stichting
mathematisch
centrum

MC

AFDELING MATHEMATISCHE STATISTIEK SN 3/73 NOVEMBER

M. van GELDEREN

MULTIPELE REGRESSIE ANALYSE MET BEHULP VAN EEN REKENAUTOMAAT

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

AMS(MOS) subject classification scheme (1970): 62J05

Summary

Performing multiple regression analysis on an electronic computer can be very laborious when preparation of the input is complicated. This paper introduces a system, that is especially designed to have flexible and comprehensible model and input specifications.

The "model" specification is given as an ALGOL-like formula, which resembles the models given in common statistical literature quite closely.

An accompanying "input" specification provides the system with information about the arrangement of the observations in the data file.

The "data" file consists of a series of numbers in ALGOL free format.

A "run" command activates the system, while an "exit" command causes the system to stop.

An example of some of these ideas is:

```
"model"  y = alfa0 + alfa1 × x

"input"  5 × ([x], n, n × [y])

"data"    1   4   1.1   0.7   1.8   0.4
          3   5   3.0   1.4   4.9   4.4   4.5
          5   3   7.3   8.2   6.2
         10   4  12.0  13.1  12.6  13.2
         15   4  18.7  19.7  17.4  17.1

"run"

"exit"
```

Inhoud

Inleiding

Er bestaan vele programma's die op een of andere manier voor de te schatten parameters uit een lineair regressiemodel schattingen berekenen, doch de meeste daarvan gaan mank aan het feit dat ze niet te gebruiken zijn voor de leek. Soms is zelfs kennis van de machine vereist, hoewel dat sinds de hogere programmeertalen van de grond gekomen zijn niet vaak meer voorkomt; als leek is men evenwel niet met die talen op de hoogte. Vrijwel altijd echter eisen de programma's dat de getallen in een standaardvorm aangeboden worden, wat in de praktijk betekent dat bestaand getallenmateriaal omgevormd moet worden tot die ene speciale vorm, of dat de getallen precies in die vorm geponst moeten worden. Moeilijker nog en vaak ondoorzichtiger is de wijze waarop aan het programma meegedeeld wordt met welk model men aan het werk wil gaan. Voor bijvoorbeeld het programma voor multipele polynomiale regressie van het MC moet een codematrix van nullen en eenen geponst worden om de vorm van het regressiepolynoom aan te geven. Transformaties van een of meer variabelen zijn vrijwel nooit automatisch mogelijk en vereisen een voorloopprogramma.

Gelukkig begint de laatste tijd het besef steeds meer door te breken dat standaardprogramma's er zijn voor het plezier van de gebruiker, die meer baat heeft bij een overzichtelijke invoer dan bij een paar seconden winst in rekentijd. Dat is dan ook een van de hoofdredenen dat een nieuw programma ontwikkeld werd, waarbij ons van het begin af aan voor ogen stond dat het gebruikerscomfort zo groot mogelijk moest zijn: met een minimale en zo eenvoudig mogelijke invoerspecificatie een maximaal resultaat. Zo accepteert het systeem als modelbeschrijving een formule "zoals men die in de boeken tegenkomt", terwijl een invoerbeschrijving aangeeft bij welke variabele bepaalde (series) getallen uit de data horen. Dit geeft de gebruiker de mogelijkheid met bestaand getallenmateriaal te werken en meerdere regressieproblemen met eventuele transformaties te verwerken zonder steeds opnieuw de aangepaste getallen in te hoeven lezen.

Overigens moet men zich wel steeds goed blijven realiseren wat men doet: het model moet met naam en toenaam gegeven worden en de struktuur van de data moet precies bekend zijn voordat de invoerbeschrijving gegeven kan worden.

De identificatie van waarnemingen en bepaalde variabelen uit het model gebeurt nu "verbaal" en hangt niet meer af van codegetallen of van een bepaalde plaats van die waarnemingen tussen de andere getallen, terwijl in de uitvoer de resultaten geidentificeerd worden met de door de gebruiker in het model opgegeven namen. Dit draagt er volgens ons toe bij dat er wat minder onzin uitgerekend zal worden. Onzin die vaak voortkwam uit vergissingen in bijvoorbeeld codegetallen of kolomnummers.

De oorspronkelijke ideen voor deze invoerregeling zijn afkomstig van A.P.B.M. Vehmeyer, terwijl het artikel "ALGOL 60 translation for everybody" van F.E.J. Kruseman Aretz [6] de basis leverde voor de vertaler en het executiegedeelte uit het programma. Ook de ideen voor de structuur van het trommelgedeelte zijn van laatstgenoemde afkomstig. Een soortgelijk invoersysteem voor lineaire programmerings problemen van Jac.M. Anthonisse leverde eveneens vele aanknopingspunten op.

Hoofdstuk 1 Multipele regressie analyse

1 - 1 Het model

Bij een regressieprobleem zoekt men een verband tussen een stochastische variabele $\underline{y}$ (waarvan de meetuitkomsten of realisaties met meetfouten of andere vormen van storing zijn behept) enerzijds en een aantal variabelen $x_1, \ldots, x_p$ (die niet of nagenoeg niet met storingen zijn behept) anderzijds.

Dat verband wordt uitgedrukt met behulp van een mathematische formule, die het model wordt genoemd, bijvoorbeeld:

$$\underline{y} = b_0 + \sum_{i=1}^{p} b_i x_i + \underline{e} \qquad (1)$$

De uiteindelijke bedoeling is waarden van de variabele $\underline{y}$ te voorspellen met behulp van de modelformule en gegevens over de variabelen $x_1, \ldots, x_p$, waarvan de onderzoeker gelooft dat ze van belang zijn (hoewel hij dat natuurlijk niet zeker weet). Of de juiste voorspellende variabelen gekozen zijn, komt tot uitdrukking in hoe goed het model voorspelt.

Enig voorbehoud ten aanzien van voorspellingen gedaan met een bepaald model is echter op zijn plaats. Immers bij elke verzameling waarden voor de variabelen $x_1, \ldots, x_p$ geeft het model een voorspelling voor de variabele $\underline{y}$. Het is zeer wel mogelijk dat het model goed voorspelt voor die waarden voor $x_1, \ldots, x_p$ die binnen bepaalde grenzen vallen, maar voor waarden daarbuiten jammerlijk faalt. Een oorzaak hiervoor kan zijn dat een in feite niet lineair proces met een lineair model beschreven wordt, of dat een proces zich binnen bepaalde grenzen lineair voordoet, maar zich daarbuiten afwijkend gedraagt. Het is daarom essentieel dat de onderzoeker iets afweet van het proces waarvoor hij een model wil opstellen.

De variabelen $x_1, \ldots, x_p$ en de variabele $\underline{y}$ in vergelijking (1) kunnen ook (andere) getransformeerde variabelen representeren. Het kan namelijk zijn dat de onderzoeker reden heeft om aan te nemen (door zijn achtergrondinformatie betreffende het experiment) dat transformaties nodig zijn, bijvoorbeeld om:

1) te bereiken dat de storingen normaal verdeeld zijn,
2) een grotere homogeniteit van de variantie van de storingen te verkrijgen,
3) modellen die niet lineair zijn te "lineariseren" (als dat mogelijk is).

Vergelijking (1) kan dus ook geschreven worden als:

$$\underline{g} = g(\underline{y}) = b_0 + \sum_{i=1}^{p} b_i f_i(x_1, \ldots, x_n) + \underline{e} \qquad (2)$$

waarin $g, f_1, \ldots, f_p$ de transformaties,
$b_0, \ldots, b_p$ de te schatten parameters,
$\underline{y}$ de te verklaren (te voorspellen) variabele,
$x_1, \ldots, x_n$ de verklarende (voorspellende) variabelen,
en $\underline{e}$ de storing voorstelt.

