/OOR CULTUURTECHNIEK EN WATERHUISHOUDING
NOTA nr.182 d.d. 1 maart 1963

Het oplossen van een derdegraadsvergelijking met behulp van de I.B.M. - 1620

Een voorbeeld van het werken met een computer

Ir. Ph.Th.Stol

1. Inleiding

Voor het oplossen van derdegraadsvergelijkingen bestaat geen algemene methode zoals die waarmee vierkantsvergelijkingen kunnen worden opgelost. In bijzondere gevallen kan nog wel een rekenproces gevolgd worden waarmee de wortels van de vergelijking

$$ax^3 + bx^2 + cx + d = 0 \qquad (1)$$

bepaald kunnen worden, doch het succes van de berekeningen is afhankelijk van de waarden van de optredende constanten. Ook kan een methode worden gevolgd waarmee uit hulpgrootheden de wortels berekend worden (zie BRONSTEIN-SEMENDJAJEW: "Taschenbuch der Mathematik", Instituut voor Cultuurtechniek en Waterhuishouding, nr.11/166, pagina 117 en 118).

Zijn de wortels van de derdegraadsvergelijking echter alle reeël dan kunnen deze iteratief benaderd worden met behulp van de methode van NEWTON. Deze methode zal in het kort worden uiteengezet voor het geval de kleinste positieve wortel moet worden berekend.

Tevens zal dit voorbeeld gebruikt worden om enkele inzichten te geven in de wijze waarop met behulp van elektronische rekenapparatuur een dergelijk numeriek probleem kan worden behandeld.

2. Rekenprocedure

De methode van NEWTON komt voor het boven ingeleide geval op het volgende neer. Allereerst wordt uit (2) berekend welke functiewaarde behoort bij $z = 0$ (zie figuur 1)

$$y = az^3 + bz^2 + cz + d \qquad (2)$$

Hiervoor wordt gevonden $(z_o, y_o) = (o, d)$. Daarna wordt de raaklijn in (z_o, y_o) aan (2) berekend. Hiervoor geldt

$$(y - y_o) = \left(\frac{dy}{dz}\right)_o (z - z_o) \qquad (3)$$

met

$$\frac{dy}{dz} = 3az^2 + 2bz + c \qquad (4)$$

Het snijpunt van de raaklijn (3) met de z-as wordt gevonden voor de waarde y = 0 (zie figuur 1), waaruit volgt

$$z_1 = z_o - y_o/(\frac{dy}{dz})_o \qquad (5)$$

Met dit nieuwe punt z_1 wordt nu weer met (2) de ordinaatwaarde uitgerekend volgens

$$y_1 = az_1^3 + bz_1^2 + cz_1 + d \qquad (6)$$

waarna de gehele procedure (3), (4) en (5) herhaald wordt. Het verschil tussen twee opeenvolgende uitkomsten is (zie figuur 1)

$$z_2 - z_1 = v_1$$

Men besluit de berekening te beëindigen indien

$$v_i < \varepsilon \qquad (7)$$

waarin ε een vooraf vastgestelde kleine waarde is. Met (7) geeft men aan dat voor het gestelde doel met z_{i+1} de wortel z* voldoende dicht benaderd is.

Achtereenvolgens dienen dus berekend te worden

$$y_i = az_i^3 + bz_i^2 + cz_i + d \qquad (8)$$

$$(y')_i = 3az_i^2 + 2bz_i + c$$

$$z_{i+1} = z_i - y_i/(y')_i$$

$$v_i = z_{i+1} - z_i$$

en getest of

$$v_i \overset{?}{<} \varepsilon \qquad (9)$$

de bewerking wordt herhaald (i = 0, 1, 2,....) tot dat voor i = n inderdaad

$$v_n < \varepsilon$$

waarmee de voldoende dichtbenaderde wortel gevonden is en gesteld wordt

$$z_{n+1} = z^*$$

Het blijkt dus dat steeds dezelfde formules doorgerekend moeten worden doch dit een groot aantal malen. Een dergelijke methode leent zich uiteraard zeer goed voor bewerking op een computer. Valt de test (9) negatief uit dan wordt in het machineprogramma teruggesprongen naar de beginopdracht (8) en wordt de berekening herhaald. Dit kan op de machine zo geprogrammeerd worden dat de machine zelf de nodige beslissingen neemt.

