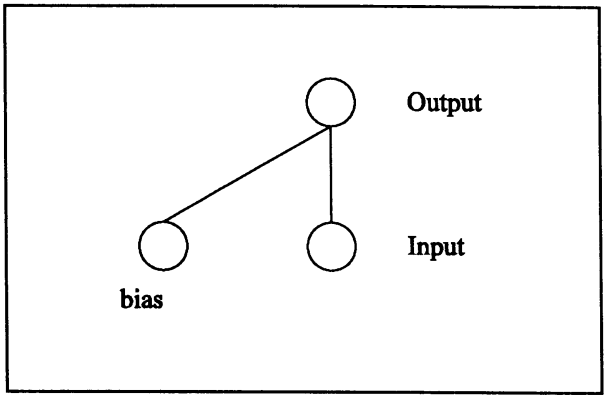
Back-propagation (Hertz 1991) is een methode om gewichten te corrigeren in een feedforward netwerk (netwerk opgebouwd uit lagen, waarin de data één richting op gaan). Er zijn verschillende regels voor het wijzigen van de gewichten. Wij gebruiken de delta regel. De delta regel is een iteratieve methode om de gewichten te bepalen, waardoor we de data meerdere malen moeten gebruiken om het netwerk te trainen.



Feedforward netwerk met een verborgen laag en met een bias.
De bias is met alle knooppunten verbonden,
uitgezonderd de input laag.

Een feedforward netwerk heeft een inputlaag, een outputlaag en eventueel één of meerdere verborgen lagen. In de inputlaag bevinden zich knooppunten die overeenkomen met de inputvector en een bias. De bias heeft als input altijd de waarde 1. De bias is met alle knooppunten verbonden, uitgezonderd de knooppunten van de inputlaag. Alle waarden van de input worden vermenigvuldigd met de gewichten. Deze produkten worden gesommeerd en als input voor de volgende laag gebruikt. Op waarden wordt een transferfunctie uitgerekend en het resultaat is als zodanig de output van deze knooppunten in deze laag. Dit gaat zo voort tot de outputlaag bereikt is. De bias heeft in elke laag een bijdrage. De bias heeft dan ook zoveel verbindingen als er knooppunten zijn min de dimensie van de inputvector.

De transferfunctie is meestal een continue niet-dalende functie, maar kan ook een stapfunctie zijn. Meestal wordt de logistische functie gebruikt. De input- en outputvectoren worden voor het trainen, d.w.z. het berekenen van de gewichten geschaald. Dit schalen hangt af van de transferfunctie.



Als we weten dat er een lineair verband is tussen de input en de output, hoeven we geen verborgen laag te gebruiken. Met de identieke transferfunctie zal het gewicht tussen de bias en de output de constante voorstellen en het gewicht tussen de input en de output zal de helling van het lineaire verband zijn. Omdat de data geschaald zijn zullen de gewichten uit het netwerk aangepast moeten worden willen we dezelfde coëfficiënten krijgen als met lineaire regressie.

De back-propagation methode kunnen we als volgt beschrijven: we gaan er van uit dat we beschikken over een testset bestaande uit N inputs en bijbehorende outputs.

Beschouw de kwadratische fout

$$E = \frac{1}{2}\sum_{k=1}^{N} (d_k - o_k)^2$$

met o_k de door het netwerk berekende output en d_k de gewenste output.

Back-propagation met de delta regel is een iteratieve methode om een stel gewichten te bepalen die E als functie van de gewichten minimaliseert. Daarbij wordt bij iedere stap het netwerk doorgerekend om de gewichten bij te stellen. Dit gaat als volgt.

Definieer $x_j^{[s]}=F(I_j^{[s]})$ als de output van het j^{de} knooppunt in laag s met, F de transferfunctie en

$I_j^{[s]}=\sum_{i=1}^{N_{s-1}} w_{ji}^{[s]} x_i^{[s-1]}$ de gewogen som van de inputs naar het j^{de} knooppunt in laag s, met $w_{ji}^{[s]}$ het gewicht tussen het i^{de} knooppunt in laag s-1 en het j^{de} knooppunt in laag s. N_{s-1} is het aantal knooppunten in laag s-1.
De output van een knooppunt is gelijk aan

$$x_j^{[s]}=F(\sum_{i=1}^{N_{s-1}} w_{ji}^{[s]}.x_i^{[s-1]})$$

Merk op dat E geschreven kan worden als functie van $I_j^{[s]}$, $j=1,\ldots,N_s$, de input van de knooppunten in laag s.
We definiëren een lokale fout voor het j^{de} knooppunt in laag s als

$$e_j^{[s]} := -\frac{\partial E}{\partial I_j^{[s]}}$$

Deze lokale fout is dus een maat voor de gevoeligheid van E voor veranderingen in $I_{j[s]}$.

De delta regel wordt gegeven door

$$\Delta w_{ji}^{[s]} = \eta e_j^{[s]} x_i^{[s-1]}$$

met η de learning rate. Voor de afleiding van de delta regel zie appendix C.

Voor de outputlaag, die aangegeven wordt met (o) van output geldt

$$\begin{aligned} e_k^{(o)} &= -\frac{\partial E}{\partial I_k^{(o)}} = -\frac{\partial E}{\partial o_k}.\frac{\partial o_k}{\partial I_k^{(o)}} \\ &= (d_k-o_k)F'(I_k^{(o)}) \end{aligned}$$

Voor de logistische transferfunctie, $F(x)=1/(1+e^{-x})$, geldt $F'(x)=F(x)(1-F(x))$.
We vinden dan wegens $x_k^{(o)}=F(I_k^{(o)})$

$$e_k^{(o)} = (d_k - o_k)\, x_k^{(o)} (1 - x_k^{(o)})$$
$$= (d_k - o_k)\, o_k (1 - o_k)$$

Learning rate en momentum term

De learning rate of leercoëfficiënt bepaalt de stapgrootte waarmee de gewichtvector zich langs het foutenoppervlak beweegt. Als deze groot is zullen de gewichten snel naar hun optimale waarden gaan. Een te grote learning rate kan als nadeel hebben dat de gewichten om de optimale waarden blijven schommelen. Nu kun je na een bepaald aantal data de learning rate verkleinen of de verandering in het gewicht laten afhangen van de vorige verandering. We voegen dan een deel van de vorige verandering toe aan de huidige verandering. De verandering van het gewicht wordt nu

$$\Delta w_{ji}^{[s]}(t) = \eta e_j^{[s]} x_i^{[s-1]} + \mu \Delta w_{ji}^{[s]}(t-1)$$

Als de bijdrage aan een gewicht van teken veranderd wordt de huidige verandering verkleind. Als het teken van de bijdrage hetzelfde blijft wordt de huidige verandering vergroot. Het deel van de vorige verandering wordt momentum term genoemd.

De standaardinstellingen in de software zijn in de gemaakte experimenten ook het beste. Deze zijn per laag verschillend nl.: .300; .250; .200; .150 voor de learning rates in resp. de eerste, de tweede, de derde en de vierde laag, tot zover ze aanwezig zijn en .4 voor de momentum term.

Opmerking 2.3.1

Als we voor de transferfunctie de sinus nemen en we nemen een verborgen laag, dan hebben we een algemenere versie van een Fourierreeks. Elke periodieke functie kan geschreven worden als een oneindige som van sinus termen

$$f(x) = a_0 + \sum_{n=1}^{\infty} c_n \sin(nx + d_n)$$

met a_0 als bias van de output, c_n de gewichten van de verborgen laag naar de output en de d_n de bias van de knooppunten in de verborgen laag. De variabele n bepaalt de gewichten tussen de input en de verborgen laag. In het neuraal netwerk kan deze variabele reëel worden, in plaats van een natuurlijk getal zoals in de Fourierreeks. Het aantal termen is gelijk aan het aantal knooppunten, deze kan nu niet oneindig zijn.

We kunnen ook de tangenshyperbolicus als transfer functie nemen. Dit komt overeen met de logistische functie op de schaling van de data na. Er geldt immers $\tanh(x)=1-2/(1+e^{2x})$.

Voor het corrigeren van gewichten kunnen we ook andere regels gebruiken. Delta-Bar-Delta regel is een regel waarbij de learning rate niet constant is zoals bij de delta regel. De learning rate neemt lineair toe of geometrisch af. Dit laatste is om de learning rate positief te houden. De Extended-Delta-Bar-Delta regel heeft een variabele momentumfactor. Met de delta regel was deze constant. Bij deze laatste regel zijn de learning rate- en momentumfactor begrensd.

Opmerking 2.3.2

Gebruiken we in het programma de logistische transferfunctie, dan wordt de data door het programma zo geschaald dat de input tussen -1 en 1 komt en de output tussen .2 en .8. Op dit stuk is de logistische functie vrijwel lineair en gaat door (0;.5) en heeft als afgeleide .25. Dus als de input uit één variabele bestaat dan kunnen we het lineaire stuk benaderen door y=.25*x+.5.

