ERRATA in syllabus Vacantiecursus 1987

blz.	regel	er staat	er moet staan
6	18↑	overeentestemmen	overeen te stemmen
6	13↑	X	x
14	9↑	zijn,	zijn.
15	3↑	een punt	punten
17	18↑	*RLRLI*	*RLRLI*
41	(13.1)	P_k	P_r
45	8↑	oneven n	oneven $n>1$
45	8↑	een a	een $a>1$
47	16↑	dat aan	dat niet aan
47	15↑	die aan	die niet aan
53	7↓	binnen	in
53	7↓	cirkel	cirkel (inclusief rand)
54	10↑	van	van twee (of meer)
59	7↑	;	,
63	4↑	voeren	voren
65	10↑	dus	zodat
74	1↓	pivoten	pivoteren
97	18↑	$f(x)=0$	$f(x)=x$
97	14↑	-grafieken	-grafiek
102	4↑	vrijvoorbeeld	bijvoorbeeld

CWI Syllabi

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

The CWI is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

CWI Syllabus 14

Vacantiecursus 1987
De personal computer
en de wiskunde op school

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

ISBN 90 6196 323 0
NUGI-code: 811

Printed in the Netherlands

INHOUD

Voorwoord

Zoals de titel van deze CWI-syllabus al aangeeft, heeft de commissie ter voorbereiding van de Vacantiecursus 1987 tot onderwerp gekozen "De Personal Computer en de Wiskunde op School".

Dit is een bijzonder actueel thema, waaraan, ook in de komende jaren, ongetwijfeld nog veel aandacht besteed zal moeten worden, wil men het gebruik van de PC in het voortgezet onderwijs op zinvolle en doeltreffende wijze benutten.

Het CWI draagt hier gaarne haar steentje toe bij.

Deze syllabus bevat de op schrift gestelde voordrachten van de vier uitgenodigde sprekers. Een blik in de inhouds-opgave zal voldoende zijn om een indruk te krijgen van de te behandelen stof.

Hierbij kan nog opgemerkt worden dat in deze syllabus o.m. een aantal GWBASIC-programma's zijn opgenomen die, in enigszins aangepaste vorm, ook op (5¼ inch) diskette bij het CWI verkrijgbaar zijn. Deze diskette bevat verder nog een aanzienlijk aantal programma's die ontleend zijn aan het binnenkort in de ϵ-reeks te verschijnen boek "Analyse en Meetkunde met de Micro Computer" van de hand van H.A. Lauwerier.

Tenslotte is hier een woord van dank aan allen die hebben bijgedragen aan de totstandkoming van deze Vacantiecursus en de bijbehorende syllabus op zijn plaats.

J. van de Lune

Fractals, Meervoudige Gelijkvormigheid

H.A. Lauwerier
Centrum voor Wiskunde en Informatica
Kruislaan 413, 1098 SJ Amsterdam

0. Inleiding

Het thema van deze voordracht is vlakke meetkunde met de computer en wel het uitvoeren van gelijkvormigheidstransformaties op het beeldscherm. De beantwoording van de vragen: hoe doe je dat en hoe kan ik begrijpen wat ik zie, motiveert ons aandacht te besteden aan (een beetje) groepentheorie, het tweetallig en drietallig talstelsel, complexe getallen, een beetje topologie en kansrekening!

Als basiskennis dient van de coördinatenmeetkunde de beschrijving van een gelijkvormigheidstransformatie. Een willekeurige transformatie T waarbij t.o.v. een vast coördinatenstelsel OXY het punt $P(x,y)$ overgaat in $P'(x',y')$ wordt beschreven als

$$T\begin{cases} x' = f(x,y) \\ y' = g(x,y)\,. \end{cases} \tag{1}$$

Hoewel we ons hier beperken tot gelijkvormigheidstransformaties zijn er een paar fundamentele zaken waarbij de aard van de transformatie er niet toe doet. Een punt P dat bij transformatie op zijn plaats blijft, $T(P)=P$, heet een *dekpunt.* Een rechte lijn, een kromme of algemener een puntverzameling V die door T in zichzelf wordt afgebeeld, heet *invariant* t.o.v. T. In notatie: $T(V)\subset V$.

Voorbeelden

1. $T: x' = x + 1,\ y' = y$ is een translatie. Er zijn geen dekpunten maar alle horizontale lijnen zijn invariant.
2. $T: x' = -y,\ y' = x$ is een kwartdraai. De oorsprong, het rotatiecentrum, is dekpunt. Alle cirkels die de oorsprong als middelpunt hebben zijn invariant.