3. In- en uitvoer

Opgemerkt wordt dat nadat van de algebraische bewerking de codering in machinetaal heeft plaatsgevonden nog gezorgd moet worden voor

1. invoer van de constanten a, b, c, d
   invoer van een waarde voor ε
2. uitvoer van de resultaten in de gewenst vorm

De in- en uitvoer kan plaatsvinden door middel van ponskaarten of door middel van de aangesloten elektrische schrijfmachine.

Normaal gebruikelijk is het een rekenprogramma door middel van de ponskaartentoevoer in het geheugen van de machine te laten opnemen.

De snelste wijze van uitvoer van resultaten is die waarbij de uitkomsten weer op kaarten worden vastgelegd, het nadeel is dat het teruglezen van de aldus verkregen kaarten plaats moet vinden door uitschrijven via de tabelleermachine. Dit betekent het verrichten van extra handelingen en maakt het direct beoordelen van de resultaten omslachtiger. Langzamer uitvoer vindt plaats met de schrijfmachine doch er ontstaat een direct leesbare tabel of tekst. De lay-out van de uitvoer dient eveneens geprogrammeerd te worden.

De uitvoer van een gedeelte tekst kan zo plaatsvinden dat een zeker samenspel met de machine ontstaat. De machine kan dan bijvoorbeeld uittypen dat nieuwe gegevens moeten worden ingevoerd, dat een bepaalde schakelaar moet worden gesteld enz. Voorbeelden van dergelijke "boodschappen" zullen nog gegeven worden.

4. De grammatica van het programmeren

Een programma voor de I.B.M. 1620 kan geschreven worden in machinetaal of in FORTRAN (Formula Translation). Het gebruik van machinetaal heeft het voordeel dat een zeer zuinig gebruik van de geheugencapaciteit kan worden

gemaakt. FORTRAN is minder zuinig in geheugengebruik doch het schrijven van een programma in deze taal kan veel sneller plaatsvinden. Het in FORTRAN geschreven source program kan door de computer zelf vertaald worden in het eigenlijke object program, en het is dit laatste programma dat als werkprogramma gebruikt wordt.

Het ligt niet in de bedoeling een bespreking van FORTRAN te geven doch enkele opmerkingen mogen ter verduidelijking dienen.

4.1. Van een algebraïsche uitdrukking in FORTRAN geeft het linkerlid aan dat de daar voorkomende variabele in het geheugen van de machine vervangen wordt door de (berekende) waarde uit het rechterlid. Voorbeeld:

$$X = A + B$$

De variabele die in het geheugen met X wordt aangeduid en bijvoorbeeld de waarde 10,3 heeft krijgt na uitvoering van deze opdracht de waarde van A + B, bijvoorbeeld - 0,051, terwijl A en B onveranderd blijven.

Een wat meer bijzonder voorbeeld is (FORTRAN!)

$$X = X + 1 \qquad (10)$$

De variabele die in het geheugen op de plaats staat die met X wordt aangeduid en dus de waarde van X heeft (bijvoorbeeld 13,5) wordt vervangen door de waarde X + 1 (dus 14,5)

4.2. Een enkele variabele kan met een combinatie (van maximaal) 5 letters of cijfers worden aangeduid. Ook kunnen geïndiceerde variabelen gebruikt worden. Er bestaat alleen een code voor hoofdletters en voor alle tekens op dezelfde hoogte. Veelal zal men van lettercombinaties gebruikmaken om een variabele gemakkelijk herkenbaar te maken. Hierin is men geheel vrij.

Voorbeelden.

	algebraïsch	FORTRAN (b.v.)
	ε	EPS
	$a(b + c)^2$	A *(B + C)* * 2
geïndiceerd	x_{ij}	X(I,J)
niet-geïndiceerd	x,y,z	X1,X2,X3

4.3. Bij in- en uitvoer, welke door middel van ponskaarten of schrijfmachine wordt verzorgd, moet aangegeven worden in welke vorm de getallen voorkomen.