Als we met de logistische transferfunctie twee parameters (gewichten of coëfficiënten) verkrijgen met a als gewicht van de input en b als gewicht van de bias dan kunnen we deze omrekenen tot coëfficiënten die uit lineaire regressie ontstaan. De output is gelijk aan

$$y=F(ax+b)=\frac{1}{1+e^{-(ax+b)}}$$

De functie is het meest lineair in x=-b/a en heeft daar de waarde .5. De afgeleide is daar gelijk aan

$$F'(ax+b)=a*F'(x)=\frac{a}{4}$$

De benaderde regressie lijn wordt dan

$$y=\frac{a}{4}(x+\frac{b}{a})+\frac{1}{2}=\frac{a}{4}x+\frac{b}{4}+\frac{1}{2}$$

De coëfficiënten van de regressie lijn zijn dan a/4 en b/4+1/2. De waarden komen nog beter overeen als we de data nog dichter bij het 'lineaire' gedeelte van de logistische functie schalen. Bijvoorbeeld de input tussen -.25 en .25 en de output tussen .438 en .562. De afgeleide in de punten -.25 en .25 is .25 in hondersten nauwkeurig. Dit stemt nog beter overeen dan .20 in de punten -1 en 1 zoals de standaard instellingen.

Voor de tangenshyperbolicus als transferfunctie hebben we als output

$$Y=\tanh(ax+b)=\frac{e^{ax+b}-e^{-(ax+b)}}{e^{ax+b}+e^{-(ax+b)}}$$

De tanh heeft als afgeleide

$$\begin{aligned} G'(x) &= \frac{(e^{x}+e^{-x})^2-(e^{x}-e^{-x})^2}{(e^{x}+e^{-x})^2} \\ &= 1-G^2(x) \\ &= (1-G(x))\,(1+G(x)) \\ G'(ax+b) &= a\,G'(x) \\ &= a(1-G(x))\,(1+G(x)) \end{aligned}$$

In het punt x=-b/a is de afgeleide gelijk aan a en neemt de functie de waarde 0 aan. De benaderde regressie lijn wordt dan

$$Y=a\left(x+\frac{b}{a}\right)=ax+b$$

Hier zien we dat de coëfficiënten voor de tanh en lineaire regressie gelijk zijn als we de data dicht genoeg bij het lineaire gedeelte van de logistische functie schalen.

2.3.2 Niet-lineaire regressie

Het feedforward netwerk, zoals zojuist beschreven, kan gezien worden als een niet-lineaire parametrische regressie schattingsmethode, met back-propagation als een iteratieve methode om de parameters te schatten.

De logistische transferfunctie wordt gegeven door

$$h(x)=\frac{1}{1+e^{-x}}$$

Als we een bias als input meenemen en één input en één output en geen verborgen laag nemen, dan wordt de regressie vergelijking

$$f(x)=h(\theta_1 x+\theta_2)=\frac{1}{1+e^{-(\theta_1 x+\theta_2)}}$$

Deze vergelijking is intrinsiek lineair, d.w.z. de vergelijking is niet lineair met betrekking tot de parameters maar kan omgeschreven worden tot een lineaire vergelijking

$$\ln(f(x))=\theta_1 x+\theta_2$$

Hierdoor verandert wel de kansverdeling van de fouten in het regressiemodel.

Als we meerdere lagen nemen, dan hebben we een samenstelling van logistische functies (Denteneer 1993). In dat geval is de vergelijking niet intrinsiek lineair. De vergelijking wordt dan

$$f(x) = S(x) = S(x_1, \ldots, x_N)$$

waarbij S geschreven kan worden door

$$S = g^{(L)} \circ g^{(L-1)} \circ \ldots \circ g^{(1)}$$

met

$$g^{(l)}(x) = h(W^{(l)}(x)), \quad l=1, \ldots, L$$

$W^{(l)}$ zijn gewichtsmatrices en h is de logistische functie.

Aan de hand van de parameters willen we de gekwadrateerde fout minimaliseren. Deze fout is gelijk aan

$$E(x, \theta) = \sum_{i=1}^{N} (y_i - S(x, \theta))^2$$

Voor een netwerk met één input en één output wordt de fout

$$E(x, \theta_1, \theta_2) = \sum_{i=1}^{N} [y_i - f(x_i, \theta_1, \theta_2)]^2$$

met N de omvang van de leerset. We nemen aan dat de fouten $Y_i - f(x, \theta_1, \theta_2)$ ongecorreleerd zijn en variantie σ^2 hebben. Er zijn verschillende methoden om S te minimaliseren: de Gauss-Newton methode, gewijzigd door Hartley (1961), de steilste helling methode, de methode van Marquardt (1963) en het DUD (doesn't use derivatives) algoritme van Ralston en Jennrich (1978), welke geen gebruik maakt van afgeleiden. Back-propagation met de delta regel is een methode die wel gebruik maakt van afgeleiden.

Voor deze methoden hebben we beginwaarden nodig voor de gewichten. Er zijn verschillende methoden om deze te bepalen. Ratkowsky (1983) beschrijft een procedure. Lawton en Sylvestre (1971) hebben een methode ontworpen voor beginwaarden voor de gewichten die niet-lineair zijn. De keuze van de beginwaarden kan invloed hebben op het wel of niet bereiken van de kleinst mogelijke fout. Het neuraal netwerk genereert willekeurige beginwaarden uit een uniforme verdeling, maar dit kan ook zo ingesteld worden dat ze uit een normale verdeling komen.

Bij een toename van de leerset heeft de met $\sqrt{N}$ opgeblazen fout van de parametervector, d.w.z. $\sqrt{N}(\hat{\theta}_N-\theta)$, een asymptotisch normale verdeling met schatting van verwachting nul en variantie-covariantie matrix $\sigma^2 V$. Een consistente schatter van V is de pxp matrix

$$\hat{V}_N = \left[\frac{F^T(\hat{\theta}_N)\,F(\hat{\theta}_N)}{N}\right]^{-1}$$

waarbij F een Nxp matrix is en het element (u,i) gelijk is aan

$$\left.\frac{\partial f(x_u,\theta)}{\partial \theta_i}\right|_{\theta=\hat{\theta}_N}$$

met u=1,...,N en i=1,...,p.
Voor p=2 wordt het element (u,1)

$$\frac{\partial (1+e^{-\theta_1 x-\theta_2})^{-1}}{\partial \theta_1} = \frac{x\,e^{-\theta_1 x-\theta_2}}{(1+e^{-\theta_1 x-\theta_2})^2} = x\,f(x)\,(1-f(x))$$

en het element (u,2)

$$\frac{\partial (1+e^{-\theta_1 x-\theta_2})^{-1}}{\partial \theta_2} = \frac{e^{-\theta_1 x-\theta_2}}{(1+e^{-\theta_1 x-\theta_2})^2} = f(x)\,(1-f(x))$$

Een consistente schatter van σ^2 is

$$\hat{\sigma}_N^2 = \frac{S(\hat{\theta}_N)}{N-p}$$

waarbij S de som van de kwadraten van de residuen is. De herschaalde schatter $(N-p)\hat{\sigma}_N^2/\sigma^2$ heeft asymptotisch een chi-kwadraat verdeling met N-p vrijheidsgraden. Voor grote N heeft $\hat{\theta}_N$ een normale verdeling met verwachting θ en een variantie-covariantie matrix $\left[F^T(\theta)F(\theta)\right]^{-1}\sigma^2 N^{-1}$. Voor een uitgebreidere foutenanalyse zie Khuri (1987).

In theorie is dus de asymptotiek van de gewichten in het neuraal netwerk bekend. Het is echter onduidelijk, vanwege het grote aantal gewichten, voor welke omvang N de asymptotiek ook praktische betekenis heeft.

3.Resultaten

De eigenschappen van neurale netwerken zijn met twee soorten datasets onderzocht:

- een kunstmatige dataset die met behulp van een randomgenerator werd geconstrueerd en waarvan de onderliggende statistische samenhang dus volledig bekend is
- een dataset ontleend aan de meteorologische praktijk. Bij deze dataset is de statistische structuur dus niet a priori bekend

3.1 Gegenereerde data

De twee beschreven netwerken GRNN en back-propagation (BP) zijn met drie datasets onderzocht. Van het BP netwerk werden twee versies met en zonder verborgen laag onderzocht. De resultaten van de klassieke lineaire regressie zijn gebruikt als referentie. Voor het uitvoeren van de lineaire regressie werd gebruik gemaakt van het software pakket SPSS. Hiermee werden tevens de testdata gegenereerd. Voor de verschillende tests werden de volgende datasets gegenereerd:

- Experiment A

  De gegevens hebben betrekking op een lineair regressie probleem

  $Y = X + \epsilon$

  Met X normaal verdeeld met gemiddelde 0 en standaarddeviatie 100 en ϵ normaal verdeeld met gemiddelde 0 en standaarddeviatie 10. Er werden leersets gegenereerd met verschillende omvang namelijk 30, 100, 300 en 1000. De gevonden regressie vergelijkingen werden steeds getest op 5 testsets met 100 getallenparen (x,y).

- Experiment B

  Dit experiment is gelijk aan experiment A, maar in de leer- en testsets wordt een derde ongecorreleerde variabele als predictor aangeboden, die normaal verdeeld is met gemiddelde 0 en standaarddeviatie 10.

- Experiment C

  Dit experiment is ook bijna gelijk aan experiment A, maar de standaarddeviatie van ϵ is nu gelijk aan 40.