Zijn T_1 en T_2 twee transformaties dan heet de gecombineerde transformatie 'eerst T_1 dan T_2' het *produkt* van T_1 en T_2. Notatie T_2T_1 (in de volgorde van rechts naar links). Analoog kunnen we het produkt T_1T_2 (eerst T_2 en dan T_1) vormen. In het algemeen zijn deze twee produkten verschillend.

Voorbeeld

$T_1: x' = x + 1,\ y' = y$ en $T_2: x' = -y,\ y' = x$.
T_2T_1 wordt berekend als

$$x'' = -y' = -y\ ,\ y'' = x' = x + 1.$$

T_1T_2 analoog

$$x'' = x' + 1 = -y + 1\ ,\ y'' = y' = x.$$

Dus

$$T_2T_1: x' = -y\ ,\ y' = x + 1 \quad \text{en}$$
$$T_1T_2: x' = -y + 1\ ,\ y' = x.$$

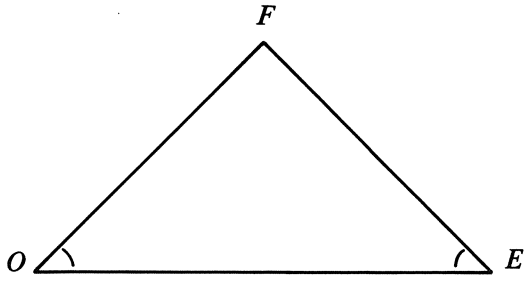
De (eenvoudige) meetkundige interpretatie laten we aan de lezer over.

Herhaling van dezelfde transformatie T levert de volgende symbolische machten $T^2, T^3, T^4, \ldots$. De transformatie die alles op zijn plaats laat, de z.g. identieke transformatie $x' = x,\ y' = y$ noteren we als I. We schrijven dan ook $T^1 = T$ en $T^0 = I$. De aanzet tot een halfgroepstructuur of een groepstructuur is hiermede gegeven. Het centrale thema is hier echter:

> *T_1 en T_2 zijn twee gegeven transformaties. Vorm alle transformaties T die hieruit door herhaalde produktvorming gevormd kunnen worden. Wat kunnen we zeggen over de puntverzameling V die invariant is t.o.v. alle T.*

1. De kromme van Levy

Bij wijze van illustratief voorbeeld laten we zien hoe de z.g. kromme van Levy gevormd wordt. De kromme zelf kan niet getekend worden, maar de gebroken lijn in fig.3, bestaande uit 2^{12} gelijke lijnstukjes, is hiervan een redelijke benadering. Uitgangspunt is een gelijkbenig rechthoekige driehoek *OEF* (zie fig.1).

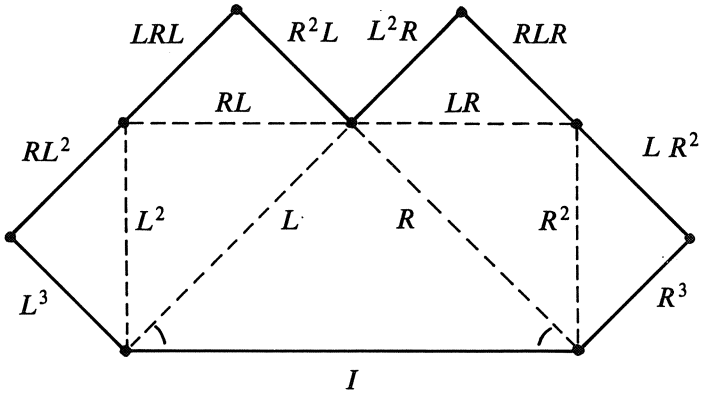


Figuur 1

De gelijkvormigheidstransformatie L heeft O als centrum en voert E in F over. De rotatiehoek is dus 45^o en de verkorting is $1/\sqrt{2}$. R is de gespiegelde transformatie die E als dekpunt heeft en O in F overvoert. Om er in een computerprogramma mee te kunnen werken, moeten L en R in coördinaten beschreven worden. Is O de oorsprong en E het punt (1,0) dan is

$$L\begin{cases}x' = (x-y)/2\\ y' = (x+y)/2\end{cases} \qquad R\begin{cases}x' = (x+y+1)/2\\ y' = (-x+y+1)/2.\end{cases} \tag{2}$$