Er bestaan drie modificaties

a. Integer

Bijvoorbeeld FORMAT (I2): er worden twee plaatsen gelezen (geschreven) waarin maximaal 2 tekens inclusief het ±teken passen dus (met b voor "blank") bijvoorbeeld

(15): heeft FORMAT (I2) (minimum)
(b3): heeft FORMAT (I2)
(-133): heeft FORMAT (I4)
algemeen ±xxxx heeft FORMAT (I5) (maximum)

b. Floating Point

Bijvoorbeeld FORMAT (F 8.3): er worden 8 plaatsen gelezen (geschreven) waarin maximaal 8 tekens passen inclusief het ±teken en de komma (punt); er staan 3 cijfers achter de komma dus bijvoorbeeld

bb12.345 F 8.3
b12.345 F 7.3
bb-0.1 F 6.1
algemeen ±.xxxxxxxx F 10.8 (maximum)

c. Exponentieel

Bijvoorbeeld FORMAT (E 8.3): er worden 8 plaatsen gelezen (geschreven) waarin maximaal 8 tekens passen inclusief het ±teken, de komma (punt) en de aanduiding van de exponent; er staan 3 cijfers achter de komma, dus bijvoorbeeld

(0,123): .123E-00 E 8.3
(-12,345): -.12345E+02 E 11.5
algemeen: ±.xxxxxxxE±xx E 14.8 (maximum)

Uitkomsten van berekeningen waarvan men de orde van grootte niet nauwkeurig kent moeten dus steeds in de maximale exponentiële modificatie gegeven worden, daar anders de mogelijkheid bestaat dat van belang zijnde cijfers niet in het FORMAT passen en dus niet uitgeschreven worden.

Tenslotte kan genoemd worden

d. Hollerith

FORMAT (5H.....): er volgen 5 te schrijven (ponsen) tekens inclusief blanks, bijvoorbeeld

```
FORMAT (5HbX1=b)
FORMAT (17HNIEUWEbBEREKENING/)
```

De "slash" aan het einde geeft aan dat bij uitvoer door middel van de schrijfmachine een regel moet worden overgeslagen.

4.4. Elk teken zoals *.),(enz. heeft zijn eigen specifieke betekenis. Bij het schrijven van een programma moeten deze tekens volgens de voorschriften gebruikt worden. Foutief gebruik geeft aanleiding tot moeilijkheden bij de vertaling van de FORTRAN-tekst, het vertaalde programma is dan onbruikbaar. Veel voorkomende fouten worden overigens door de machine zelf, tijdens de vertaling, gesignaleerd onder vermelding van de soort van de fout.

4.5. Alle opdrachten worden in de volgorde waarin zij geschreven zijn uitgevoerd, tenzij een sprongopdracht een andere volgorde aangeeft. De opdrachten mogen op geheel willekeurige wijze met nummers worden aangeduid, er behoeft geen doorlopende nummering te worden gebruikt.

5. Het blokdiagram (flow-chart)(zie figuur 2)

Een gemakkelijk hulpmiddel bij het schrijven van een programma is het blokdiagram. Hierin worden schematisch, al of niet uitgebreid, de uit te voeren bewerkingen overzichtelijk bijeengebracht. Het blokdiagram kan als de plattegrond beschouwd worden waarmee de weg door het programma te vinden is. Het programma zelf zou dan als "stratenboekje" beschouwd kunnen worden.

6. Voorbeeld van het gebruik van FORTRAN

Als voorbeeld van de wijze waarop een iteratief proces in FORTRAN wordt geschreven dient het volgende programmagedeelte

```
      .
      .
      .
      .
      Z2 = 0.0
10    Z1 = Z2
      Z2 = functie van Z1
      IF((Z2-Z1)-EPS)5,5,10
 5    TYPE 12,Z2
12    FORMAT (19HEINDWAARDEbVOORbZ2=F6.1/)
      .
      .
      .
      .
```

De bovenstaande regels, elk één opdracht vertegenwoordigende, worden elk op één kaart geponst. Een opdracht mag daarom ook nooit langer zijn dan het aantal tekens (72) dat op een ponskaart aangebracht mag worden.

De eerste opdracht luidt: zet op het in het geheugen met Z2 aangeduide veld een 0.0.

De tweede opdracht luidt: breng de waarde van Z2 (dus 0.0) over naar het veld Z1.

De derde opdracht luidt: breng de waarde die in het rechterlid ontstaat wanneer de (hier niet gespecificeerde) functie wordt uitgerekend naar Z2. Hiermede is dus Z2 niet meer gelijk aan 0.0 doch is bijvoorbeeld 32.6 geworden.

De vierde opdracht luidt: bereken Z2-Z1 (in dit geval dus 32.6-0.0 = 32.6) Trek hiervan af EPS (b.v. eerder in het programma gedefinieerd als 0.5) en er komt 32.1. De laatste drie getallen van deze opdracht betekenen

indien 32.1 < 0 ga naar 5
indien 32.1 = 0 ga naar 5
indien 32.1 > 0 ga naar 10

de laatste keuze wordt hier aangehouden en de volgende opdracht is weer die welke met 10 is aangeduid (2e regel).