3.1.1 GRNN

Experiment A

De instellingen in de programmatuur worden zodanig gekozen voor een voorlopige leerset, dat de correlatie tussen de gewenste en de berekende output gemaximaliseerd wordt. Bij variërende instellingen verschillen de correlaties in duizendsten. Dan wordt het netwerk door een andere, maar even grote leerset getraind. In de appendix worden de resultaten van de testsets weergegeven. Het gemiddelde en de standaarddeviatie van de fout, de hieruit berekende wortel van de gemiddelde kwadratische fout en de correlatie van de gewenste en de
berekende output voor zowel GRNN als voor lineaire regressie met SPSS worden vermeld in (GRNN A) en hieruit afgeleid de wortel van de gemiddelde kwadratische fout (RMSE A).

In het eerste experiment (GRNN A en RMSE A) is in elk van de vier gevallen lineaire regressie met SPSS beter dan met GRNN. Het nadeel van GRNN is dat alleen tussen en vlakbij de data uit de leerset een soort regressie lijn wordt benaderd. Ver uit de buurt van de data van de leerset wordt de lijn constant. GRNN is dus slecht in extrapoleren. De vierde testset met waarden ver buiten het interval van de waarden uit de leerset heeft duidelijk een kleinere correlatie dan testsets met data tussen de data van de leerset.

Experiment B

Lineaire regressie met SPSS ondervindt nauwelijks hinder van de tweede input variabele, terwijl GRNN duidelijk een grotere fout maakt. Het is veel moeilijker om met normale verdelingen in de tweede dimensie een hellend vlak te construeren, dan zoals in het eerste voorbeeld een rechte lijn met normale verdelingen in de eerste dimensie. In grafiek (PLOT 1a) is te zien dat het oppervlak met 30 data niet zo goed een hellend vlak benaderd dan het oppervlak met 1000 data (PLOT 1b). Ook in dit experiment neemt de waarde van de vergelijking buiten het gebied van de leerset een constante aan. Ook hier is GRNN slecht in extrapoleren.

Experiment C

Net als in het eerste experiment is regressie met SPSS een stuk beter dan GRNN. Voor een voldoende grote leerset (1000 vectoren) is GRNN net zo goed als regressie met SPSS.

3.1.2 Back-propagation

Experiment A

In de eerste twee experimenten (BP_1A, BP_1B, RMSE A en B) gebruiken we geen verborgen lagen. Het eerste experiment bestaat uit een input en een output en een bias. Er zijn 20.000 iteraties geweest. Steeds na 10.000 iteraties zijn de gewichten nauwelijks veranderd. Het netwerk is dus steeds voldoende getraind. Uit de resultaten kunnen we afleiden dat voor dit regressie probleem back-propagation duidelijk beter is dan GRNN. Lineaire regressie en back-propagation ontlopen elkaar niet zo veel. Bij back-propagation worden de data door een stukje van de logistische functie benaderd. Ook hier geldt dat wanneer we testen met een testset met waarden buiten het interval van de waarden van de leerset de correlatie kleiner wordt. Dit komt doordat BP ook slecht is in extrapoleren. Voor kleinere waarden convergeert de vergelijking naar een minimum en voor grote waarden naar een maximum.

Experiment B

Net als het vorige experiment is back-propagation (BP_1B) beter dan GRNN en vrijwel net zo goed als lineaire regressie. Back-propagation wordt in tegenstelling tot GRNN niet door de extra input variabele gestoord.

3.1.3 Back-propagation met één knooppunt in een verborgen laag

Het derde en vierde experiment (BP_2A, BP_2B, RMSE A en B) zijn gelijk aan de eerste twee, maar uitgebreid met een verborgen laag met één knooppunt. Voor beide experimenten geldt dat met uitbreiding met een verborgen laag met één knooppunt de resultaten slechter worden.

Uit experimenten met meerdere knooppunten en meerdere lagen blijkt ook dat de resultaten slechter worden (Deze zijn niet vermeld). Waarschijnlijk komt dit door de grote hoeveelheid parameters die geschat moeten worden.

Opmerking 3.1.1

In appendix D worden resultaten vermeldt over gegenereerde data met een kwadratisch verband en interacties. Hierbij wordt ook een andere transferfunctie gebruikt en een ander initialisatie van de beginwaarden van de gewichten.

3.2 Praktijk data

De praktijkdata zijn ontleend aan een dataset welke was samengesteld voor een internationale zomercursus op het gebied van statistische interpretatie (Glahn 1991). In deze dataset wordt de dagelijkse maximumtemperatuur in de Bilt (Tx) gekoppeld aan twee meteorologische grootheden:

- de dikte van de luchtlaag tussen 100 hPa en 850 hPa (Th) boven de Bilt, deze dikte is evenredig met de gemiddelde temperatuur in die laag
- de oost-west component van de wind op 850 hPa (U) boven de Bilt, het effect van die windcomponent is in winter en zomer tegengesteld.

en aan twee niet meteorologische grootheden:

- de sinus en de cosinus van het dagnummer in het jaar, met deze grootheden kan het verschil in jaarlijkse gang van Tx en Th worden opgevangen.

Doordat het effect van de windcomponent in winter en zomer tegengesteld is (zie tabel SPSS A), worden meestal voor zomer en winter aparte vergelijkingen ontwikkeld, stratificatie op basis van seizoen. Men kan dit echter ook beschouwen als een interactie tussen cosinus- en sinustermen en de windcomponent en dan jaarvergelijkingen ontwikkelen.

3.2.1 SPSS

Er zijn zes experimenten. De data uit de zes jaar wordt in twee delen gesplitst, twee keer drie jaar, de ene drie jaar als leerset en de andere drie jaar als testset en omgekeerd. Hetzelfde met zomer en winter afzonderlijk. Uit de betacoëfficiënten (zie tabel SPSS A) (coëfficiënten gedeeld door de standaardafwijking van de bijbehorende variabele) blijkt dat de dikte de grootste invloed heeft. De windsnelheid heeft in de zomer een negatieve invloed op de temperatuur en in de winter een positieve. Over het hele jaar gezien heeft de wind snelheid een veel kleinere invloed. Met interacties tussen cosinus- en sinustermen en de windcomponent kunnen we dit verschil opvangen. In tabel SPSS A blijkt ook dat de dikte in de zomer een grotere invloed heeft op de maximale temperatuur dan in de winter.

3.2.2 GRNN

Uit veel experimenten met verschillende instellingen is de correlatie tussen de gewenste en de voorspelde output van GRNN kleiner dan die van SPSS. In tabel NN is één resultaat vermeld. Wanneer de correlatie van GRNN groter is blijkt uit de zeer kleine correlatie van de testset dat er sprake is van overfitting. Overfitting wil zeggen dat de vergelijking zo goed aan de leerset is aangepast, dat data in de testset verder van de vergelijking afliggen dan wanneer de vergelijking minder goed aan de leerset zou zijn aangepast.

3.2.3 Back-propagation

Back-propagation kan een redelijke benadering maken, deze evenaart lineaire regressie met SPSS. Netwerken met verborgen lagen vertonen na training een spoor van grote gewichten van de input, dikte naar de output, maximale temperatuur. Dit wijst op de grootte bijdrage die de dikte aan de output heeft. Hoe groter het gewicht, hoe groter de logistische functie, hoe groter de output van dit knooppunt enz... tot de output van de bovenste laag. Als we dit spoor van dikte naar de output uit de verborgen laag halen krijgen we een hogere correlatie tussen gewenste en de berekende output. Dit wijst op een lineair verband met de output.

3.3 Praktijk data met interacties

Het is gebleken dat de windsnelheid in de zomer een tegengestelde invloed heeft dan in de winter, net als de cosinus van het dagnummer. We nemen nu de windsnelheid maal de sinus en de windsnelheid maal de cosinus van het dagnummer mee als nieuwe predictor voor SPSS.

We kunnen met behulp van lineaire regressie zien of de interacties een bijdrage leveren aan de regressie vergelijking aan de berekende coëfficiënten. Met de methoden van neurale netwerken kunnen we dit niet direct zien. Om na te gaan hoe het neuraal netwerk omgaat met de invloed van de wind hebben we twee kunstmatige testsets geconstrueerd. Deze testsets bevatten 12 inputvectoren. In de eerste testset bevatten de inputvectoren de maandelijkse gemiddelden van de vier inputgrootheden dikte, windsnelheid en de sinus en cosinus van het dagnummer. In de eerste testset is echter van de windsnelheid een constant bedrag afgetrokken in de tweede testset een constante erbij opgeteld. Door nu de output van het netwerk voor ieder van de 12 vectoren van elkaar af te trekken vinden we een schatting van de lokale afgeleide naar de windsnelheid in iedere maand van het jaar. Als een netwerk het verschil in invloed van de windsnelheid in zomer en winter heeft gedetecteerd dan verwachten we een soort sinus functie als afgeleide.

3.3.1 SPSS

De correlaties (SPSS B) van de experimenten over het hele jaar zijn allemaal ongeveer met .004 gestegen. De standaard afwijking van de fout is met .1 verminderd. Dus lineaire regressie met interacties geeft een betere vergelijking tussen de in- en de output. Uit tabel SPSS B kunnen we ook de gemiddelde kwadratische fout berekenen

$$\mathbf{MSE} = \frac{1}{N}\sum_{i=1}^{N}(f_i - x_i)^2 = (\mathbf{ME})^2 + s_e^2$$

waarbij f_i de voorspelde output is, x_i de gewenste output is, ME het gemiddelde van de fouten is en s_e is de standaard afwijking van de fouten. Voor alle gevallen in de tabellen SPSS B en C hieronder de gemiddelde kwadratische fout.