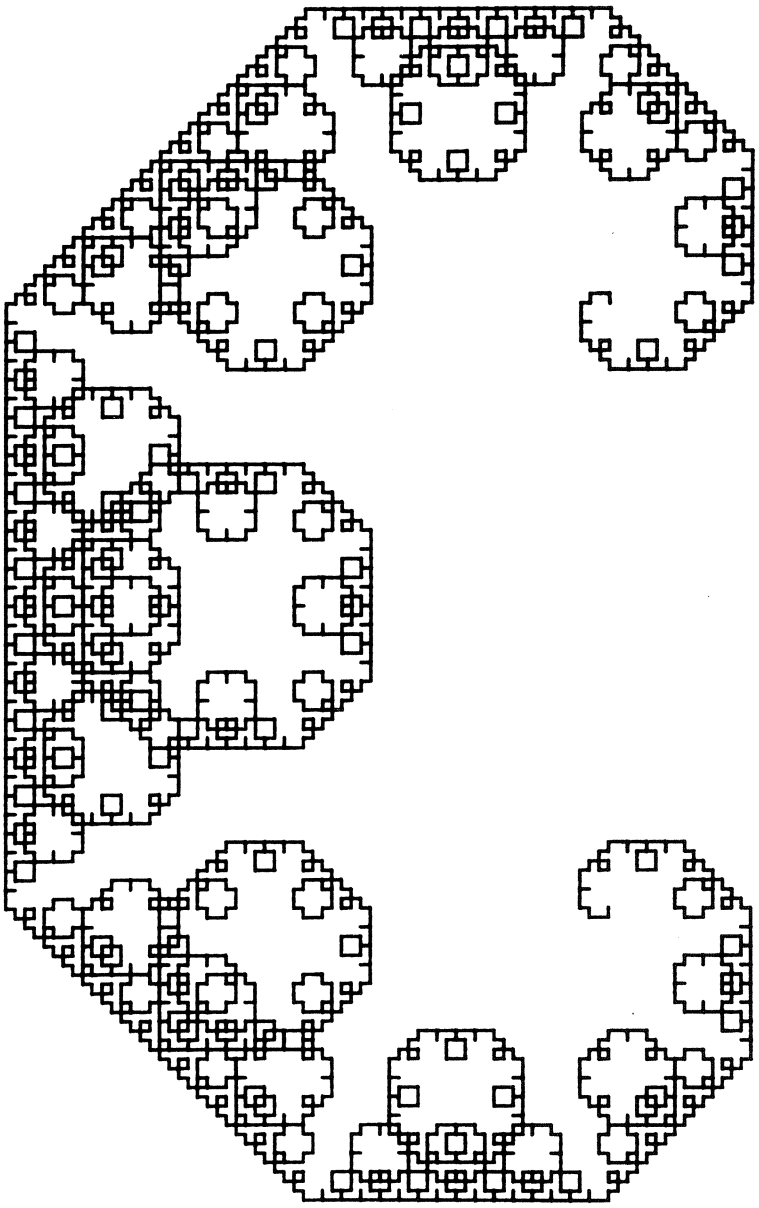
In fig.2 zien we wat er na drie transformaties (dat zijn de acht combinaties van L en R) met het basislijnstuk OE is gebeurd. De acht beelden voegen zich aaneen tot een gebroken lijn van acht gelijke stukjes.



Figuur 2

Na 12 stappen is fig.3 ontstaan, verkregen met het bijgaande computerprogramma genaamd LEVY (zie voor alle programma's bij dit hoofdstuk de Appendix op pagina's 20 t/m 23). De limietfiguur heet de kromme van Levy.

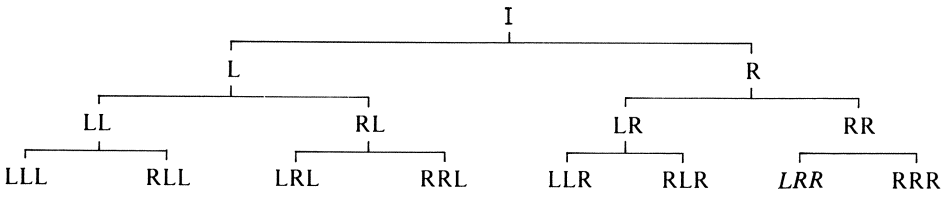
Die kromme, een echte fractal, heeft een aantal merkwaardige eigenschappen. Elk onderdeel, hoe klein ook, is gelijkvormig met het geheel. De kromme is continu maar heeft nergens een raaklijn en bovendien is de kromme oneindig lang.