Nu wordt dus de waarde van Z2 welke 32.6 is geworden overgebracht naar Z1. Vervolgens wordt een nieuwe Z2 uitgerekend stel 32.9 en de test verloopt als volgt

$$((32.9 - 32.6) - 0.5) = -0.2 < 0$$

zodat nu de opdracht met aanduiding 5 uitgevoerd wordt. Deze opdracht houdt

in dat de waarde van Z2 volgens het gegeven FORMAT moet worden uitgeschreven. Dit FORMAT telt een tekst van 19 tekens en er komt:

EINDWAARDE VOOR Z2 = 32.9

daarna wordt nog een regel overgeslagen (slash).

Vervolgens wordt de volgende, hier niet meer aangegeven opdracht uitgevoerd.

7. Voorbeeld van het schrijven van een programma in FORTRAN

Van het waterstandsproefveld te Nieuw Beerta staan onder meer de volgende gegevens ter beschikking

nr	Waterstand H	Stikstof N	Opbrengst q_o
1	40	30	33,5
2	40	30	32,6
3	40	90	43,8
4	60	30	37,8
.	.	.	.
i	.	.	.
.	.	.	.

in totaal 14 gegevens. VISSER stelt voor de relatie tussen de variabelen te beschrijven met de volgende formule

$$(Q - q)(a - \frac{q}{H+b})\ (c - \frac{q}{N+dH+e}) = Fq \qquad (11)$$

Als onderdeel van de nu uit te voeren vereffening moet bij elk waardenpaar (H,N) en bij voorlopige waarden van de constanten a, b, c, d, e, F, Q berekend worden welke q aan (11) voldoet. Opgemerkt wordt dat (11) een derdegraadsvergelijking in q is en dat de kleinste positieve wortel (q*) bepaald moet worden. Dit probleem is dus identiek aan dat uit paragraaf 2. Tenslotte kan nog de som van kwadraten van afwijkingen berekend worden, namelijk:

$$\Sigma (q_o - q^*)^2 \qquad (12)$$

Dit onderdeel van het vereffeningsprogramma werd afzonderlijk getest,

waarvoor het als een zelfstandig programma werd, geschreven.

Het is de snelste wijze van werken om uitdrukkingen die meer dan eenmaal voorkomen een eigen symbool te geven. Het opzoeken van een in het geheugen opgeborgen waarde kost de machine minder tijd dan het opnieuw uitrekenen van een algebraïsche uitdrukking. Deze symbolen worden hieronder nader gedefinieerd.

Voor het oplossen van (11) als derdegraadsvergelijking werd de volgende vorm gebruikt

$$Y = (Q - q)[a(H + b) - q][c(N + dH + e) - q] - F(H + b)(N + dH + e)q$$

In de FORTRAN-tekst werden deze symbolen veranderd in

$Y = UITK$
$Y' = TANG$

a = A(1)	e = A(5)	H = X(1)
b = A(2)	F = A(6)	N = X(2)
c = A(3)	Q = A(7)	q = X(3)
d = A(4)		

met als hulpberekeningen voor de functie van Z_1 (zie paragraaf 6)

```
T1 = A(7) - Z1                    (oorspronkelijk dus (Q - q) enz)
T2 = X(1) + A(2)
T3 = A(1) * T2 - Z1
T4 = X(2) + A(4) * X(1) + A(5)
T5 = A(3) * T(4) - Z1
```

en

```
UITK = T1 * T3 * T5 - A(6) * T2 * T4 * Z1
```

waarmee dan (4) en (5) de volgende vorm krijgen

```
TANG = (- T3 * T5 - T1 * T5 - T1 * T3 - A(6) * T2 * T4)
  Z2 = Z1 - UITK/TANG
```

Op identieke wijze als uiteengezet is in paragraaf 2 kan de wortel q (dus X(3)) met de methode van NEWTON opgelost worden. Hiertoe worden, evenals in paragraaf 6 werd aangegeven, iteratieve bewerkingen met de grootheden Z2 en Z1 uitgevoerd. Is de oplossing Z2 met voldoende nauwkeurigheid

vastgesteld, dan wordt (12) berekend, namelijk

$$S^2 = \sum_{i=1}^{14} (X(3) - Z2)^2 \qquad (13)$$

waarvoor gebruikt werd het symbool

SQO = som kwadraten

In FORTRAN kan (13) in de volgende twee stappen berekend worden door

1. vóór de gehele bewerking te definiëren

SQO = 0.0

2. telkens nadat een oplossing Z2 verkregen is te laten berekenen

SQO = SQO + (X(3) - Z2) * * 2

waardoor - zie paragraaf 4.1 en de eigenschap van (10) - vanzelf de sommatie wordt verkregen.