MSE	eerste drie jaar leerset	laatste drie jaar leerset
SPSS zonder inter.	4.86	5.02
SPSS zomer	3.28	2.74
SPSS winter	5.38	6.34
SPSS na stratif.	4.33	4.54
SPSS met inter.	4.44	4.42
BP 2 lagen van 10	4.32	4.99
BP 3 lagen van 5	4.03	5.26
GRNN	7.65	11.40

De jaren in zomer en winter stratificeren of interactie meegeven, geeft een beter resultaat dan lineaire regressie zonder stratificatie of interacties. Nemen we de eerste drie jaar als leerset dan is stratificatie beter, maar als we de laatste drie jaar als leerset nemen is het meegeven van interacties beter.

De partiële afgeleide van de windsnelheid geeft een cosinus daar alleen de interactie van de windsnelheid en de cosinus een bijdrage geeft aan de regressie vergelijking (SPSS A).

3.3.2 Back-propagation met verborgen laag

Met de tanh als transferfunctie kwamen er betere resultaten. Met gebruik van drie verborgen lagen met in elke laag vijf knooppunten en na een training van 100.000 iteraties van de 931 data uit de eerste drie jaar waren de correlatie voor de leerset en de testset resp. .9703 en .9603, wat zelfs beter is dan de correlaties bij lineaire regressie met de twee eerder genoemde interacties als extra input. Als voor de leerset de laatste drie jaar en voor de testset de eerste drie jaar worden gebruikt, verder hetzelfde aantal knooppunten en lagen, dan vinden we de correlaties: .9636 en .9644, wat beter is dan lineaire regressie zonder extra input van eerder genoemde interacties, maar iets slechter dan de correlatie van de testset (SPSS B, NN). In bovenstaande tabel zien we dat back-propagation beter kan zijn dan lineaire regressie en zelfs beter dan lineaire regressie met stratificatie of met interacties. Dit is in het tweede experiment echter niet zo.

Als we gaan testen of back-propagation rekening houdt met de interacties blijkt na het plotten dat dit inderdaad zo is, aan de op cosinus lijkende grafieken (PLOT 2 a t/m d). Naast de interacties heeft back-propagation, in het eerste experiment, waarschijnlijk ook rekening gehouden met het feit dat de dikte in de zomer een grotere bijdrage heeft dan in de winter.

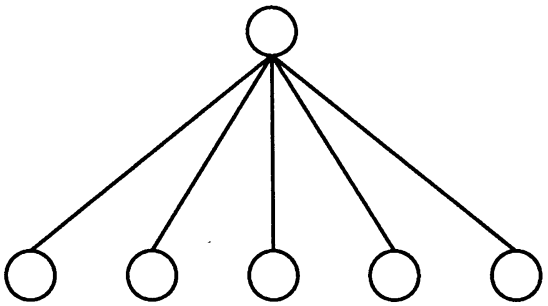
3.3.3 GRNN

GRNN heeft een veel lagere score dan lineaire regressie gezien de correlatie (tabel NN) en de gemiddelde kwadratische fout (tabel hierboven). De instellingen zijn hier: 400 knooppunten in de Pattern laag, τ=2000, radius of influence=.100, S=.500 en E=.500.

Net als bij BP gaan we ook hier kijken of GRNN de interacties herkent. De plots van de testsets vertonen ook een op een cosinus lijkende grafiek (PLOT 2 e t/m f).

4. Lerend Vermogen

In dit experiment onderzoeken we of we, na een netwerk getraind te hebben, met nieuwe data het netwerk kunnen verbeteren. Hiervoor nemen we een feedforward netwerk met back-propagation zonder verborgen laag. We beschouwen de gewichten tussen de inputvariabelen en de bias, en de output.



Feedforward netwerk met back-propagation, met de vier inputvariabelen en de bias, en de output, de maximale temperatuur

We gaan trainen met de praktijkdata van alleen de zomerperioden. Deze verdelen we in zes delen. Elk deel is een jaar. De data gaan we eerst schalen op dezelfde manier als het programma het zou doen. Dit doen we omdat wij de totale dataset kennen en het programma alleen de leerset en de testset ziet. De leerset laten we steeds variëren. Als we het programma zouden laten schalen zouden we steeds een andere schaling krijgen.

We maken eerst een netwerk met als gewicht voor de bias de gemiddelde temperatuur, en de andere gewichten nemen we gelijk aan nul. Vervolgens testen we dan met het zesde jaar. Daarna gaan we trainen met het eerste jaar maar met een beperkte aantal iteraties en testen dan weer met het zesde jaar. Dan nemen we het tweede jaar en trainen het getrainde netwerk weer met een beperkte aantal iteraties en testen dan weer met het zesde jaar. Dit proces herhalen we voor de eerste vijf jaar. Het aantal iteraties kiezen we zodanig dat alle data uit een jaar precies tien keer door het netwerk gegaan is. De gewichten na elke stap worden hieronder vermeld.

Dit jaarlijks aanpassen is een strategie die robuust is tegen mogelijke veranderingen. Als referentie nemen we een netwerk dat getraind is met een leerset bestaande uit de eerste vijf jaar. De gewichten zijn nauwelijks meer veranderd na 100.000 iteraties.

10X	bias	dikte	wind	cos	sin	cor.
begin	.4700	0	0	0	0	-.0001
jaar1	-.0186	.9507	-.6091	-.5127	-.4826	.8276
jaar2	-.1278	1.2489	-.6542	-.5649	-.2240	.8891
jaar3	-.4516	1.4731	-.5494	-.3315	-.2070	.9268
jaar4	-.5305	1.5303	-.4373	-.3975	-.1788	.9272
jaar5	-.5901	1.7374	-.4654	-.3642	-.0486	.9350
5jaar	-.5696	1.7067	-.4950	-.3763	-.1608	.9330

In een volgend experiment laten we het netwerk zo vaak itereren dan elk jaar 40 maal gebruikt wordt.

40X	bias	dikte	wind	cos	sin	cor.
begin	.4700	0	0	0	0	-.0001
jaar1	-.3082	1.6050	-.7009	-.4371	-.3671	.9061
jaar2	-.5479	1.7282	-.5329	-.4142	-.0966	.9316
jaar3	-.7947	1.8324	-.3808	-.2169	-.0690	.9403
jaar4	-.8443	1.8133	-.2613	-.2959	-.0673	.9375
jaar5	-.8544	1.9953	-.3146	-.2514	.0336	.9403
5jaar	-.5696	1.7067	-.4950	-.3763	-.1608	.9330

Uit het eerste experiment blijkt dat het netwerk naar de eindsituatie kunnen laten leren door de leerdata in delen aan te bieden. Uit het tweede experiment blijkt dat we het aantal keren dat we op een deel laten leren zorgvuldig moeten kiezen om overaanpassing te voorkomen. De correlatie tussen de gewenste en voorspelde output van het zesde jaar is in de eerste tabel monotoon stijgend en in de tweede tabel is deze

niet monotoon stijgend. Hieruit volgt ook dat bij een groot aantal iteraties het netwerk volledig aan het nieuwe jaar aanpast en dus de voorgaande jaren nauwelijks of niet meer meetellen. Het aantal iteraties bepaalt de mate waarin de voorgaande jaren meetellen.

5.Conclusies

In dit onderzoek zijn twee typen neurale netwerken bestudeerd als alternatief voor klassieke lineaire regressie. Dit betrof het General Regression Neural Network (GRNN) en een feedforward netwerk met back-propagation (BP). De netwerken werden getest op twee soorten datasets: de eerste dataset betrof een strikt lineair regressie probleem; de tweede dataset een dataset uit de meteorologische praktijk met een overwegend lineaire samenhang en een mogelijke (niet-lineaire) interactie tussen de predictoren. De resultaten van de experimenten leiden tot de volgende conclusies.

Strikt lineaire regressie

- GRNN is alleen toepasbaar indien grote datasets met onafhankelijke data beschikbaar zijn voor de training van het netwerk

- GRNN is niet in staat om de regressie buiten het interval van de leerset te extrapoleren

- GRNN is zeer gevoelig voor extra predictoren die niet gerelateerd aan de afhankelijke variabele, predictor selectie is onmogelijk

- BP kan ook met kleine datasets resultaten behalen die vergelijkbaar zijn met klassieke regressie

- Extrapolatie door BP buiten het leerset interval stuit op problemen

- predictoren die niet aan de afhankelijke variabele gerelateerd zijn, worden door BP genegeerd dus predictorselectie is mogelijk

- netwerkinstellingen zoals transferfunctie, beginwaarden van gewichten zijn belangrijk voor een optimale toepassing van BP

- het introduceren van verborgen lagen leidt tot slechtere resultaten bij BP netwerken bij de gebruikte data

- de ontwikkelinspanning (en benodigde rekentijd) is bij de aanpak van het neuraal netwerk aanzienlijk groter dan bij de klassieke regressie

Praktijkdata

- GRNN is niet in staat om het praktijkprobleem op een bevredigende manier op te lossen

- bij de klassieke regressie kunnen de interacties via stratificatie of introductie van interactietermen worden meegenomen

- door de introductie van verborgen lagen kan het BP netwerk de interacties in de samenhang detecteren en in de regressie opnemen

- een BP netwerk kan op eenvoudige wijze met de beschikbare dataset meegroeien en stapsgewijs leren

Een feedforward netwerk met back-propagation is een goed instrument om de samenhang in meteorologische datasets mee te bestuderen en vast te leggen. De ontwikkelinspanning voor een gegeven probleem dient men echter niet te onderschatten en het verdiend aanbeveling om tijdens de ontwikkelfase ook steeds klassieke analyses van de dataset uit te voeren.