Hoewel we ons kunnen beperken tot relatief eenvoudige transformaties, is de keuze van een transformatie door middel van de "trial and error" methode nogal tijdrovend en kostbaar. De moeilijkheid zit in het belang van de lokatieparameter van de transformatie. Het is niet ongewoon dat log(x) geen verbetering oplevert, maar dat log(c + x) zeer goed werkt voor een bepaalde keuze van $c \neq 0$. Daar dit waar is voor bijna alle transformaties van enig belang, moeten we eigenlijk voor iedere toegepaste transformatie op de data een niet lineair aanpassingsprobleem oplossen. Vaak echter wordt de vorm van de transformatie gesuggereerd door de onderzoeker, die beter op de hoogte is met de eigenaardigheden van het experiment.

1 – 2 Kleinste kwadraten

In feite bestaat regressie analyse uit het aanpassen van een hypervlak van de gewenste dimensie aan de data. Dat aanpassen gebeurt met de methode der kleinste kwadraten, hetgeen inhoudt dat de som van de kwadraten van de verschillen tussen de waargenomen waarden voor $\underline{y}$ en de (door het model) geschatte waarden voor de verwachting van $\underline{y}$ bij de betrokken waarden van $x_1, \ldots, x_p$, geminimaliseerd wordt. Die kwadratensom wordt ook wel de som van de kwadraten van de residuen genoemd.

In matrix notatie wordt het model geschreven als:

$$\underline{Y} = X\beta + \underline{e},$$

waarin $\underline{Y}$ een $(n \times 1)$ stochastische vector van waarnemingen,

X een $(n \times p)$ matrix van bekende (vaste) waarden,

β een $(p \times 1)$ vector van (onbekende) parameters,

en $\underline{e}$ een $(n \times 1)$ stochastische vector van storingen is.

Verondersteld wordt dat $E(\underline{e}) = 0$ en $\text{var}(\underline{e}) = I\sigma^2$ is, waarin I de eenheidsmatrix is. Het model kan dus ook geschreven worden als:

$$E(\underline{Y}) = X\beta.$$

Zij Y een realisatie van $\underline{Y}$. De som van de kwadraten van de residuen is dan:

$$\begin{aligned}(Y - X\beta)'(Y - X\beta) &= Y'Y - \beta'X'Y - Y'X\beta + \beta'X'X\beta \\ &= Y'Y - 2\beta'X'Y + \beta'X'X\beta \qquad (1)\end{aligned}$$

(immers $\beta'X'Y$ is een scalar en dus gelijk aan $Y'X\beta$)

De kleinste kwadraten schatting voor β is de waarde b, die als hij gesubstitueerd wordt in (1) de kwadratensom minimaliseert en wordt dus gegeven door de oplossing van het stelsel:

$$\left[\frac{\partial (Y - X\beta)'(Y - X\beta)}{\partial \beta}\right]_{\beta = b} = 0$$

Volgens de definitie van de afgeleide van een matrix en een elementaire stelling erover geldt:

$$\frac{\partial}{\partial \beta} \; -2Y'X\beta = -2X'Y$$

en $$\frac{\partial}{\partial \beta} \; \beta'X'X\beta = \beta'X'X + X'X\beta = 2X'X\beta.$$

Die minimaliserende waarde b voor β wordt dus gevonden uit:

$$-2X'Y + 2X'Xb = 0,$$

dus uit: $$X'Y = X'Xb \qquad (2)$$

Dit wordt het stelsel van de <u>normaalvergelijkingen</u> genoemd.

Is de rang van X gelijk aan p dan is X'X niet singulier en bestaat zijn inverse. In dit geval kan de oplossing van de normaalvergelijkingen geschreven worden als:

$$b = (X'X)^{-1} X'Y \qquad (3)$$

Merk op dat tenminste moet gelden dat $n \geq p$ is, opdat de rang van X inderdaad p kan zijn. Er moeten dus tenminste evenveel observaties gedaan worden als er parameters in het model zijn.

$$\underline{b} = (X'X)^{-1} X'\underline{Y} \qquad (4)$$

heet de kleinste kwadraten schatter voor β en heeft de volgende eigenschappen:

1) Het is een schatter die de som van de kwadraten van de residuen minimaliseert, onafhankelijk van eigenschappen van de verdeling van de storingen. Een veronderstelling dat de storingen normaal verdeeld zijn is slechts nodig om toetsen uit te voeren die gebaseerd zijn op normaliteit, zoals de t- en de F-toets, of voor het bepalen van betrouwbaarheidsintervallen gebaseerd op de t- en F-verdelingen.
2) Volgens de stelling van Gauss-Markov heeft $\underline{b}$ van alle schatters voor β die lineair zijn in de $\underline{y}_i$ de kleinste variantie, ook weer onafhankelijk van eigenschappen van de verdeling van de storingen.

3) Als de storingen o.o. en normaal verdeeld zijn,

(met $E(\underline{e}) = 0$ en $var(\underline{e}) = I\sigma^2$),

is $\underline{b}$ de meest aannemelijke schatter, immers de aannemelijkheidsfunctie is in dat geval:

$$\frac{1}{(\sigma\sqrt{2\pi})^n} \exp\left(\frac{(Y - X\beta)'(Y - X\beta)}{2\sigma^2}\right)$$

Dus voor een vaste waarde van σ is het maximaliseren van de aannemelijkheidsfunctie equivalent met minimaliseren van $(Y - X\beta)'(Y - X\beta)$.

De variantie-covariantie-matrix van $\underline{b}$ is:

$$var(\underline{b}) = var((X'X)^{-1}X'\underline{Y}) = var((X'X)^{-1}X'(X\beta + \underline{e}))$$

$$= var(\beta + (X'X)^{-1}X'\underline{e}) = (X'X)^{-1}X'I\sigma^2((X'X)^{-1}X')'$$

$$= (X'X)^{-1}X'X(X'X)^{-1}\sigma^2 = (X'X)^{-1}\sigma^2 .$$

De varianties staan op de hoofddiagonaal en de covarianties daarbuiten.

Een schatter voor σ^2 wordt gegeven door:

$$\underline{s}^2 = \frac{(\underline{Y} - X'\underline{b})'(\underline{Y} - X'\underline{b})}{n - p} = \frac{\underline{Y}'\underline{Y} - \underline{b}'X'\underline{Y}}{n - p}$$

(dit is de residuele kwadratensom, gedeeld door het bijbehorende aantal vrijheidsgraden).

Een veelvuldig gebruikte grootheid bij het beoordelen van regressie modellen is de multipele correlatie coefficient, die gedefinieerd wordt als dat deel van de totale kwadratensom dat door de regressie wordt bijgedragen, dus:

$$\underline{R}^2 = \frac{\underline{b}'X'\underline{Y}}{\underline{Y}'\underline{Y}}$$

Bij kleine steekproeven wordt vaak een korrektie toegepast op de $\underline{R}^2$ ten einde rekening te houden met het verlies van vrijheidsgraden:

$$\underline{R}'^2 = 1 - (1 - \underline{R}^2) \frac{n - 1}{n - p - 1} ,$$

waarin p het aantal te schatten parameters en n het aantal waarnemingen is. Toch moet men niet een al te groot vertrouwen in de schattingen voor de multipele korrelatiecoefficient stellen. Voor een psycholoog die een onderzoek doet naar gedragingen van mensen kan een model met $R = .3$ al bemoedigend zijn, terwijl in veel technische situaties waar nauwkeurige voorspellingen het doel zijn een R van .8 zeer kan teleurstellen omdat de residuele fout nog zo groot is, dat het model geen praktische waarde heeft.