Een blokdiagram van het programma wordt gegeven in figuur 2. Het begin van het diagram is de "START". Achtereenvolgens moeten een waarde voor EPS ingelezen worden en de waarden die de constanten A(I) aannemen. Het inlezen geschiedt door middel van ponskaarten. Vervolgens wordt SQO op nul gesteld en wordt de ingelezen waarde van EPS uitgetypt. (Vergelijk met bijlage 1, waarin het volledige programma staat uitgeschreven).

Vervolgens moet nu voor elk waardenpaar (H,N) de derdegraadsvergelijking (11) opgelost worden zodat er in totaal met 14 gegevens ook 14 oplossingen moeten worden berekend. In FORTRAN kan een dergelijke cyclus aangegeven worden met een DO-opdracht. Dit wordt gedemonstreerd in het blokdiagram. De opdracht luidt:

DO 712 L = 1,14

dat wil zeggen dat alle volgende opdrachten in volgorde worden uitgevoerd de eerste maal met L = 1, met als laatste de opdracht met aanduiding 712. Daarna wordt automatisch de index opgehoogd tot L = 2 en worden dezelfde opdrachten herhaald. Nadat de cyclus is doorgewerkt met L = 3, 4,....., en tenslotte met L = 14 wordt het programma verder afgewerkt, "wordt de DO-lus verlaten", in dit geval met als volgende opdracht

TYPE SQO

In de "DO-lus" is in het programma de mogelijkheid opengelaten door het afzetten van schakelaar 1 op het bedieningspaneel, het laten uittypen van de tussenresultaten over te slaan. Na het typen van de laatste regel: LOAD EPS, A, X, START stopt de computer op de PAUSE. Na indrukken van de startknop wordt opdracht 715 uitgevoerd.

8. Het vertalen van het source (FORTRAN) program

Zoals reeds werd vermeld kan het FORTRAN-programma door de computer zelf vertaald worden in het objectprogramma. Hiertoe worden de volgende handelingen verricht (zie bijlagen 1 en 2).

1. het geheugen wordt schoongemaakt (op nul gezet) met de opdracht 160001000000, via de schrijfmachine in te voeren

2. de compiler (kaartenpakket dat het vertaalprogramma omvat) wordt in de kaartenlezer geplaatst waarna gestart wordt.

Zijn alle kaarten van het vertaalprogramma gelezen dan meldt de computer:

ENTER SOURCE PROGRAM, PUSH START

Vervolgens worden de kaarten met de FORTRAN-tekst geladen waarna de vertaling begint. De FORTRAN-tekst wordt uitgetypt en voorzien van de adresnummers te beginnen bij 8300. Deze adresnummers verwijzen naar de plaats in het geheugen waar de betrokken opdracht zal komen (zie bijlage 1).

Opgemerkt wordt dat een nummer waarachter een C staat niet een vertaalde opdracht aanwijst, doch een regel tekst die als verduidelijking aan het programma is toegevoegd.

Met de opdracht (zie bijlage 1)

DIMENSION A(7), X(3)

wordt aangegeven dat de index van de variabelen A(I) en X(I) respectievelijk tot 7 en 3 maal zal lopen.

Na het voorgaande (paragraaf 6 en 7) spreekt bijlage 1 verder voor zich.

Nadat de kaart met END de lezer gepasseerd is, wordt de vertaling beeindigd. De schrijfmachine typt nu de boodschap (bijlage 2)

PROG SW 1 ON FOR SYMBOL TABLE, PUSH START

Met schakelaar (switch) 1 "aan" worden vervolgens alle gebruikte symbolen uitgetypt onder vermelding van het adres waar deze in het geheugen te

vinden zijn. Elk symbool beslaat 10 posities. Het hoogste getal wordt hiervan gegeven. Het eigenlijke adres van bijvoorbeeld EPS is dus 19740. De lijst wordt begonnen met de subroutines sinus,....., vierkantswortel. Daarna komen de 7 symbolen A(1), A(2),.....,A(7) enz.