Voor uitsluitend lineaire verbanden zal een neuraal netwerk het niet beter doen dan lineaire regressie. Voor niet-lineaire verbanden kan een neuraal netwerk het wel beter doen. Dit hangt sterk af van de verschillende keuzen die men kan maken.
Bij BP:
het aantal verborgen lagen
het aantal knooppunten per verborgen laag
de learning rate
de momentum term
de transferfunctie
initialisatie van de beginwaarden van de gewichten.

Bij GRNN:
het aantal knooppunten in de patternlaag
de spreidingsparameter
de parameter voor de vervalfunctie
de parameter om te clusteren.

Appendix A Afleiding van formules bij GRNN

Afleiding van formule (1)

De voorwaardelijke kansdichtheid van Y, gegeven X=x, wordt gegeven door

$$f_{Y|X=x}(y) = \frac{f(x,y)}{\int_{-\infty}^{\infty} f(x,y)\,dy}$$

waarbij f(x,y) de simultane dichtheid is van X en Y.
We berekenen nu de voorwaardelijke verwachting als volgt

$$m(x) = E(Y|X=x) = \int_{-\infty}^{\infty} y \frac{f(x,y)}{\int_{-\infty}^{\infty} f(x,y)\,dy}\,dy = \frac{\int_{-\infty}^{\infty} y\, f(x,y)\,dy}{\int_{-\infty}^{\infty} f(x,y)\,dy}$$

$\int f(x,y)dy$ is een constante dus kan deze buiten het integraalteken gehaald worden.
Hiermee hebben we formule (1) afgeleid.

Afleiding van formule (3)

Als we formule (2) invullen in formule (1) vinden we

$$m(x) = E(Y|X=x) = \frac{\sum_{t=1}^{T} e^{-\frac{D_t^2}{2\sigma^2}} \int_{-\infty}^{\infty} y\, e^{-\frac{(y-Y^{(t)})^2}{2\sigma^2}}\,dy}{\sum_{t=1}^{T} e^{-\frac{D_t^2}{2\sigma^2}} \int_{-\infty}^{\infty} e^{-\frac{(y-Y^{(t)})^2}{2\sigma^2}}\,dy}$$

Substitueer nu $z=(y-Y^{(t)})$.
De integraal in de teller wordt dan gelijk aan

$$\int_{-\infty}^{\infty} y\, e^{-\frac{(y-Y^{(t)})^2}{2\sigma^2}}\,dy = \int_{-\infty}^{\infty} (z+Y^{(t)})\, e^{-\frac{z^2}{2\sigma^2}}\,dz =$$

$$\left[-\sigma^2 e^{-\frac{z^2}{2\sigma^2}}\right]_{-\infty}^{\infty} + Y^{(t)} \int_{-\infty}^{\infty} e^{-\frac{z^2}{2\sigma^2}}\,dz = Y^{(t)} \int_{-\infty}^{\infty} e^{-\frac{z^2}{2\sigma^2}}\,dz$$

en de integraal in de noemer wordt dan gelijk aan

$$\int_{-\infty}^{\infty} e^{-\frac{(y-Y(t))^2}{2\sigma^2}}\, dy = \int_{-\infty}^{\infty} e^{-\frac{z^2}{2\sigma^2}}\, dz$$

Als we deze uitdrukkingen invullen en de integralen tegen elkaar wegstrepen dan houden we over

$$\hat{m}(x) = \frac{\sum_{t=1}^{T} Y(t)\, e^{-\frac{D_t^2}{2\sigma^2}}}{\sum_{t=1}^{T} e^{-\frac{D_t^2}{2\sigma^2}}}$$

Hiermee hebben we formule (3) afgeleid.

Appendix B Een niet-consistente schatter

De parameter τ is in het GRNN een constante. We schrijven τ om tot $\alpha=1/\tau$ en we laten α afhangen van n, de omvang van de leerset. GRNN met vervalfunctie komt overeen met het berekenen van gewogen gemiddelde van het type

$$\hat{f}_{nh}(x) = \sum_{i=1}^{n} p_{i,n} \frac{1}{h} K\left(\frac{x-X_i}{h}\right)$$

, met $p_{i,n}=(1-\alpha_n)(1-\alpha_{n-1})\ldots(1-\alpha_{i+1})\alpha_i$ voor $i=1,\ldots,n-1$ en $p_{n,n}=\alpha_n$. We veronderstellen dat $\alpha_1=1$.

Merk op dat $\sum_{i=1}^{n} p_{i,n}$ gelijk is aan 1.

In dit geval zijn de verwachting en de variantie van de kernfunctie gelijk aan

$$E\ \hat{f}_{nh}(x) = E\ \frac{1}{h}\ K\left(\frac{x-X_i}{h}\right)$$

$$var\ \hat{f}_{nh}(x) = \sum p_{k,n}^2\ var\ \frac{1}{h}\ K\left(\frac{x-X_i}{h}\right)$$

Analoog voeren we de gewogen Nadaraya-Watson schatter in als

$$\hat{m}_{nh}(x) = \frac{\sum_{i=1}^{n} p_{i,n} Y_i K\left(\frac{x-X_i}{h}\right)}{\sum_{i=1}^{n} p_{i,n} K\left(\frac{x-X_i}{h}\right)}$$

We nemen nu alle α_n gelijk aan een constante α, uitgezonderd $\alpha_1=1$, wat het geval is bij GRNN.

Nu kan de variantie van de teller met de laatste term afgeschat worden. Dan vinden we als ondergrens

$$\frac{\alpha^2}{h}\ var\ \left(Y_1 \frac{1}{\sqrt{h}} K\left(\frac{x-X_1}{h}\right)\right)$$

Aan gezien de laatste variantie naar een constante convergeert, convergeert de ondergrens naar oneindig ($h\to 0$). Hieruit volgt dat de teller geen consistente schatter is van $\int y\ f(x,y)dy$. Analoog kan worden aangetoond dat de noemer geen consistente schatter is van $f_x(x)$. Dit heeft tot gevolg dat voor α_n vast de gewogen Nadaraya-Watson schatter niet-consistent is.

Appendix C Technische details bij back-propagation

Een paar voorbereidingen

$$x_j^{[s]} = F(I_j^{[s]}) \quad , \quad \frac{\partial x_j^{[s]}}{\partial I_j^{[s]}} = F'(I_j^{[s]})$$

$$I_k^{[s+1]} = \sum_{l=1}^{N_l} w_{kl}^{[s+1]} x_l^{[s]} \quad , \quad \frac{\partial I_k^{[s+1]}}{\partial x_j^{[s]}} = w_{kj}^{[s+1]}$$

$$\sum_{k=1}^{N_k} \frac{\partial E}{\partial I_k^{[s+1]}} = -\sum_{k=1}^{N_k} e_k^{[s+1]}$$

Twee maal de kettingregel toepassen levert

$$\begin{aligned} e_j^{[s]} = -\frac{\partial E}{\partial I_j^{[s]}} &= -\sum_{k=1}^{N_k} \frac{\partial E}{\partial I_k^{[s+1]}} \cdot \frac{\partial I_k^{[s+1]}}{\partial x_j^{[s]}} \cdot \frac{\partial x_j^{[s]}}{\partial I_j^{[s]}} \\ &= \sum_{k=1}^{N_k} e_k^{[s+1]} \cdot w_{kj}^{[s+1]} \cdot F'(I_j^{[s]}) \\ &= F'(I_j^{[s]}) \sum_{k=1}^{N_k} e_k^{[s+1]} \cdot w_{kj}^{[s+1]} \end{aligned}$$

Als transferfunctie wordt meestal de logistische functie genomen, $F(z)=(1+e^{-z})^{-1}$. De afgeleide is uit te drukken in een functie van zichzelf $F'(z)=F(z)*(1-F(z))$.

De voorgaande formule wordt dan

$$e_j^{[s]} = x_j^{[s]} (1 - x_j^{[s]}) \cdot \sum_{k=1}^{N_k} e_k^{[s+1]} w_{kj}^{[s+1]}$$

De delta regel is als volgt,

$$\Delta w_{ji}^{[s]} = -\eta \frac{\partial E}{\partial w_{ji}^{[s]}}$$

, met η de learning rate.

Bovendien

$$\frac{\partial E}{\partial w_{ji}^{[s]}} = \frac{\partial E}{\partial I_j^{[s]}} \cdot \frac{\partial I_j^{[s]}}{\partial w_{ji}^{[s]}} = -e_j^{[s]} x_i^{[s-1]}$$

Combineren we de voorgaande twee, dan vinden we

$$\Delta w_{ji}^{[s]} = \eta \; e_j^{[s]} x_i^{[s-1]}$$

Hiermee hebben we de delta regel afgeleid.

Appendix D Extra experimenten

Kwadratisch verband

Als output nemen we het kwadraat van de input. De input is geschaald naar het interval [-1,1] en de output naar [.2,.8] voor de logistische functie en naar [-.8,.8] voor de tanh.