In de volgende variantie-analyse-tabel worden de verschillende bijdragen tot de totale som van kwadraten van de waarnemingen van $\underline{y}$ in matrixnotatie aangegeven:

source	df	sum of squares	mean square	F-ratio
total (uncorrected)	n	$\underline{Y}'\underline{Y}$		
mean	1	$n\,\underline{\bar{y}}^2$		
regression | mean	$p - 1$	$\underline{b}'\underline{X}'\underline{Y} - n\,\underline{\bar{y}}^2$	$\dfrac{\underline{b}'\underline{X}'\underline{Y} - n\,\underline{\bar{y}}^2}{p - 1}$	$\dfrac{\underline{b}'\underline{X}'\underline{Y} - n\,\underline{\bar{y}}^2}{(p - 1)\,\underline{s}^2}$
residual	$n - p$	$\underline{Y}'\underline{Y} - \underline{b}'\underline{X}'\underline{Y}$	$\underline{s}^2$	

De kolom 'mean square' wordt verkregen door elke kwadratensom door het bijbehorend aantal vrijheidsgraden te delen. Zijn de storingen o.o. en normaal verdeeld, (met $E(\underline{e}) = 0$ en $var(\underline{e}) = I\sigma^2$), dan volgt het quotient van de 'mean square' van de regressie en de 'mean square' van het residu een F-verdeling met $p - 1$ en $n - p$ vrijheidsgraden.

Zijn er replicaties voor de te verklaren variabele, dan kan de som van kwadraten van de residuen als volgt gesplitst worden in een 'lack of fit' term en een 'pure error' term:

lack of fit	$k - p$	$\tilde{\underline{Y}}'\tilde{\underline{Y}} - \underline{b}'\underline{X}'\underline{Y}$	$\dfrac{\tilde{\underline{Y}}'\tilde{\underline{Y}} - \underline{b}'\underline{X}'\underline{Y}}{k - p}$	$\dfrac{n - k}{k - p}\ \dfrac{\tilde{\underline{Y}}'\tilde{\underline{Y}} - \underline{b}'\underline{X}'\underline{Y}}{\underline{Y}'\underline{Y} - \tilde{\underline{Y}}'\tilde{\underline{Y}}}$
pure error	$n - k$	$\underline{Y}'\underline{Y} - \tilde{\underline{Y}}'\tilde{\underline{Y}}$	$\dfrac{\underline{Y}'\underline{Y} - \tilde{\underline{Y}}'\tilde{\underline{Y}}}{n - k}$	

Hierin is $\tilde{\underline{Y}}' = (\tilde{\underline{y}}_1, \ldots, \tilde{\underline{y}}_k)$, met $\tilde{\underline{y}}_i = \dfrac{1}{\sqrt{m_i}} \sum_{j=1}^{m_i} \underline{y}_{ij}$,

k het aantal groepen replicaties,

en m_i het aantal replicaties per groep.

Hoofdstuk 2 Het in- en uitvoersysteem

2 - 0 De bedoeling van het invoersysteem is, dat de gebruiker op een zo eenvoudig en duidelijk mogelijke manier aan het programma kan meedelen wat hij wil. Hiertoe moet hij in een gebruikersprogramma een aantal beschrijvingen ('statements') verstrekken en hierin een aantal korrespondenties leggen, zodat bekend is welk regressiemodel bedoeld wordt en hoe de struktuur van de data er uit ziet. Elke beschrijving begint met een kodewoord en eindigt bij het eerstvolgende kodewoord. De kodewoorden zijn: "model", "input", "data", "run" en "exit" (inclusief de aanhalingstekens).

2 - 1 De modelbeschrijving

Om aan het programma kenbaar te maken tussen welke variabelen de statisticus een bepaalde relatie verwacht, moet hij een modelbeschrijving opgeven. Dat bestaat uit het kodewoord "model" gevolgd door een formule ('model statement') die veel lijkt op het model zoals dat gewoonlijk in de literatuur gegeven wordt. Voor de te verklaren en de verklarende variabelen en voor de parameters moeten onderscheidbare namen gegeven worden, dus bijvoorbeeld:

"model" y = alfa0 + alfa1 × x1 + alfa2 × x2 (1)

Iedere naam moet beginnen met een letter en mag willekeurig veel letters en/of cijfers bevatten, dus b.v. piet05jan is een korrekte naam. Aangezien de meeste randapparatuur van een rekenautomaat niet in staat is tot het behandelen van een index bij een naam (van een variabele of van een parameter) of tot het verwerken van Griekse letters, schrijven we b.v. alfa0 en alfa1 in plaats van α_0 en α_1. Namen (ofwel identifiers) hebben geen eigen betekenis, maar dienen voor de identificatie van variabelen en functies. Dezelfde identifier kan niet gebruikt worden om twee verschillende variabelen aan te geven.

Een modelformule bestaat uit de naam van een te verklaren variabele, gevolgd door een gelijkteken, gevolgd door de som van een aantal termen, die elk het product moeten zijn van de naam van een parameter en de naam van een verklarende variabele. Een uitzondering hierop wordt gemaakt voor een eventuele konstante term in het model, die alleen met de naam van de parameter gegeven wordt.
Korrekte modelbeschrijvingen zijn bijvoorbeeld:

die uit (1)

en: "model" y variabele = konstante term + parameter × x variabele

en: "model" depvar = const + beta1 × xvar1 + beta2 × xvar2.

2 - 1a Transformaties

Vrijwel alle transformaties die een gebruiker op zijn data zou willen uitvoeren, kunnen in deze opzet op geheel natuurlijke wijze tot uitdrukking gebracht worden, namelijk ook in formulevorm.
Wil men bijvoorbeeld als verklarende variabele de (natuurlijke) logarithme van de som van twee andere variabelen opnemen in het model, dan schrijft men: (als die andere variabelen b.v. xvar1 en xvar2 heten)

ln(xvar1 + xvar2)

Als operatoren zijn toegestaan: + , − , × , en / met hun normale betekenis.
Ook de volgende tien standaardfuncties mogen gebruikt worden:

abs(E), sign(E), sqrt(E), sin(E), cos(E), arctan(E),
ln(E), exp(E), entier(E) en arcsin(E).

Hierin is E een expressie die uiteraard ook weer een of meer functies mag bevatten.

Speciale operatoren zijn: // , de heling en xx , de machtsverheffing. De operatie // is slechts gedefinieerd voor gehele operanden en levert een geheel resultaat af dat als volgt gedefinieerd is:

$$a \,//\, b = \text{sign}(a / b) \times \text{entier}(\text{abs}(a / b))$$

Ook de verklarende variabele mag op deze manier getransformeerd worden, zodat de modelbeschrijving er in zijn meest algemene vorm uit ziet als in (2) op blz. 2. Slechts de storingsterm $\underline{e}$ is weggelaten (er hoeven dan ook geen streepjes onder de namen voor de te verklaren variabelen gezet te worden).

N.B. Het programma accepteert elke modelformule, mits die maar voldoet aan de formele definitie uit appendix 1 op blz. 40 , maar bekommert zich niet over de vraag of het getransformeerde model statistisch gezien wel zin heeft.

Enige voorbeelden van transformaties zijn:

"model" y = a0 + a1 × sqrt(x1 + x2) + a2 × sqrt(x3)

en: "model" arcsin(sqrt(yvar)) = a0 + a1 × x1 + a2 × x2 xx 2
+ a3 × x3 xx 3 + a4 × x4 xx 4

2 - 2 De invoerbeschrijving

Om aan te geven welke getallen of series getallen uit de data bij welke variabelen uit het model horen en welke getallen overgeslagen kunnen worden, wordt door het systeem een invoerbeschrijving verwacht. Dat bestaat uit het kodewoord "input" gevolgd door een beschrijving ('input statement') van waar de getallen in de data voorkomen. Bijvoorbeeld:

"input" 100 × (codenr, 8 × [yvar], [xvar1, xvar2])

Het idee is dat getallen uit de data door middel van de namen uit die beschrijving geidentificeerd worden met die namen, zodanig dat getallen die bij dezelfde naam horen in volgorde van binnenkomst als het ware "op een rijtje achter die naam worden gezet".