Tenslotte verschijnt de boodschap

PROCESSING COMPLETE

Het objectprogramma is tijdens de vertaling vastgelegd op een nieuwe serie ponskaarten en het is dit programma dat voor de berekeningen gebruikt wordt. Het FORTRAN-programma heeft hiermee zijn dienst gedaan. De handelingen zijn nu: het op nul stellen van het geheugen, het laden van het objectprogramma, het laden van de gegevens.

9. Enkele voorbeelden van de snelheid van werken

De snelheid waarmee de bovenbeschreven berekening tot stand komt hangt af van

1. de snelheid waarmee de computer alle voorgeschreven opdrachten uitvoert
2. de snelheid waarmee de convergentie naar z* (zie figuur 1) plaatsvindt.

Het is duidelijk dat naarmate men ε in (7) kleiner neemt, het aantal iteratiestappen toeneemt. Het programma uit bijlagen 1 en 2 biedt het voordeel dat ε gemakkelijk een andere waarde gegeven kan worden zodat oplossingen met verschillende mate van nauwkeurigheid onderling vergeleken kunnen worden. Het blijkt echter dat de convergentie zo snel verloopt dat reeds spoedig een bevredigende benadering van z* wordt verkregen.

In bijlage 3 staat aangegeven volgens welke lay-out de resultaten worden uitgetypt. Het eerste getal is de SQO van een vorige serie berekeningen, daarna de "boodschap" die aangeeft welke handelingen moeten worden verricht. Een en ander kan met het volledige programma (bijlage 1) gemakkelijk geverifieerd worden (zie ook figuur 2).

Dan volgt de uitgetypte waarde voor ε , hier dus gelijk 0.

Vervolgens de waarde van q_o en daaronder de achtereenvolgende benaderingen van de derdegraadswortel. De som van de kwadraten van afwijkingen wordt dan berekend met

$$(0.49000000E + 02 - 0.46833270E + 02)^2$$

als eerste bijdrage tot de kwadraatsom.

Het valt op dat in beide gevallen reeds na 7 stappen de in totaal 8 significante cijfers gelijk zijn geworden. De verschillen tussen de opeenvolgende uitkomsten worden weergegeven in de volgende tabel.

Benadering	Verschil = V_i	ε waarvoor $V_i < \varepsilon$
0,000000		
	25,670781	
25,670781		
	14,274452	
39,945233		
	5,797486	10,000000
45,742719		
	1,056333	5,000000
46,799052		
	0,034183	0,500000
46,833235		
46,833270		
46,833270		
0,000000		
	15,761986	
15,761986		
	9,649723	10,000000
25,411709		
	5,024226	
30,435935		
	1,662623	5,000000
32,098558		
	0,185540	0,500000
32,284098		
	0,002227	
32,286325		
	0,000000	
32,286325		

Wil men dus de uitkomst in 1 decimaal nauwkeurig hebben dan zou reeds de laatst berekende waarde met $\varepsilon = 0,5$ gebruikt kunnen worden.

Wat de snelheid van de computer betreft tenslotte nog het volgende. Voor steeds 5 achtereenvolgende bewerkingen (in plaats van 14) werd de tijd gemeten die verliep tussen het starten van de machine tot en met het uittypen van de boodschap LOAD EPS, A, X, START. Dit werd gedaan voor een paar waarden van ε en al of niet met uitschrijven van tussenresultaten (SWITCH 1!)

Het resultaat was als volgt.