Bij back-propagation met de tanh als transferfunctie ging het al bij twee knooppunten in de verborgen laag vrij goed met een correlatie van .9961. De parabool werd benaderd door $f(x)=\tanh(.9725-1.3978*A(x)-1.1563*B(x))$, met $A(x)=\tanh(.9922+1.6763x)$ en $B(x)=\tanh(1.1888-1.9757x)$ Terug geschaald is $f(x)$ ongeveer gelijk aan x^2 op het gebied van de leerset. Met de logistische functie en de beginwaarden van de gewichten geïnitialiseerd uit de uniforme verdeling zijn de resultaten veel slechter. De correlatie werd .1745. Door de gewichten te initialiseren met waarden uit de normale verdeling kwamen er wel betere resultaten. De correlatie werd .9841.

Interacties

Als output is het produkt van twee inputvariabelen gebruikt. Ook hier doet de tanh het beter dan de logistische functie. Als we twee lagen met elk vijf knooppunten nemen en 50.000 data door voeren krijgen we een correlatie van .9988. Uit de gelijkenis tussen de contourplots van de leerset en de testset en de correlatie is de tanh voor dit probleem een geschikte transferfunctie. Hetzelfde netwerk met de logistische functie als transferfunctie geeft een correlatie van .1279. Maar als we de beginwaarden uit de normale verdeling halen, krijgen we een correlatie van .8749. Als we nogmaals 50.000 data doorvoeren wordt de correlatie .8990.

Net als bij het kwadratisch verband is het belangrijk om met geschikte beginwaarden voor de gewichten te starten als we de logistische functie als transferfunctie nemen. De situatie met de logistische functie en de beginwaarden uit de uniforme verdeling vertoonde steeds na 50.000 data een slechte correlatie resp. .1279 ; .1485 ; .1510 en .1516 na 200.000 data. Hier is sprake van een lokaal minimum van de fout. Hieruit en het geval met de beginwaarden uit de normale verdeling blijkt dat de keuze van de beginwaarden van de gewichten van groot belang kunnen zijn.

RMSE A (De wortel van de gemiddelde kwadratische fout)

30	SPSS	GRNN	BP_1	BP_2
test1	9.82	14.80	10.45	14.47
test2	10.69	13.30	11.64	14.86
test3	9.19	13.17	9.67	13.14
test4	11.38	18.95	13.51	20.03
test5	11.28	13.73	11.98	15.53

100	SPSS	GRNN	BP_1	BP_2
test1	10.00	11.44	10.37	14.98
test2	10.68	10.81	11.41	14.39
test3	9.22	9.91	9.41	12.44
test4	11.42	15.72	13.01	18.90
test5	11.35	11.50	11.75	14.97

300	SPSS	GRNN	BP_1	BP_2
test1	10.09	12.79	10.31	12.81
test2	10.51	11.29	10.89	13.49
test3	9.13	10.74	9.31	11.44
test4	11.46	15.09	13.16	17.78
test5	11.11	12.25	11.34	13.13

1000	SPSS	GRNN	BP_1	BP_2
test1	9.91	10.53	10.00	13.49
test2	10.56	10.91	10.90	13.48
test3	9.10	9.39	9.06	11.34
test4	11.39	11.85	12.28	16.51
test5	11.13	11.05	11.17	13.65

RMSE B

30	SPSS	GRNN	BP_1	BP_2
test1	10.45	26.85	11.12	14.66
test2	11.11	24.40	11.98	15.12
test3	9.28	26.83	9.71	13.30
test4	11.81	43.00	13.74	20.31
test5	11.15	23.47	11.98	15.72

100	SPSS	GRNN	BP_1	BP_2
test1	9.96	18.61	10.11	15.03
test2	10.70	16.55	10.90	14.50
test3	9.36	16.58	9.29	12.64
test4	11.42	24.08	12.32	19.01
test5	11.53	19.45	11.52	15.18

300	SPSS	GRNN	BP_1	BP_2
test1	10.09	14.78	10.47	13.21
test2	10.51	13.80	11.38	13.04
test3	9.13	14.88	9.42	11.36
test4	11.46	19.30	12.79	17.46
test5	11.12	12.80	11.82	13.83

1000	SPSS	GRNN	BP_1	BP_2
test1	9.88	13.30	10.21	13.56
test2	10.56	12.35	10.87	13.46
test3	9.17	12.62	9.39	11.65
test4	11.38	22.47	13.17	16.55
test5	11.22	13.03	11.35	14.17

RMSE C

30	SPSS	GRNN
test1	47.86	48.94
test2	41.13	60.82
test3	43.04	48.31
test4	45.07	48.14
test5	48.67	56.76

100	SPSS	GRNN
test1	44.10	44.97
test2	40.14	57.38
test3	40.79	46.50
test4	40.86	41.31
test5	43.19	47.75

300	SPSS	GRNN
test1	44.08	43.18
test2	39.25	56.42
test3	40.60	42.34
test4	40.94	41.58
test5	44.30	47.32

1000	SPSS	GRNN
test1	44.06	44.35
test2	39.41	40.45
test3	40.55	40.80
test4	40.88	41.23
test5	43.91	45.09

GRNN A

Instellingen: #PE=30 τ=1000 radius=.05 S=1 E=.5

30	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	-3.27	14.43	.9936	1.4388	9.7092	.99538
test2	-4.80	12.40	.9912	.0320	10.6894	.99265
test3	-4.24	12.47	.9942	.3780	9.1808	.99588
test4	-5.05	18.26	.9884	.2284	11.3779	.99363
test5	-3.04	13.39	.9916	1.3301	11.2053	.99322

Instellingen: #PE=100 τ=1000 radius=0.05 S=1 E=.5

100	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	.36	11.43	.9949	2.2703	9.7368	.99538
test2	-.76	10.78	.9925	.8971	10.6385	.99265
test3	-.34	9.90	.9954	1.2290	9.1423	.99588
test4	-.81	15.70	.9896	1.0815	11.3718	.99363
test5	-.34	11.49	.9928	2.1592	11.1525	.99322

Instellingen: #PE=100 τ=1000 radius=.05 S=1 E=.5

300	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	.58	12.78	.9946	1.5229	9.9786	.99538
test2	-.34	11.28	.9925	.2957	10.5054	.99265
test3	-.04	10.74	.9955	.5661	9.1115	.99588
test4	-.46	15.08	.9916	.4275	11.4541	.99363
test5	.01	12.25	.9927	1.4012	11.0236	.99322

Instellingen: #PE=100 τ=1000 radius=.01 S=.25 E=.5

1000	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	.84	10.50	.9948	1.2823	9.8260	.99538
test2	-.21	10.91	.9921	-.0223	10.5580	.99265
test3	-.07	9.39	.9957	.2807	9.1000	.99588
test4	.01	11.85	.9932	.1373	11.3885	.99363
test5	.53	11.04	.9932	1.1663	11.0715	.99322

GRNN B

Instellingen: #PE=30 τ=1000 radius=.100 S=.5 E=.5

30	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	-3.08	26.67	.9801	1.9690	10.2645	.99481
test2	-3.02	24.21	.9679	.5068	11.0971	.99221
test3	-2.72	26.69	.9738	.8848	9.2374	.99593
test4	-3.20	42.88	.9198	.7654	11.7824	.99323
test5	-5.03	22.92	.9776	2.1964	10.9324	.99353

Instellingen: #PE=50 τ=1000 radius=.200 S=.5 E=.5

100	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	4.50	18.06	.9894	2.3639	9.6726	.99544
test2	3.50	16.18	.9841	.9883	10.6529	.99264
test3	4.41	15.98	.9905	1.3189	9.2648	.99577
test4	3.40	23.84	.9796	1.1626	11.3583	.99365
test5	5.91	18.53	.9845	2.1670	11.3239	.99305

Instellingen: #PE=100 τ=1000 radius=.100 S=.375 E=.5

300	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	.07	14.78	.9932	1.5265	9.9764	.99538
test2	.12	13.80	.9894	.2994	10.5051	.99265
test3	-.19	14.88	.9911	.5697	9.1150	.99588
test4	.23	19.30	.9870	.4308	11.4534	.99363
test5	-.54	12.79	.9921	1.4020	11.0280	.99321

Instellingen: #PE=200 τ=1000 radius=.050 S=.375 E=.5

1000	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	1.07	13.26	.9947	1.3304	9.7877	.99542
test2	.15	12.35	.9912	.0250	10.5595	.99265
test3	.39	12.61	.9941	.3271	9.1615	.99583
test4	1.91	22.39	.9808	.1790	11.3777	.99365
test5	.62	13.02	.9918	1.1675	11.1551	.99313

GRNN C

Instellingen: #PE=30 τ=1000 radius=0 S=.5 E=.5

30	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	14.37	46.78	.9049	19.6656	43.6377	.91035
test2	18.85	57.82	.8710	12.4825	39.1940	.94173
test3	18.22	44.74	.9128	14.1015	40.6648	.92591
test4	24.06	41.70	.9350	19.8289	40.4755	.93755
test5	24.54	51.18	.9257	20.9935	43.9071	.93998

Instellingen: #PE=100 τ=1000 radius=.001 S=.5 E=.5

100	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	2.35	44.91	.9050	2.3408	44.0417	.91035
test2	-1.51	57.36	.8733	-4.2155	39.9153	.94173
test3	-5.85	46.13	.9045	-3.3512	40.6510	.92591
test4	2.80	41.21	.9375	2.8857	40.7611	.93755
test5	3.32	47.63	.9257	4.0026	43.0087	.93998

Instellingen: #PE=100 τ=500 radius=.005 S=.5 E=.5

300	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	4.89	42.90	.9137	6.1127	43.6537	.91035
test2	1.39	56.40	.8752	-.9505	39.2423	.94173
test3	-1.43	42.32	.9191	.5242	40.5956	.92591
test4	4.01	41.39	.9346	6.3490	40.4471	.93755
test5	6.52	46.87	.9305	7.5045	43.6573	.93998

Instellingen: #PE=100 τ=500 radius=.010 S=.750 E=.5

1000	GRNN			SPSS		
	gem.	std.dev	cor.	gem.	std.dev	cor.
test1	7.58	43.70	.9103	5.4030	43.7268	.91035
test2	.12	40.45	.9384	-1.4893	39.3852	.94173
test3	1.23	40.78	.9258	- .2204	40.5513	.92591
test4	6.53	40.71	.9372	5.7434	40.4747	.93755
test5	8.12	44.35	.9407	6.8858	43.3641	.93998

BP_1A
De gegenereerde data uit de vergelijking $Y=X+\epsilon$ met X $N(0,100^2)$ en ϵ $N(0,10^2)$ verdeeld.