Dit kan gebeuren door een lijstje variabelen gescheiden door komma's (een 'variable list') op te geven. Aparte, series of blokken getallen kunnen nu behandeld worden door een variable list tussen vierkante haakjes of een input statement tussen ronde haakjes te zetten en er dan een repetitiefactor gevolgd door een maalteken voor te plaatsen. Als die factor 1 is mag hij weggelaten worden evenals het maal-teken, maar niet het daarop volgende ronde of vierkante haakjes paar. Wanneer als repetitiefactor een naam gebruikt wordt, moet die (vanzelfsprekend) eerst een waarde gekregen hebben, wat gebeurt door die naam van te voren een keer apart te noemen, gevolgd door een komma. Het eerst volgende getal uit de data wordt dan als waarde aan die naam toegekend (een soort initialisatie). Als die naam daarna nog een keer apart gebruikt wordt, wordt het eerst volgende getal uit de data vergeleken met de (dan dus bekende) waarde van die naam, waarna bij ongelijkheid een foutmelding volgt. Een dergelijke kontrole tegen verschoven inlezen kan men ook verkrijgen door een getal apart te gebruiken (dus een getal gevolgd door een komma). Gekontroleerd wordt dan of er in de data op de juiste plaats wel het zelfde getal staat.

De koppeling wordt nu tot stand gebracht door in een variable list dezelfde namen als in het model te gebruiken. Getallen uit de data die bij dergelijke namen horen zullen worden opgevat als waarnemingen voor die variabelen uit het model, terwijl getallen die horen bij namen uit de variable list, die niet in het model voorkomen, worden overgeslagen.

Vaak worden meerdere waarnemingen gedaan aan de te verklaren variabele bij dezelfde waarden van de verklarende variabelen. Om dit automatisch te kunnen verwerken, wordt geeist dat een variable list die alleen uit te verklaren variabelen bestaat, vooraf gegaan wordt door een repetitiefactor die dan het aantal replicaties aangeeft. Komen in een variable list zowel te verklaren als verklarende variabelen voor, dan wordt als repetitiefactor 1 genomen.

Geeft men bijvoorbeeld het volgende input statement op:

"input" 25 × (10 × [yvar], 2 × [loze], [xvar1, xvar2], 5 × [loze], -1)

dan wil dat zeggen dat men 25 series getallen heeft, die elk afgesloten worden door -1. Iedere serie bestaat uit 19 getallen, namelijk eerst 10 waarden voor de variabele yvar, dan 2 getallen die weggelezen worden naar de naam loze, dan een waarde voor de variabele xvar1, gevolgd door een waarde voor de variabele xvar2, en als laatste 5 getallen die weer weggelezen worden naar de naam loze. (Voor de naam loze had men overigens ook best een andere niet in het model voorkomende naam mogen nemen). Er wordt bovendien 25 maal gekontroleerd of er na die 19 getallen wel een -1 staat. Is de variabele yvar de te verklaren variabele in een bijbehorend model statement, dan zijn er dus 10 replicaties voor yvar bij elke waarneming voor xvar1 en xvar2.

2 - 3 De data

De data bestaat uit een ongestruktureerde rij getallen, voorafgegaan door het kodewoord "data". (de struktuur wordt erop aangebracht door het input statement) De rij eindigt bij het eerst volgende kodewoord.

Een rij symbolen vormt een getal, wanneer ze voldoet aan de definitie van <number> in appendix 1 - 4, waarbij bovendien spaties als lay-out-symbolen zijn toegestaan en wel:

a) na het teken van een getal, of na het symbool " # ", of na het teken van de exponent: willekeurig veel spaties,

b) na een cijfer of na een decimale punt: een enkele spatie.

Indien op een cijfer twee of meer spaties volgen, fungeert de tweede spatie als getalscheider. Overigens fungeren als getalscheider de volgende symbolen, indien ze direct op een cijfer volgen of daarvan door ten hoogste een spatie worden gescheiden:

a) alle symbolen die geen cijfer, "." of " # " zijn,

b) " # " na een cijfer van de exponent,

c) "." na een cijfer van de exponent of een cijfer van het breukgedeelte (een getal kan slechts een decimale punt bevatten).

Voorbeelden:

reeel getal	waarde
1.234	1.234
-0.5673#2	-56.73
0.02#-1	0.002
+#3	1000.0
-463.89#2	-46389.0

2 - 4 Het gebruikersprogramma

Er kunnen in een zogenaamd 'users program' meerdere 'jobs' achter elkaar aangeboden worden. Elke job wordt van zijn voorgaande gescheiden door het kodewoord "run" , terwijl het gehele users program afgesloten wordt door het kodewoord "exit". In de eerste job moet in een of andere volgorde het model, de input en de data gegeven worden. In elke volgende job mag een statement (zie: paragraaf 2 - 0), dat niet gewijzigd wordt, weggelaten worden.

Voor elke job of voor het kodewoord "exit" mag een tekst gegeven worden ter nadere identificatie van de output van een job of van de output van het gehele users program. Het verdient aanbeveling het gebruik van aanhalingstekens in die tekst te vermijden, in verband met een eventuele verwarring met de aanhef van de kodewoorden. Het programma begint een tekst te lezen vlak na het kodewoord "run" van de vorige job (bij de eerste job begint het programma met de eerste kolom van de eerste kaart).

2 - 4a De uitvoer van het programma

Nadat het kodewoord "run" gelezen is, wordt een (volgende) job in behandeling genomen. Als eerste wordt van de op dat moment bekende model en input statement teksten een afdruk gemaakt over de regeldrukker, eventueel vooraf gegaan door het begeleidende tekstje. Hierna wordt getracht de statements te vertalen, waarbij foutmeldingen worden gegeven voor fouten tegen de definitie of de syntaxis. Zijn bij vertaling geen fouten ontdekt dan wordt de job verder verwerkt: de (getransformeerde) data matrix wordt gevormd. Bij het dedecteren van een ongeoorloofde situatie wordt de executie na een foutmelding direct afgebroken. Er zal dan wel getracht worden een volgende job in behandeling te nemen, wat echter in het algemeen weinig zin zal hebben als het statement waarin de fout optrad niet veranderd wordt.

Is tot zover alles foutloos verlopen, dan wordt de (getransformeerde) data matrix aangeboden aan de regressie routine, die de volgende output verschaft:

1) per getransformeerde variabele het gemiddelde en de standaardafwijking
2) de korrelatiematrix
3) de multipele korrelatiecoefficient met een korrektie
4) de schattingen voor de regressiecoefficienten met de standaardafwijkingen
5) de variantie-analyse-tabel
6) als het aantal replicaties steeds 1 is een residuen-analyse.

In de output wordt iedere (eventueel getransformeerde) variabele aangegeven met zijn bijbehorende parameter. Dit is gedaan omdat het onduidelijk is of men de getransformeerde variabele: arcsin(sqrt(y + 25)) moet aangeven met de naam arcsin, met de naam sqrt, of misschien met de naam y zelf.

2 - 5 Foutmeldingen

Foutmeldingen zijn van de vorm: error, type: <foutnummer>

Betekenis van de foutnummers:

601 in een getal volgt op + , - of . geen cijfer.
602 in een getal volgt op # geen + , - of cijfer.

611 na left hand part volgt geen gelijk teken.
612) ontbreekt in model statement.
613 primary begint met ontoelaatbaar symbool.
614 te veel of te weinig parameters bij standaardfunctie.
615 parameterlijst niet afgesloten met) .
616 model statement niet correct.