Tabel van gemeten tijden in seconden voor 5 achtereenvolgende berekeningen

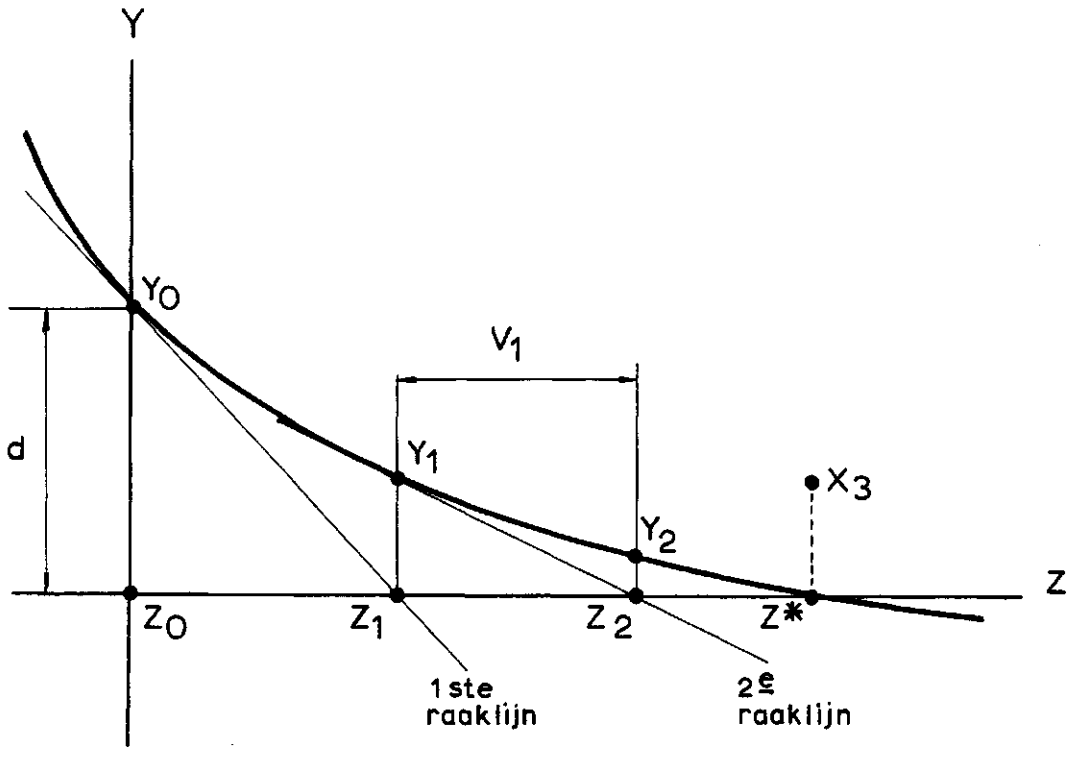
ε	met uittypen	zonder uittypen
0.0	148	26
0.5	114	24
10.5	73	22

Met deze tabel wordt duidelijk gedemonstreerd welke vertraging het laten uittypen van veel resultaten betekent. Het liefst moet men zich in het gebruik van de schrijfmachine zoveel mogelijk beperken ten einde het volle profijt te kunnen trekken van de snelheid van de computer zelf.

```
          1600010000000RS
 FORTRAN FORMAT - PROCESSOR -  20K
ENTER SOURCE PROGRAM, PUSH START
08300 C     OPL.3DE-GRAADS VERG. OPBRENGSTFUNCTIE NW-BEERTA JAN 1963
08300 C                  LOAD RESP EPS (1KRT),A(7KRTN),X(14KRTN)
08300 C     1 ON= PRINT
08300 C
08300       DIMENSION A(7),X(3)
08300 C
08300   715 READ 709,EPS
08324       DO 710 I=1,7
08336   710 READ 711,A(I)
08420       SQO = 0.0
08444       PRINT 609,EPS
08468       DO 712 L=1,14
08480       READ 713,X(1),X(2),X(3)
08528       IF (SENSE SWITCH 1) 10,11
08548    10 TYPE 719,X(3)
08572    11 Z2 = 0.0
08596   714 Z1 = Z2
08620 C
08620       T1 = A(7)-Z1
08656       T2 = X(1)+A(2)
08692       T3 = A(1)*T2-Z1
08740       T4 = X(2)+A(4)*X(1)+A(5)
08800       T5 = A(3)*T4-Z1
08848 C
08848       TANG=(-T3*T5-T1*T5-T1*T3-A(6)*T2*T4
09052       UITK= T1*T3*T5-A(6)*T2*T4*Z1
09184       Z2 = Z1-UITK/TANG
09244 C
09244       IF (SENSE SWITCH 1) 12,13
09264    12 TYPE 709,Z2
09288 C
09288    13 IF((Z2-Z1)-EPS) 712,712,714
09368   712 SQO = SQO + (X(3)-Z2)**2
09464       TYPE 717,SQO
09488       TYPE 718
09512       PAUSE
09524       GOTO 715
09532 C
09532   609 FORMAT (/E14.8/)
09564   709 FORMAT (E14.8)
09586   711 FORMAT (E14.8)
09608   713 FORMAT (2F5.o,F8.1)
09640   717 FORMAT (/E14.8)
09668   718 FORMAT (/18HLOAD EPS,A,X,START)
09734   719 FORMAT (//F6.2)
09766 C
09766       END
```

```
 .82254558E+02

LOAD EPS,A,X,START

 .00000000E-99


 49.00
 .25670781E+02
 .39945233E+02
 .45742719E+02
 .46799052E+02
 .46833235E+02
 .46833270E+02
 .46833270E+02


 33.50
 .15761986E+02
 .25411709E+02
 .30435935E+02
 .32098558E+02
 .32284098E+02
 .32286325E+02
 .32286325E+02



 enz
```