30	gem.	std.dev	cor.
test1	2.62	10.12	.9950
test2	.53	11.63	.9920
test3	.53	9.66	.9956
test4	.14	13.51	.9911
test5	2.57	11.70	.9929

100	gem.	std.dev	cor.
test1	2.72	10.01	.9951
test2	.59	11.39	.9921
test3	.77	9.37	.9958
test4	.36	13.01	.9917
test5	2.60	11.46	.9930

300	gem.	std.dev	cor.
test1	1.88	10.14	.9951
test2	.36	10.88	.9922
test3	.25	9.31	.9957
test4	-.04	13.16	.9920
test5	1.83	11.19	.9930

1000	gem.	std.dev	cor.
test1	1.16	9.93	.9953
test2	-.67	10.88	.9923
test3	-.40	9.05	.9959
test4	-.74	12.26	.9928
test5	1.01	11.12	.9932

BP_1B

BP met input en output uit vorig experiment en een tweede input.

30	gem.	std.dev	cor.
test1	3.53	10.54	.9946
test2	1.45	11.89	.9916
test3	1.45	9.60	.9957
test4	1.09	13.70	.9908
test5	3.78	11.37	.9932

100	gem.	std.dev	cor.
test1	2.13	9.88	.9954
test2	.33	10.89	.9923
test3	.57	9.27	.9957
test4	.22	12.32	.9928
test5	1.85	11.37	.9929

300	gem.	std.dev	cor.
test1	2.63	10.13	.9950
test2	.26	11.38	.9919
test3	.75	9.39	.9957
test4	.25	12.79	.9920
test5	2.35	11.58	.9927

1000	gem.	std.dev	cor.
test1	1.41	10.11	.9952
test2	-.12	10.87	.9922
test3	-.22	9.39	.9956
test4	-.53	13.16	.9920
test5	1.29	11.28	.9929

BP_2A
De gegenereerde data uit de vergelijking Y=X+ϵ met X $N(0,100^2)$ en ϵ $N(0,10^2)$ verdeeld, met een knooppunt in verborgen laag.

30	gem.	std.dev	cor.
test1	5.35	13.45	.9912
test2	1.46	14.79	.9882
test3	1.36	13.07	.9920
test4	.58	20.03	.9803
test5	5.42	14.55	.9890

100	gem.	std.dev	cor.
test1	4.34	14.34	.9899
test2	-.01	14.39	.9875
test3	.91	12.41	.9925
test4	-.41	18.90	.9828
test5	3.92	14.45	.9884

300	gem.	std.dev	cor.
test1	-.06	12.81	.9921
test2	-3.65	12.99	.9894
test3	-3.03	11.03	.9940
test4	-4.04	17.32	.9859
test5	-.39	13.12	.9904

1000	gem.	std.dev	cor.
test1	1.95	13.35	.9913
test2	-1.86	13.35	.9888
test3	-.67	11.32	.9937
test4	-1.81	16.41	.9870
test5	1.43	13.57	.9897

BP_2B
BP een knooppunt in een verborgen laag. De input en output uit vorig experiment met een tweede input.

30	gem.	std.dev	cor.
test1	5.29	13.67	.9909
test2	1.22	15.07	.9877
test3	1.34	13.23	.9918
test4	.40	20.31	.9797
test5	5.44	14.75	.9885

100	gem.	std.dev	cor.
test1	4.03	14.48	.9897
test2	-.40	14.49	.9873
test3	.60	12.63	.9922
test4	-.78	18.99	.9826
test5	3.49	14.77	.9879

300	gem.	std.dev	cor.
test1	3.34	12.78	.9921
test2	-.27	13.04	.9893
test3	.34	11.35	.9936
test4	-.71	17.45	.9858
test5	2.82	13.54	.9898

1000	gem.	std.dev	cor.
test1	2.79	13.27	.9914
test2	-1.04	13.42	.9887
test3	.13	11.65	.9933
test4	-1.04	16.52	.9869
test5	2.08	14.02	.9891

SPSS A

zomer	eerste drie jaar leerset	laatste drie jaar leerset
dikte	.871118	.822267
windsnelheid	-.084457	-.134930
cosinus	-.164859	-.171981
sinus		
winter	**eerste drie jaar leerset**	**laatste drie jaar leerset**
dikte	.612518	.647016
windsnelheid	.208401	.216641
cosinus	-.360351	-.386200
sinus	-.102794	
hele jaar	**eerste drie jaar leerset**	**laatste drie jaar leerset**
dikte	.699319	.658135
windsnelheid	.037957	.025185
cosinus	-.333521	-.385259
sinus		.033006

SPSS B

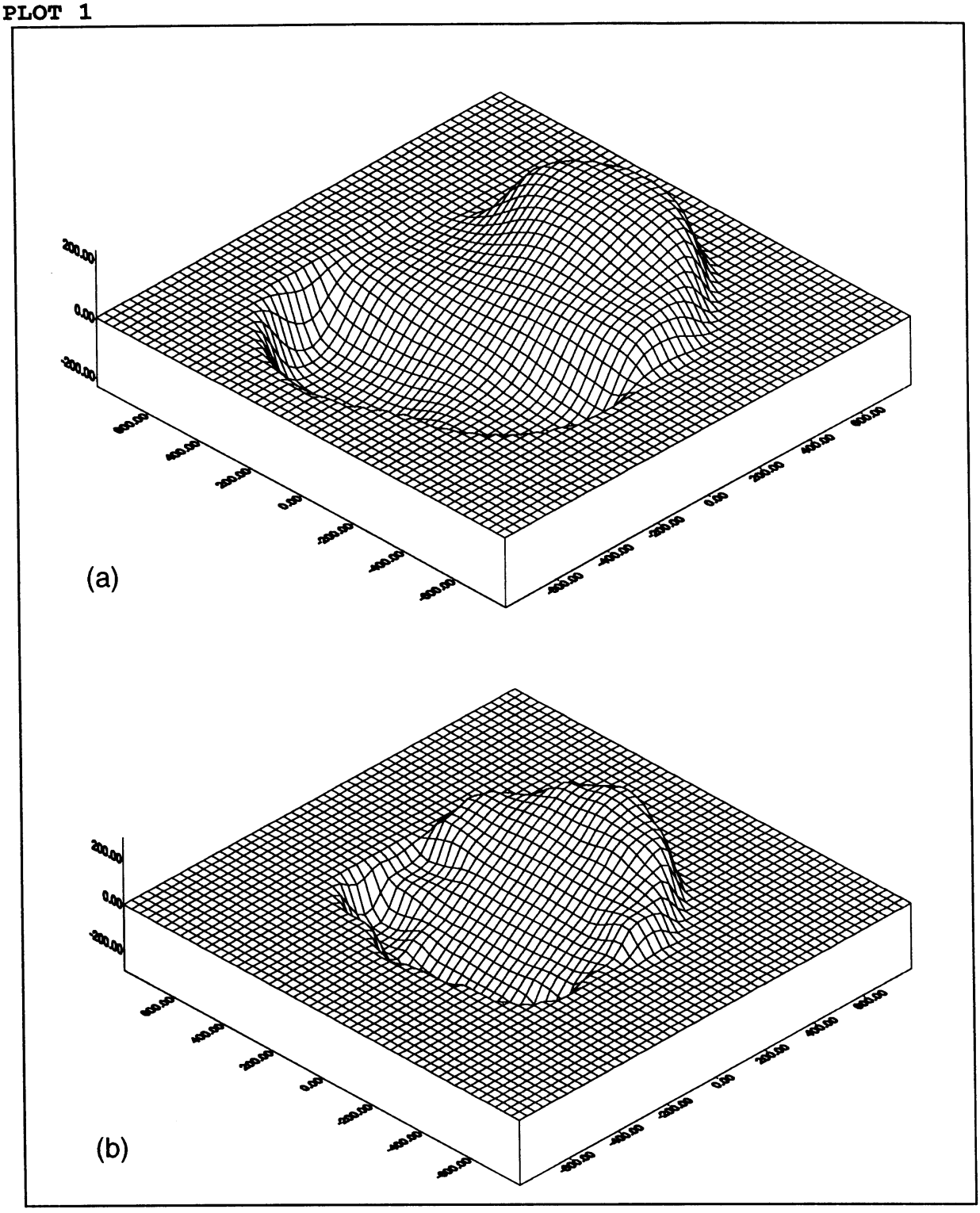
zomer					
eerste drie jaar leerset			laatste drie jaar leerset		
0	1.6294	.92936	0	1.7805	.91185
0.2320	1.7966	.91084	-.1784	1.6464	.92851
winter					
eerste drie jaar leerset			laatste drie jaar leerset		
0	2.2853	.91206	0	2.1216	.85904
0.8681	2.1507	.85691	-.8859	2.3568	.90836
hele jaar					
eerste drie jaar leerset			laatste drie jaar leerset		
0	2.1459	.96320	0	2.1414	.95023
0.4232	2.1630	.94956	-.4883	2.1876	.96219
hele jaar met interacties					
eerste drie jaar leerset			laatste drie jaar leerset		
0	2.0373	.96689	0	2.0483	.95456
0.4010	2.0696	.95386	-.3929	2.0662	.96610