Figuur 3

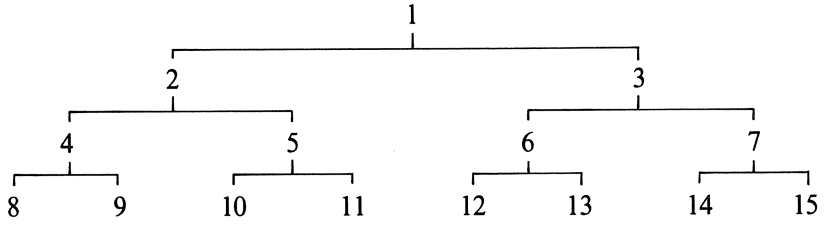
2. TWEETALLIG TALSTELSEL

De constructie van figuren als de kronkellijn van Levy kan beschreven worden met een passende boekhouding van de uit L en R te vormen samengestelde transformaties. Een prettige rangschikking is die van fig.4.



Figuur 4

Eenvoudiger, zeker om typografische redenen, is een nummering volgens het schema van fig.5.



Figuur 5

Wanneer we deze getallen in het tweetallig stelsel zouden schrijven, zouden we een opmerkelijke overeenkomst ontdekken:

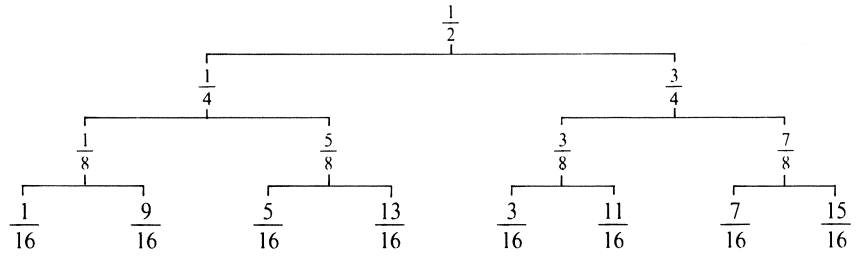
$$5 = 1\underline{01} \approx RL$$

$$11 = 1\underline{01}1 \approx RRL \text{ enz.}$$

De voorste binaal, toch altijd een 1, doet niet mee. We identificeren L met 0, R met 1 en we lezen de symbolen in tegengestelde volgorde. Het komt er dus op neer dat we alle transformaties die uit eindig veel factoren L en R bestaan, kunnen nummeren volgens de rij der natuurlijke getallen te beginnen met 1. Omdat we de mogelijkheid willen hebben het aantal factoren tot oneindig uit te breiden, is het beter met tweetallige breuken te werken, d.w.z. de transformaties af te beelden op getallen uit [0,1]. Er zijn verschillende manieren om dat te bewerkstelligen. Een aardige methode is om wat we met de transformaties L en R doen, te imiteren met de transformaties

$$L_0: x' = x/2, \quad R_0: x' = (x+1)/2, \tag{3}$$

werkend op getallen x uit [0,1]. L_0 is een centrale vermenigvuldiging met factor 1/2 en dekpunt $x=0$, R_0 is een overeenkomstige transformatie met dekpunt $x=1$. Het is het eenvoudigst het punt $x=1/2$ als transformatieobject te nemen. We krijgen dan het lijstje



Figuur 6

Het verband met fig.4 wordt duidelijk wanneer we de breuken in het tweetallig stelsel ontwikkelen. Doen we dit voor b.v. de onderste rij

.0001	.1001	.0101	.1101	.0011	.1011	.0111	.1111
LLL	RLL	LRL	RRL	LLR	RLR	LRR	RRR

dan blijkt de volgorde van de binalen precies overeentestemmen met die van de transformaties. De laatste binaal, altijd een 1, doet daarbij niet mee.

We kunnen ons nu ook voorstellen wat er gebeurt wanneer we oneindig veel transformaties achter elkaar uitvoeren. Dat correspondeert met een oneindig voortlopende breuk in het tweetallig talstelsel

$$X = .b_1b_2b_3b_4b_5 \cdots$$

al of niet periodiek, een gewone breuk waarbij de noemer geen macht van twee is of een onmeetbaar getal. Het effect van L_0 en R_0 is

$$\begin{cases} L_0x = .0b_1b_2b_3b_4b_5... \\ R_0x = .1b_1b_2b_3b_4b_5... \end{cases} \tag{4}$$

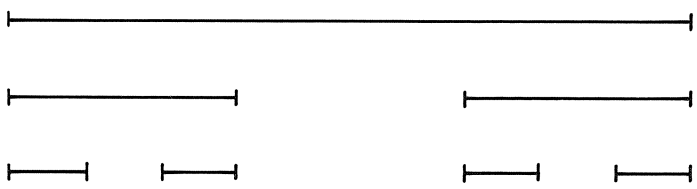
dus het opschuiven van de rij binalen naar rechts met toevoeging van een nul of één.