621 ontoelaatbare identifier als control.
622) ontbreekt in input statement.
623] ontbreekt in input statement.
624 description begint met ontoelaatbaar symbool.
625 ontoelaatbare identifier in variable list.
626 variable begint met ontoelaatbaar symbool.
627 input statement niet correct.

640 geen gedefinieerde identifier rechts van het gelijk teken.
641 parameter onjuist gebruikt.
642 ongedefinieerde identifier links van het gelijk teken.
643 term niet van de vorm: par × factor of factor × par.
644 ongedefinieerde identifier in een term.
645 geen parameter in een term.

701 constantlist vol.
702 namelist vol.
703 orderlist vol.
704 stack vol.

811 control leest fout getal in de data.
812 in left data kloppen de replicaties op een regel niet.
813 in left data klopt het totale aantal in de kolommen niet.
814 in right data klopt het totale aantal in de kolommen niet.

Als het foutnummer begint met:

60, 61 of 62 wordt het gevolgd door het laatst gelezen symbool, of als dit een systemsymbol is door de eerste twee karakters daarvan, gevolgd door de eerste vier karakters van de laatst verwerkte identifier.
64 wordt het gevolgd door het nummer van de desbetreffende term.
70 wordt het gevolgd door de te grote bovengrens, waarna de job opnieuw start met grotere lijsten.
81 wordt het gevolgd door twee getallen die in de regel aangeven wat het programma denkt dat het moet zijn en wat het is, waarna een afdruk van de data wordt gegeven.

Enige voorbeelden

Op de volgende vier bladzijden worden een aantal voorbeelden uit de literatuur gegeven, die precies zo op kaarten zijn geponst en aan het programma zijn aangeboden. Op de daarop volgende bladzijden vindt men de resulterende output.

```
CHANNEL,63=60
CHANNEL,64=61
CHANNEL.END

************************************************************
*                                                          *
*  VOORBEELD  1  KOMT UIT:                                 *
*                                                          *
*  MEDISCHE STATISTIEK DEEL II (DE JONGE) BLZ. 472, 479    *
*                                                          *
************************************************************

"MODEL"  Y = A + B * X + C * (LN(X) / 2.30258 50930)


"INPUT"  5 * ([X], 10 * [Y])


"DATA"
 25  0.67  0.70  0.75  0.76  0.78  0.80  0.83  0.84  0.88  0.89
 50  0.88  0.92  0.93  0.96  0.98  1.00  1.01  1.03  1.06  1.07
 80  0.96  0.98  0.99  1.03  1.05  1.06  1.08  1.11  1.15  1.17
130  1.07  1.09  1.11  1.13  1.14  1.14  1.19  1.22  1.25  1.29
180  1.10  1.13  1.17  1.19  1.20  1.21  1.23  1.25  1.28  1.33
"RUN"
```

```
**************************************************************
*                                                            *
*  VOORBEELD  2  KOMT UIT:                                   *
*                                                            *
*  APPLIED REGRESSION ANALYSIS (DRAPER & SMITH) BLZ. 218, E.V.  *
*                                                            *
**************************************************************

"MODEL"  CHAMBER PRESSURE = BETA0 + BETA4 * STATIC FIRE + BETA34 * DROP SHOCK *
                            STATIC FIRE + BETA2 * VIBRATION

"INPUT"  24 * (UNIT NO, CYCLE TEMP, VIBRATION, DROP SHOCK, STATIC FIRE, CHAMBER
               PRESSURE)

"DATA"
 1   -75     0     0   -65    1.4
 2   175     0     0   150   26.3
 3     0   -75     0   150   26.5
 4     0   175     0   -65    5.8
 5     0   -75     0   150   23.4
 6     0   175     0   -65    7.4
 7     0     0   -65   150   29.4
 8     0     0   165   -65    9.7
 9     0     0     0   150   32.9
10   -75   -75     0   150   26.4
11   175   175     0   -65    8.4
12     0   -75   -65   150   28.8
13     0   175   165   -65   11.8
14   -75   -75   -65   150   28.4
15   175   175   165   -65   11.5
16     0   -75     0   150   26.5
17     0   175     0   -65    5.8
18     0     0   -65   -65    1.3
19     0     0   165   150   21.4
20     0   -75   -65   -65    0.4
21     0   175   165   150   22.9
22     0   -75   -65   150   26.4
23     0   175   165   -65   11.4
24     0     0     0   -65    3.7
"RUN"
```

```
**********************************************************************
*                                                                    *
*  VOORBEELD 3  KOMT UIT:                                            *
*                                                                    *
*  APPLIED REGRESSION ANALYSIS (DRAPER & SMITH) BLZ. 228, 339        *
*                                                                    *
**********************************************************************

"MODEL"  LN(MEAN SURFACE VOLUME)  =

                  LNALFA + BETA * LN(FEED RATE) + GAMMA * LN(WHEEL VELOCITY)

                  + DELTA * LN(FEED VISCOSITY)

"INPUT"  35 * [RUN NO, FEED RATE, WHEEL VELOCITY, FEED VISCOSITY,

               MEAN SURFACE VOLUME]

"DATA"
 1    0.0174     5300    0.108    25.4
 2    0.0630     5400    0.107    31.6
 3    0.0622     8300    0.107    25.7
 4    0.0118    10800    0.106    17.4
 5    0.1040     4600    0.102    38.2
 6    0.0118    11300    0.105    16.2
 7    0.0122     5800    0.105    26.5
 8    0.0122     8000    0.100    19.3
 9    0.0408    10000    0.106    22.3
10    0.0408     6600    0.105    26.4
11    0.0630     8700    0.104    25.8
12    0.0408     4400    0.104    32.2
13    0.0415     7600    0.106    25.1
14    0.1010     4800    0.106    39.7
15    0.0170     3100    0.106    35.6
16    0.0412     9300    0.105    23.5
17    0.0170     7700    0.098    22.1
18    0.0170     5300    0.099    26.5
19    0.1010     5700    0.098    39.7
20    0.0622     6200    0.102    31.5
21    0.0622     7700    0.102    26.9
22    0.0170    10200    0.100    18.1
23    0.0118     4800    0.102    28.4
24    0.0408     6600    0.102    27.3
25    0.0622     8300    0.102    25.8
26    0.0170     7700    0.102    23.1
27    0.0408     9000    0.613    23.4
28    0.0170    10100    0.619    18.1
29    0.0408     5300    0.671    30.9
30    0.0622     8000    0.624    25.7
31    0.1010     7300    0.613    29.0
32    0.0118     6400    0.328    22.0
33    0.0170     8000    0.341    18.8
34    0.0118     9700    1.845    17.9
35    0.0408     6300    1.940    28.4
"RUN"
```

```
      ************************************************************
      *                                                          *
      *  VOORBEELD  4  KOMT UIT:                                 *
      *                                                          *
      *  STATISTICAL ANALYSIS (AFIFI & AZEN) BLZ. 88, 93, E.V.   *
      *                                                          *
      ************************************************************

"MODEL"  Y  =  ALFA0  +  ALFA1 * X

"INPUT"  5 * ([X], N, N * [Y])

"DATA"
 1   4    1.1    0.7    1.8    0.4
 3   5    3.0    1.4    4.9    4.4    4.5
 5   3    7.3    8.2    6.2
10   4   12.0   13.1   12.6   13.2
15   4   18.7   19.7   17.4   17.1
"RUN"

      ***************************
      *                         *
      *  MARTEN VAN GELDEREN    *
      *                         *
      *  MATHEMATISCH CENTRUM   *
      *                         *
      ***************************
"EXIT"
```

```
**********************************************************
*                                                        *
*  VOORBEELD  1  KOMT UIT:                               *
*                                                        *
*  MEDISCHE STATISTIEK DEEL II (DE JONGE) BLZ. 472, 479  *
*                                                        *
**********************************************************
```