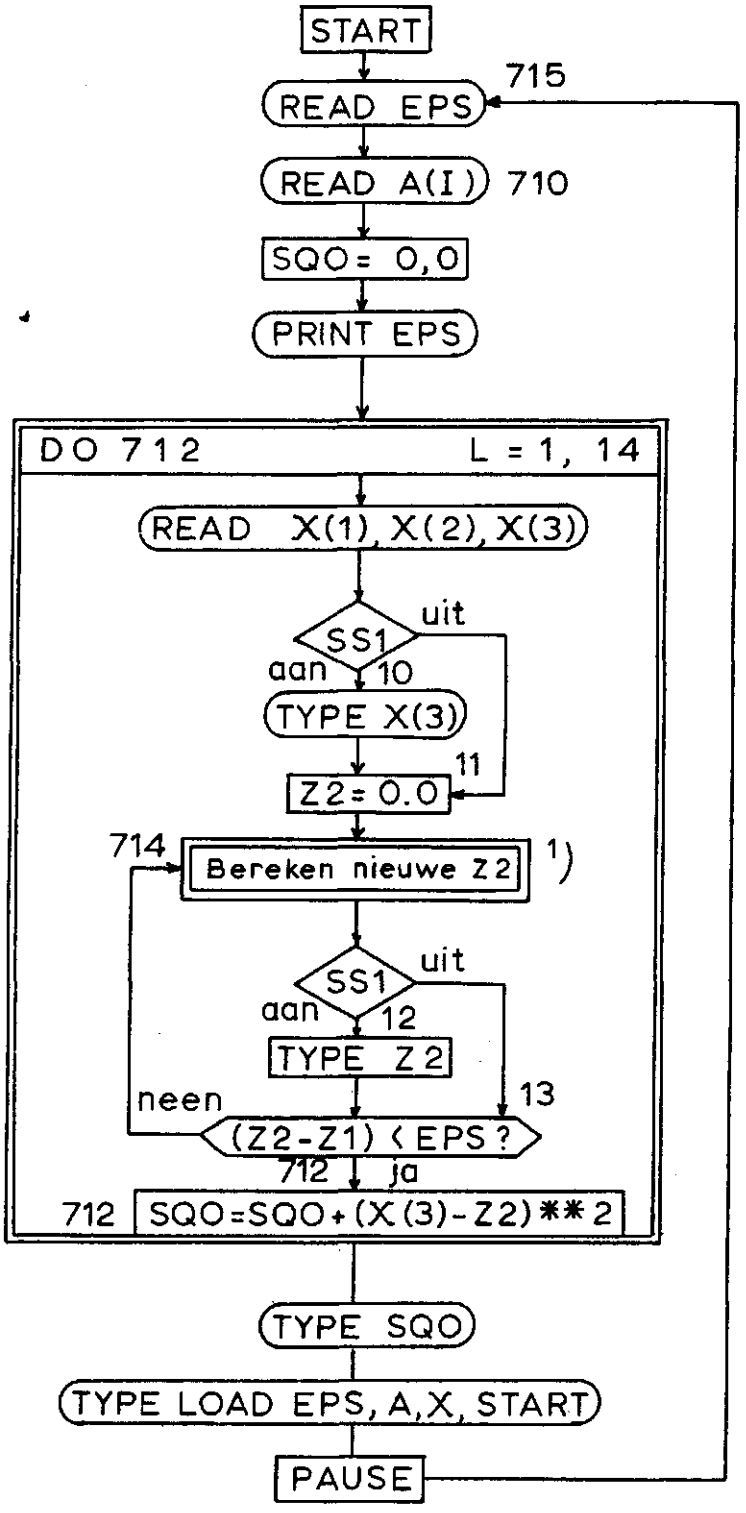
FIG. 1



BENADERING VAN DE KLEINSTE (positieve) WORTEL Z* VAN EEN DERDE GRAADS VERGELIJKING

FIG. 2

HOOFDELEMENTEN VAN HET PROGRAMMA TOT OPLOSSEN VAN EEN DERDE GRAADS VERGELIJKING (zie tekst)



(SS1 : SENSE SWITCH 1)

[1]) dit gedeelte bevat de opdrachten 08596 t/m 09184 (zie bijlage 1)