In het programma STOFT is deze methode gevolgd voor de kromme van Levy. In dit geval worden van die kromme alleen punten afgebeeld. In een eindig aantal stappen p hangen ze als los zand, stof, aan elkaar. Pas wanneer $p\to\infty$ wordt samenhang verkregen en vormen ze met elkaar een oneindigvoudig kronkelende continue lijn. In het programma gaan we p niveau's diep wat wil zeggen dat met (1/2,1/2) als beginpunt $1+2+2^2+2^3+...+2^p$

punten worden afgebeeld. Voegen we de dekpunten (0,0) en (1,0) er aan toe dan zijn dat dus $2^{p+1}+1$ punten. Bij elk punt behoort een rangnummer n dat in het programma in binalen ontbonden wordt. Afhankelijk van de volgorde $b_1b_2b_3...$ wordt of de transformatie L of R uitgevoerd. In het programma zijn L en R door (2) gegeven, maar ze kunnen zonder moeite door algemenere transformaties vervangen worden. Het programma is nog wel voor verbetering vatbaar, we komen er later op terug.

3. Cantor en Koch

Aan het wiskundig rariteitenkabinet ontlenen we twee merkwaardige figuren die in de tijd van hun ontstaan, omstreeks 1900, veel opzien baarden. De eerste is de puntenverzameling van Cantor. De meetkundige constructie, in oneindig veel stappen, is uiterst eenvoudig. In elke stap worden van de aanwezige lijnsegmenten (lijnstuk inclusief eindpunten) de middelste derde intervallen (lijnstuk zonder eindpunten) weggenomen (zie fig.7).



Figuur 7

De constructie kan in getallen worden uitgedrukt door te beginnen met het gesloten interval [0,1]. Bij de eerste stap verwijderen we het open interval (1/3,2/3), bij de tweede stap verdwijnen (1/9,2/9) en (7/9,8/9). Wanneer we de getallen in het drietallig talstelsel schrijven

$$x = .c_1c_2c_3c_4c_5...$$

met $c_k = 0,1$ of 2, kunnen we beter overzien wat er gebeurt. Bij de eerste stap verdwijnen alle getallen waarvoor $c_1 = 1$, bij de tweede stap de getallen met $c_2 = 1$ en zo gaat het verder. Tenslotte blijven alle getallen over waarvoor $c_k = 0$ of 2. Ook het randpunt $x = 1$ hoort er bij omdat $1 = .22222...$, een oneindig voortlopende periodieke breuk. De pvz. van Cantor bestaat uit oneindig veel losse punten, of getallen, en heeft een aantal bijzondere eigenschappen: elk punt is limietpunt en omgekeerd, en het aantal elementen is onaftelbaar oneindig.

De pvz. van Cantor voldoet aan de twee gelijkvormigheidstransformaties

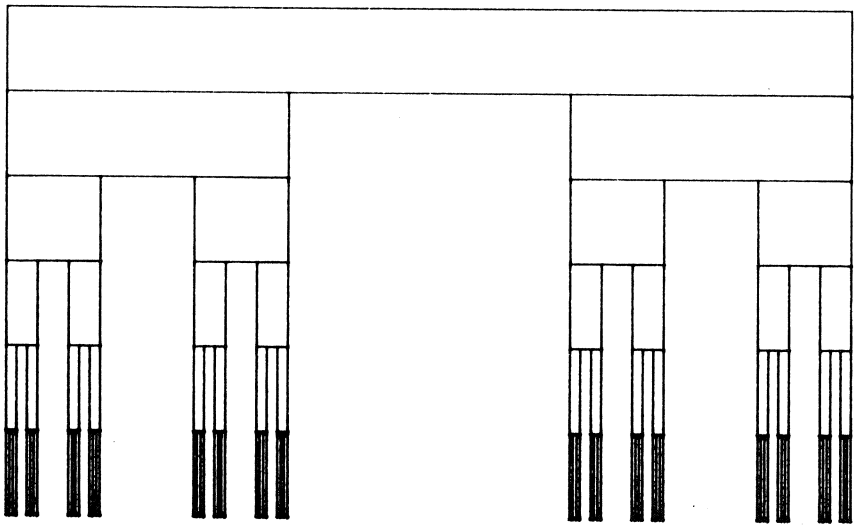
$$L: x' = x/3, \quad R: x' = x/3+2/3$$

die maar heel weinig van (3) verschillen. Het effect op $x = .c_1c_2c_3c_4c_5...$ is

$$\begin{cases} Lx = .0c_1c_2c_3c_4c_5... \\ Rx = .2c_1c_2c_3c_4c_5... \end{cases} \tag{6}$$