MODEL: Y = A + B * X + C * (LN(X) / 2.30258 50930)

INPUT: 5 * ([X], 10 * [Y])

CONTROL INFORMATION

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION
A	1.00000	.00000
B	93.00000	56.38787
C	1.87384	.30675
DEP.VAR.	1.04080	.16366

CORRELATION MATRIX

	A	B	C	DEP.VAR.
A	1.000000			
B	*	1.000000		
C	*	.962417	1.000000	
DEP.VAR.	*	.849838	.907742	1.000000

MULTIPLE CORRELATION COEFFICIENT

DIRECT ESTIMATE	.911959
WITH CORRECTION	.905920

REGRESSION ANALYSIS

PARAMETER	ESTIMATE	STANDARD DEVIATION
A	-.0899819319	.1641469135
B	-.0009361326	.0006395695
C	.6499168547	.1175694649

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO
TOTAL (UNCORRECTED)	50	55.47560		
MEAN	1	54.16323		
REGRESSION \ MEAN	2	1.09146	.54573	116.10596
RESIDUAL	47	.22091	.00470	
LACK OF FIT	2	.00501	.00251	.52234
PURE ERROR	45	.21590	.00480	

END OF REGRESSION ANALYSIS

```
**********************************************************************
*                                                                    *
*  VOORBEELD  2  KOMT UIT:                                           *
*                                                                    *
*  APPLIED REGRESSION ANALYSIS (DRAPER & SMITH) BLZ. 218, E.V.       *
*                                                                    *
**********************************************************************
```

MODEL: CHAMBER PRESSURE = BETA0 + BETA4 * STATIC FIRE + BETA34 * DROP SHOCK * STATIC FIRE + BETA2 * VIBRATION

INPUT: 24 * (UNIT NO, CYCLE TEMP, VIBRATION, DROP SHOCK, STATIC FIRE, CHAMBER PRESSURE)

CONTROL INFORMATION

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION
BETA0	1.00000	.00000
BETA4	42.50000	109.81209
BETA34	-997.91667	9503.49740
BETA2	33.33333	107.00129
DEP.VAR.	16.57917	10.85798

CORRELATION MATRIX

	BETA0	BETA4	BETA34	BETA2	DEP.VAR.
BETA0	1.000000				
BETA4	*	1.000000			
BETA34	*	.201315	1.000000		
BETA2	*	-.596668	.058825	1.000000	
DEP.VAR.	*	.943534	-.032554	-.463978	1.000000

MULTIPLE CORRELATION COEFFICIENT

DIRECT ESTIMATE	.987006
WITH CORRECTION	.984249

REGRESSION ANALYSIS

PARAMETER	ESTIMATE	STANDARD DEVIATION
BETA0	10.7131353245	.5067665166
BETA4	.1123394874	.0046333576
BETA34	-.0003139797	.0000430382
BETA2	.0233483268	.0046657977

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO
TOTAL (UNCORRECTED)	24	9308.45000		
MEAN	1	6596.85042		
REGRESSION \ MEAN	3	2641.59051	880.53017	251.54745
RESIDUAL	20	70.00907	3.50045	

RESIDUAL ANALYSIS

OBS.NO.	OBSERVATION	PREDICTION	RESIDUAL	STANDARDISED RESIDUAL
1	1.40000	3.41107	-2.01107	-1.07489
2	26.30000	27.56406	-1.26406	-.67562
3	26.50000	25.81293	.68707	.36723
4	5.80000	7.49703	-1.69703	-.90704
5	23.40000	25.81293	-2.41293	-1.28968
6	7.40000	7.49703	-.09703	-.05186
7	29.40000	30.62536	-1.22536	-.65494
8	9.70000	6.77850	2.92150	1.56151
9	32.90000	27.56406	5.33594	2.85200
10	26.40000	25.81293	.58707	.31378
11	8.40000	7.49703	.90297	.48263
12	28.80000	28.87424	-.07424	-.03968
13	11.80000	10.86446	.93554	.50004
14	28.40000	28.87424	-.47424	-.25347
15	11.50000	10.86446	.63554	.33969
16	26.50000	25.81293	.68707	.36723
17	5.80000	7.49703	-1.69703	-.90704
18	1.30000	2.08450	-.78450	-.41931
19	21.40000	19.79306	1.60694	.85889
20	.40000	.33338	.06662	.03561
21	22.90000	23.87902	-.97902	-.52327
22	26.40000	28.87424	-2.47424	-1.32245
23	11.40000	10.86446	.53554	.28624
24	3.70000	3.41107	.28893	.15443

SUM OF RESIDUALS -.00000

END OF REGRESSION ANALYSIS

```
**********************************************************************
*                                                                    *
*  VOORBEELD 3  KOMT UIT:                                            *
*                                                                    *
*  APPLIED REGRESSION ANALYSIS (DRAPER & SMITH) BLZ. 228, 339        *
*                                                                    *
**********************************************************************
```

```
MODEL:  LN(MEAN SURFACE VOLUME)  =
                  LNALFA + BETA * LN(FEED RATE) + GAMMA * LN(WHEEL VELOCITY)
                  + DELTA * LN(FEED VISCOSITY)
```

```
INPUT:  35 * [RUN NO, FEED RATE, WHEEL VELOCITY, FEED VISCOSITY,
              MEAN SURFACE VOLUME]
```

CONTROL INFORMATION
==========

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION
LNALFA	1.00000	.00000
BETA	-3.45447	.74805
GAMMA	8.84989	.29818
DELTA	-1.77847	.89959
DEP.VAR.	3.23975	.22850

CORRELATION MATRIX
==========

	LNALFA	BETA	GAMMA	DELTA	DEP.VAR.
LNALFA	1.000000				
BETA	*	1.000000			
GAMMA	*	-.181207	1.000000		
DELTA	*	-.036208	.180086	1.000000	
DEP.VAR.	*	.680447	-.811063	-.202936	1.000000

MULTIPLE CORRELATION COEFFICIENT
==========

DIRECT ESTIMATE	.977342
WITH CORRECTION	.974281

REGRESSION ANALYSIS

PARAMETER	ESTIMATE	STANDARD DEVIATION
LNALFA	8.5495323331	.2660238985
BETA	.1684244052	.0118081196
GAMMA	-.5371370141	.0300960773
DELTA	-.0144134670	.0098170458

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO
TOTAL (UNCORRECTED)	35	369.13353		
MEAN	1	367.35829		
REGRESSION \ MEAN	3	1.69570	.56523	220.30626
RESIDUAL	31	.07954	.00257	

RESIDUAL ANALYSIS

OBS.NO.	OBSERVATION	PREDICTION	RESIDUAL	STANDARDISED RESIDUAL
1	3.23475	3.29308	-.05833	-1.15155
2	3.45316	3.49988	-.04672	-.92237
3	3.24649	3.26683	-.02034	-.40160
4	2.85647	2.84558	.01089	.21498
5	3.64284	3.67112	-.02828	-.55834
6	2.90142	2.82141	.08001	1.57965
7	3.27714	3.18526	.09188	1.81394
8	2.96011	3.01323	-.05313	-1.04887
9	3.10459	3.09586	.00872	.17221
10	3.27336	3.31919	-.04583	-.90470
11	3.25037	3.24411	.00626	.12360
12	3.47197	3.53712	-.06515	-1.28624
13	3.22287	3.24614	-.02327	-.45943
14	3.68135	3.64277	.03858	.76164
15	3.57235	3.57750	-.00515	-.10175
16	3.15700	3.13662	.02038	.40228
17	3.09558	3.08993	.00564	.11143
18	3.27714	3.29042	-.01327	-.26199
19	3.68135	3.55160	.12975	2.56167
20	3.44999	3.42421	.02578	.50893
21	3.29213	3.30783	-.01570	-.30997
22	2.89591	2.93862	-.04270	-.84310
23	3.34639	3.28172	.06467	1.27680
24	3.30689	3.31961	-.01272	-.25113
25	3.25037	3.26752	-.01715	-.33855
26	3.13983	3.08936	.05048	.99651
27	3.15274	3.12716	.02557	.50488
28	2.89591	2.91763	-.02172	-.42885
29	3.43076	3.41028	.02047	.40419
30	3.24649	3.26119	-.01470	-.29023
31	3.36730	3.39228	-.02498	-.49322
32	3.09104	3.11036	-.01931	-.38129
33	2.93386	3.05143	-.11757	-2.32119
34	2.88480	2.86210	.02270	.44809
35	3.34639	3.30214	.04425	.87358
		SUM OF RESIDUALS	.00000	