NN

back-propagation twee lagen van tien knooppunten					
eerste drie jaar leerset			laatste drie jaar leerset		
.05	1.99	.9686	-.01	1.83	.9639
.53	2.01	.9571	-.67	2.13	.9666
back-propagation drie lagen van vijf knooppunten					
eerste drie jaar leerset			laatste drie jaar leerset		
.03	1.93	.9703	.04	1.84	.9636
.52	1.94	.9603	-.65	2.20	.9644
GRNN met 400 knooppunten in de pattern laag					
eerste drie jaar leerset			laatste drie jaar leerset		
-.13	2.35	.9583	-.05	2.46	.9383
.97	2.59	.9265	-1.36	3.09	.9427

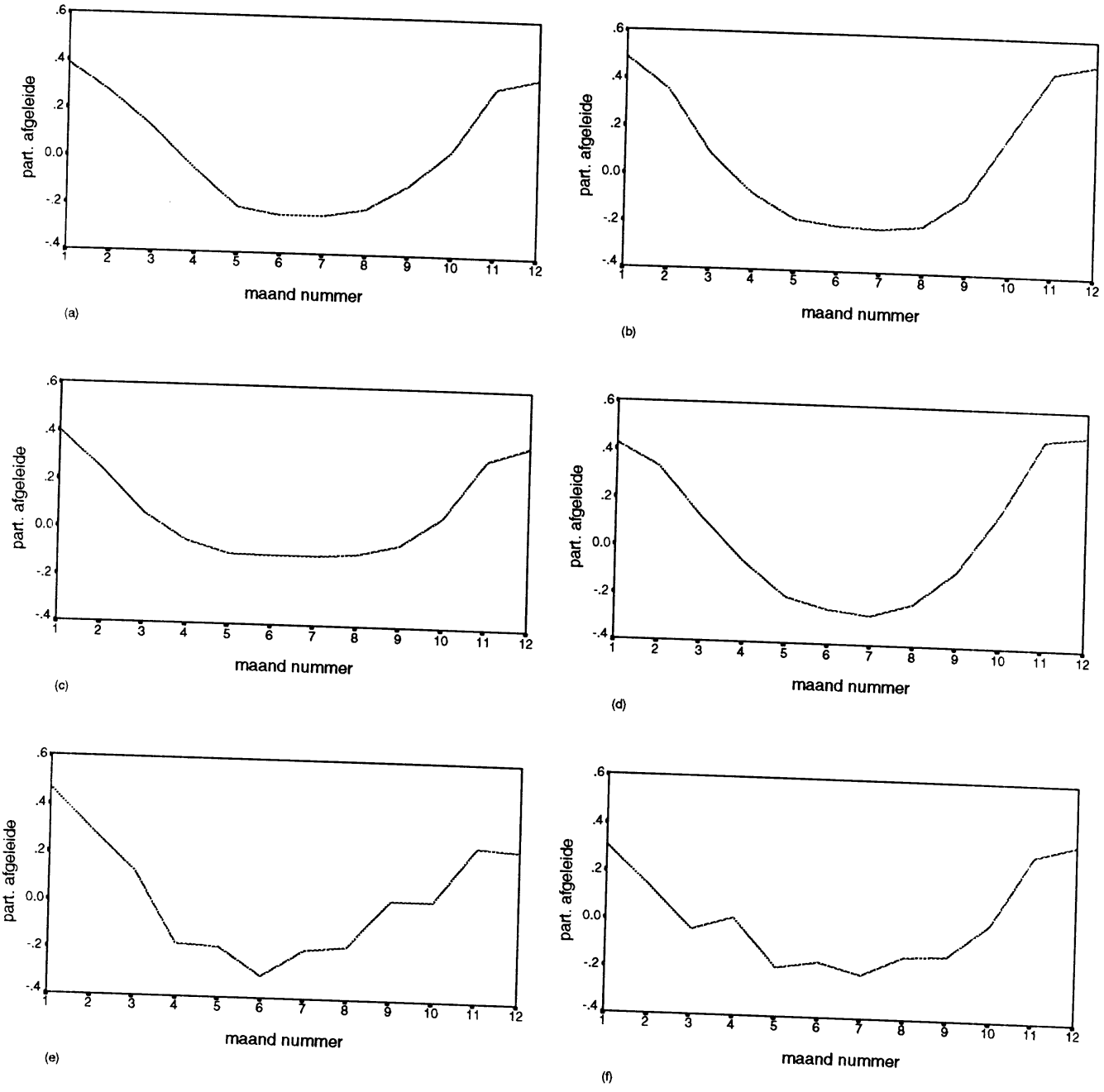
In bovenstaande tabellen zijn steeds het gemiddelde en de standaardafwijking van de fout en de correlatie van de berekende en gewenste output gegeven, voor zowel de data in de leerset als de data uit de testset.

PLOT 1



De bovenste grafiek is opgebouwd uit 30 normale verdelingen en heeft gebruik gemaakt van 30 data. De onderste grafiek is uit 200 normale verdelingen opgebouwd en heeft gebruik gemaakt van 1000 data. De onderste grafiek benadert hierdoor veel beter het hellend vlak dan de bovenste.

PLOT 2



(a) BP_2 leerset eerste drie jaar
(b) BP_2 leerset laatste drie jaar
(c) BP_3 leerset eerste drie jaar
(d) BP_3 leerset laatste drie jaar
(e) GRNN leerset eerste drie jaar
(f) GRNN leerset laatste drie jaar

BP_2 is back-propagation met twee verborgen lagen van tien knooppunten en BP_3 is back-propagation met drie verborgen lagen van vijf knooppunten.

Referenties

Cheng, B., D.M. Titterington, 1994. Neural networks: A review from a statistical perspective. Statistical Science., 1, 2-54.

Denteneer, D., 1993. Kwantitatieve Methoden nr.44, Vereniging van Statistiek, 21-34

Glahn, H.R., 1980. Plans for the development of a Local AFOS MOS Program (LAMP). Preprints Eighth Conference on Weather Forecasting and Analysis, Denver, Amer. Meteor. Soc., 302-305.

Glahn, H.R., D.A. Lowry, 1972. The use of Model Output Statistics (MOS) in objective weather forecasting. J. Appl. Meteor.,11, 1203-1211.

Glahn, H.R., A.H. Murphy, L.J.Wilson, J.S.Jensenius, Jr, 1991. Programme on short- and medium-range weather prediction research. WMO PSMP Report Series, 34

Härdle, W., 1990. Applied nonparametric regression, Cambridge Univ. Press, Cambridge.

Härdle, W., 1991. Smoothing Techniques, Springer, New York.

Hartley, H.O., 1961. The Modified Gauss-Newton Method for the Fitting of Nonlinear Regression Functions by Least Squares. Technometrics, 3, 269-280.

Hertz, J., A.Krogh, R.G.Palmer, 1991. Introduction to the theory of neural computation. Addison Wesley Publishing Company, Redwood city, USA.

Khuri A.I., J.A. Cornell, 1987. Response surfaces designs and analyses, Marcel Dekker, New York.

Kröse, B.J.A., P.P.vd Smagt, 1993. An introduction to Neural Networks, syllabus FWI, Univ. van Amsterdam.

Lawton, W.H., E.A. Sylvestre, 1971. Elimination of Linear Parameters in Nonlinear Regression. Technometrics, 13, 461-467.

Marquardt, D.W., 1963. An Algorithm for Least Squares Estimation of Nonlinear Parameters. J. Soc. Industrial and Applied Math., 11, 431-441.

Ralston, M.L., R.I. Jennrich, 1978. DUD, A Derivative-Free Algorithm for Nonlinear Least Squares. Technometrics, 20, 7-14.

Ratkowsky, D.A., 1983. Nonlinear Regression Modeling, Marcel Dekker, New York.

Specht, D.F., 1991. A General Regression Neural Network IEEE Transactions on Neural Networks, 6, 568-576,.

Specht, D.F., 1990. Probabilistic Neural Networks, Neural Networks, 109-118.

Handleidingen van de software:

Neural Computing, A Technology Handbook for Professional II/PLUS and NeuralWorks Explorer, 171-180.

Reference Guide, Software Reference for Professional II/PLUS and NeuralWorks Explorer.

Advanced Reference Guide, Software Reference for Professional II/PLUS and NeuralWorks Explorer.

Using NeuralWorks, A Tutorial for NeuralWorks Professional II/PLUS and NeuralWorks Explorer.