END OF REGRESSION ANALYSIS

```
************************************************************
*                                                          *
*  VOORBEELD  4  KOMT UIT:                                 *
*                                                          *
*  STATISTICAL ANALYSIS (AFIFI & AZEN) BLZ, 88, 93, E.V.   *
*                                                          *
************************************************************
```

MODEL: Y = ALFA0 + ALFA1 * X

INPUT: 5 * ([X], N, N * [Y])

CONTROL INFORMATION
==========

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION
ALFA0	1.00000	.00000
ALFA1	6.70000	5.26258
DEP.VAR.	8.38500	6.54557

CORRELATION MATRIX
==========

	ALFA0	ALFA1	DEP.VAR.
ALFA0	1.000000		
ALFA1	*	1.000000	
DEP.VAR.	*	.987051	1.000000

MULTIPLE CORRELATION COEFFICIENT
==========

DIRECT ESTIMATE	.987051
WITH CORRECTION	.985516

REGRESSION ANALYSIS

PARAMETER	ESTIMATE	STANDARD DEVIATION
ALFA0	.1594830863	.3968072487
ALFA1	1.2276890916	.0470261690

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO
TOTAL (UNCORRECTED)	20	2220.21000		
MEAN	1	1406.16450		
REGRESSION \ MEAN	1	793.09943	793.09943	681.54980
RESIDUAL	18	20.94607	1.16367	
LACK OF FIT	3	4.25240	1.41747	1.27366
PURE ERROR	15	16.69367	1.11291	

END OF REGRESSION ANALYSIS

MARTEN VAN GELDEREN

MATHEMATISCH CENTRUM

Appendix 1 Formele Definitie

Voor de formele definitie van het systeem wordt een notatie gebruikt die bekend staat als de Backus Normal Form (kortweg: BNF), speciaal geintroduceerd voor de beschrijving van ALGOL. De notatie lijkt veel op de technieken voor het beschrijven van formele talen, die door logici en linguisten gebruikt worden. Volgens Peter Zilahy Ingerman werd een analoge techniek al ongeveer 400 tot 200 voor Christus door Panini gebruikt om grammatikale konstrukties in Sanskriet te beschrijven.

De BNF kan beschouwd worden als een metataal voor de formele beschrijving van het systeem. Hiertoe worden een aantal metasymbolen ingevoerd, n.l.

::= , | , < , > . Er worden hier slechts vier symbolen gegeven, de "," en "." behoren tot de metataal Nederlands, waarin we BNF beschrijven. We schrijven:

<identifier>::= <letter> | <identifier> <letter> | <identifier> <digit>.

De metasymbolen < en > worden gebruikt als scheiders om de naam van een te definieren klasse, de ::= kan gelezen worden als "wordt gedefinieerd als" of als "bevat", terwijl de | gelezen wordt als "of". Een <identifier> wordt zo dus gedefinieerd als een <letter> of als een <identifier> gevolgd door een <letter> of als een <identifier> gevolgd door een <digit>.

In huis tuin en keuken taal is een identifier tenminste een letter, gevolgd door nul of meer letters en/of cijfers.

Deze (recursieve) manier van definieren lijkt in het begin een beetje vreemd, aangezien een klasse gedefinieerd wordt in termen van zichzelf. Dit heeft slechts betekenis omdat steeds tenminste een alternatief in de definitie van een klasse die klassenaam niet bevat en zo als het ware de recursie aan de gang zet.

Grammatica

1 - 1 <letter>::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z

1 - 2 <digit>::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<adding operator>::= + | -
<multiplying operator>::= × | / | //

1 - 3 <model symbol>::= "model"
<input symbol>::= "input"
<data symbol>::= "data"
<run symbol>::= "run"
<exit symbol>::= "exit"

1 - 4 <number>::= <unsigned number> | <adding operator> <unsigned number>
<unsigned number>::= <decimal number> | <exponent part> |
<decimal number> <exponent part>
<decimal number>::= <unsigned integer> | <decimal fraction> |
<unsigned integer> <decimal fraction>
<exponent part>::= # <integer>
<decimal fraction>::= . <unsigned integer>
<integer>::= <unsigned integer> | <adding operator> <unsigned integer>
<unsigned integer>::= <digit> | <unsigned integer> <digit>

1 - 5 <identifier>::= <letter> | <identifier> <letter> | <identifier> <digit>

1 - 6 <data>::= <data symbol> <data list>
<data list>::= <number> | <data list> <number>

1 - 7 <input>::= <input symbol> <input statement>
<input statement>::= <part> | <input statement> , <part>
<part>::= <control> | <description> | <control> × <description>
<control>::= <number> | <identifier>
<description>::= (<input statement>) | [<variable list>]
<variable list>::= <variable> | <variable list> , <variable>
<variable>::= <identifier>

1 - 8 <model>::= <model symbol> <model statement>
<model statement>::= <left hand part> = <right hand part>
<left hand part>::= <simple arithexp>
<right hand part>::= <term> | + <term> | <right hand part> + <term>
<simple arithexp>::= <term> | <adding operator> <term> |
<simple arithexp> <adding operator> <term>
<term>::= <factor> | <term> <multiplying operator> <factor>
<factor>::= <primary> | <factor> ×× <primary>
<primary>::= <unsigned number> | <identifier> |
<function designator> | (<simple arithexp>)
<function designator>::= <identifier> | <identifier> (<parameter list>)
<parameter list>::= <simple arithexp> |
<parameter list> , <simple arithexp>

1 - 9 <users program>::= <job> <exit symbol> | <job> <users program>
<job>::= <statement> <run symbol> | <statement> <job>
<statement>::= <model> | <input> | <data>

Appendix 2 Beschrijving van enige procedures

In dit gedeelte wordt een meer technische beschrijving gegeven van een aantal procedures uit het programma. Elke naam die verwijst naar een identifier uit een procedure of uit het programma wordt tussen apostrofes gezet.

2 - 1 Het inleesgedeelte werkt met behulp van de procedures: 'readsymbol', 'readnumber', 'read1' en 'readlist'.

'readsymbol' leest een symbool en bergt de waarde van de interne representatie op in 'lastsymbol'. Text tussen quotes wordt, als die niet begint met: mo, in, da, ru of ex genegeerd, anders wordt dat stuk opgevat als systemsymbol. De koderingen voor de systemsymbols is als volgt:
100 × interne representatie eerste karakter + interne representatie tweede karakter.

'readnumber' leest met behulp van 'readsymbol' en 'read1' een <number> zoals dat gedefinieerd wordt in appendix 1 - 4 (met MC restricties genoemd in hoofdstuk 2 - 3) en kent de waarde van de getalscheider toe aan 'lastsymbol'. Als 'readnumber' een systemsymbol als getalscheider vindt, wordt de waarde van de datawijzer afgeleverd. Dit dient voor een later uit te voeren kontrole in 'check input'.

'readlist' leest een statement (model, input of data) in een buffer en zet die uitgaande van 'start' op de trommel. In 'size' komt de totale grootte van het statement terecht (het aantal symbolen of getallen).

De informatieve procedures 'digitlastsymbol', 'letterlastsymbol' en 'numberlastsymbol' komen met true of false terug op vragen of 'lastsymbol' al dan niet een cijfer of een letter is, of tot een getal behoort.

2 - 2 Het trommelgedeelte werkt met trajecten (pagina's) ter lengte van 'pagelength' op een magnetische trommel. Programma buffers zoals bijvoorbeeld 'symbollist' of een kolom van 'left data' of 'right data' passen precies in een pagina. In een paginatabel bestaat een zogeheten vrije keten met behulp waarvan we de eerst volgende vrije pagina op de trommel te pakken kunnen krijgen. Daarnaast bestaat zonodig voor iedere programma buffer een keten, die gemaakt wordt door bij het wegschrijven van een bufferbeeld naar een pagina op de trommel, in de paginatabel een verwijzing naar die pagina achter de reeds bestaande keten voor die buffer te hangen. Bij het teruglezen van trommelbeelden naar de programmabuffers kunnen de pagina's al of niet (afhankelijk van 'return') teruggegeven worden aan de vrije keten.

2 - 3 De functie procedure 'nextlistsymbol' levert aan de procedure naam het eerst volgende symbool van het model of input statement af, terwijl 'nextnumber' het eerst volgende getal uit de data list aflevert.

'nextsymbol' is de procedure met behulp waarvan de vertaler de te verwerken text leest. Hierbij wordt van twee opeenvolgende maal-tekens een tot-de-macht-teken en van twee opeenvolgende deel-tekens een helings-teken gemaakt, bovendien worden eventuele spaties geskipt.

'unsigned number' leest een getal, zoekt het op in de konstantenlijst en kent aan zichzelf een adres toe in die lijst waar de waarde van het getal gevonden kan worden.

'identifier' leest een naam, (pakt vier karakters in een machinewoord) zoekt hem op in de naamlijst en komt terug met een adres dat in de naamlijst een verwijzing naar de eerste vier karakters van de naam oplevert en in de stapel de waarde van de naam.

De door de vertaler gegenereerde macro opdrachten komen via 'store' in een opdrachtenlijst terecht.

2 - 4 Bij het ontwerpen van de vertaler is er van uitgegaan dat er een (geprogrammeerd pseudo) register 'F' en een stuk adresseerbaar geheugen genaamd 'stack' aanwezig is. 'F' is het register voor floating point arithmetiek, maar kan ook gehele getallen behandelen. De kop van de stapel werkt volgens het "last in first out" principe.

In het model of input statement voorkomende namen krijgen via 'identifier' een adres in de stapel toegewezen, waar tijdens vertaling het type van de naam wordt bijgehouden en tijdens executie de waarde. De kop van de stapel wordt gebruikt voor het onthouden van tussenresultaten bij het evalueren van expressies en voor het onthouden van repetitiefactoren. Alle binaire arithmetische operaties vinden plaats met de kop van de stapel als eerste operand en 'F' als tweede, het resultaat wordt afgeleverd in 'F' en als neveneffect wordt de stapelwijzer met 1 verlaagd.

Het fundamentele idee achter de vertaalprocedures is dat het eerste symbool van de door een procedure te vertalen syntactische eenheid al gelezen is (waarbij zijn waarde aan 'lastsymbol' is toegekend), terwijl de procedure zijn taak als geeindigd beschouwt bij het lezen van het eerste symbool dat syntactisch niet meer tot die eenheid kan behoren. Ondertussen is de vertaling voor die eenheid geproduceerd.

Een uitvoerige beschrijving van het procedure stelsel 'simple arithexp' is te vinden in [6] en [7], hier wordt volstaan met op te merken dat iedere 'simple arithexp' getransformeerd wordt in een macro programma dat korrespondeert met de zogeheten reversed polish form, bijvoorbeeld: $(a + b) \times (c - d) \uparrow e$ wordt: $(a\ b\ +)\ (c\ d\ -)\ e \uparrow \times$.

De vertaler voor het input statement moet de opdrachten genereren die de koppeling tussen de getallen uit de data en de namen uit het model en de input tot stand brengen. Dit gebeurt voornamelijk in de procedure 'variable'. Tijdens vertaling van het model statement hebben de namen rechts van het gelijk teken type 1 gekregen, de namen links ervan type 2. Komen deze namen in een variable list voor, dan worden de types veranderd in respectievelijk 3 en 5. Variabelen in de variable list die niet in het model voorkomen krijgen type 4; namen in het input statement die niet in een variable list voorkomen krijgen type 6. (dit laatste gebeurt overigens in de procedure 'inputname').

Ondertussen worden de leesopdrachten gegenereerd, die het volgende getal uit de data in de gewenste kolom zetten of weglezen. Voor variable listen, die geheel uit variabelen bestaan die niet in het model voorkomen worden opdrachten gegenereerd die de korresponderende getallen in een keer weglezen.

2 - 5 In 'check model' wordt gekontroleerd of het model statement na de koppeling aan de data door middel van het input statement nog wel aan enige elementaire eisen voldoet, zoals de eis dat iedere term het product moet zijn van een parameter en een factor, of de eis dat in die factor geen namen mogen voorkomen die niet in het input statement voorkomen (men zou dan n.l. regressie willen plegen met een te verklaren variabele zonder daarvoor waarden te hebben).

In 'check input' wordt met behulp van de datawijzer gekontroleerd of alle in de data list aangeboden getallen wel behandeld zijn. Bovendien wordt gekontroleerd of voor iedere variabele in het model wel even veel getallen gegeven zijn.

2 - 6 Het executiegedeelte van het programma wordt geactiveerd door de procedure 'execute' die onder meer de basis cyclus van een rekenautomaat simuleert:

```
basis cyclus: haal een opdracht
              verhoog de opdrachtteller met 1
              splits het adres- en opdrachtgedeelte af
              voer de opdracht uit
              ga naar: basis cyclus
```

Deze cyclus eindigt als de opdrachtteller het (vertaalde) programma dreigt te verlaten. De opdrachten zelf staan gekodeerd via de switch listen 'macro' en 'macro2'. Bij een foutmelding verricht 'endrun' enige eindriten in verband met de trommeladministratie.

Literatuur

[1] Afifi, A.A. + Azen, S.P.,
Statistical Analysis; A Computer Oriented Approach.
Academic Press, (1972).

[2] Crocker, D.C.,
Some interpretations of the multiple correlation coefficient.
The American Statitician, 26 (1972), 2, p. 31 - 33.

[3] Dekker, T.J.,
ALGOL 60 procedures in numerical algebra, part 1.
MC tract 22, Mathematisch Centrum, (1970).

[4] Draper, N.R. + Smith, H.,
Applied Regression Analysis.
John Wiley + Sons, (1966).

[5] Grune, D. (ed.),
LR 1.1, Handleiding MILLI-systeem voor de EL X8.
Mathematisch Centrum, (1971).

[6] Kruseman Aretz, F.E.J.,
ALGOL 60 translation for everybody.
Elektronische Datenverarbeitung, 6 (1964), 6, p. 233 - 244.

[7] Kruseman Aretz, F.E.J.,
Programmeren voor rekenautomaten. (De MC ALGOL 60 vertaler voor de EL X8).
MC syllabus 13, Mathematisch Centrum, (1971).

[8] Meurs, A. van + Reeken, A.J. van (ed.),
Regressie Analyse.
Rapport Hoogovens, (1970).

[9] Naur, P. (ed.),
Revised report on the algorithmic language ALGOL 60.
Regnecentralen, Copenhagen, (1964).

[10] Rosen, S. (ed.),
Programming systems and languages.
McGraw Hill Book Company, (1967).