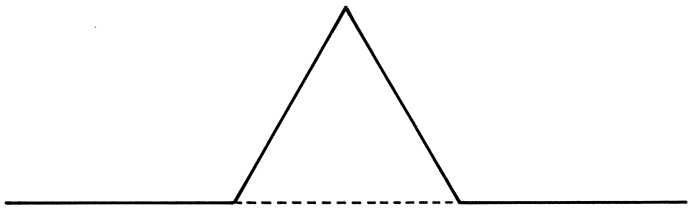
praktisch hetzelfde als (4).

Door L, R en de daaruit te vormen combinaties is de pvz. van Cantor volledig vastgelegd. Tegenwoordig interpreteren we die pvz. als de invariante pvz. voor de twee, zeer eenvoudige, transformaties (6) en we zeggen dat de pvz. een, zeer eenvoudige, fractal is. De parallel tussen wat we in het drietallig stelsel doen, en wat we volgens (2) in het tweetallig stelsel doen, geeft o.a. aanleiding tot de volgende constatering. We vormen een willekeurig oneindig patroon van nullen en enen en interpreteren het als een oneindig voortlopende binaire breuk $x = .b_1 b_2 b_3 b_4 b_5 \ldots$. Alle getallen x vullen het interval [0,1] op continue wijze op. Maar nu interpreteren we de breuk in het drietallig talstelsel! De getallen x vormen nu precies de pvz. van Cantor, verkleind met factor 2, een oneindigheid van losse punten, een z.g. discontinuum. Het is rekentechnisch gesproken vrij lastig een goede illustratie van de Cantor pvz. te maken, maar met een trucje kunnen we er een kam van maken als in fig.8. Uiteraard is dit ook weer een benadering, want in werkelijkheid is de kam oneindig lang met zich alsmaar verfijnende tanden.



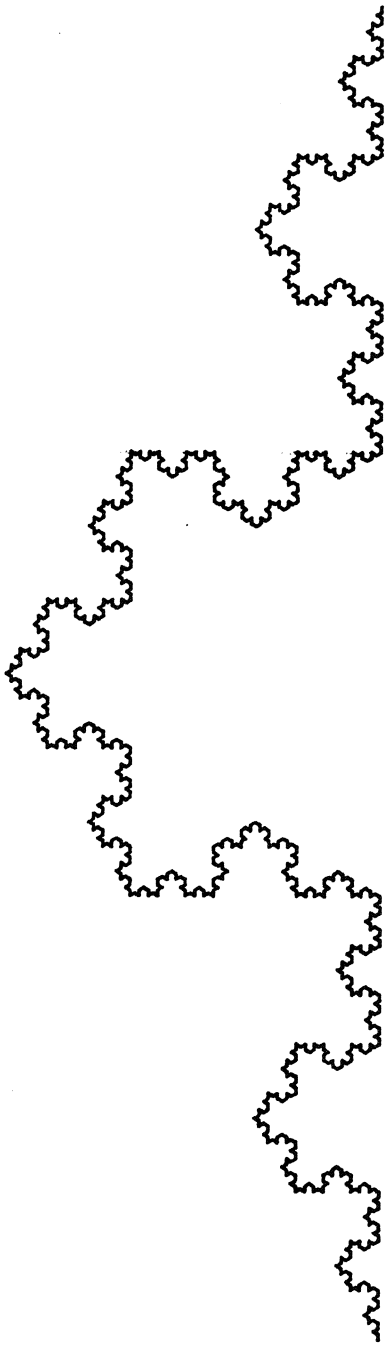
Figuur 8

De constructie van Helge von Koch lijkt erg op die van Cantor. Ook hierbij wordt van een lijnstuk het middelste derde gedeelte weggenomen, maar nu wordt het gat opgevuld met opstaande zijden van een gelijkzijdige driehoek als geschetst in fig.9.



Figuur 9

Wanneer we die constructie enige malen herhalen krijgen we fig.10. In gedachten kunnen we steeds verder gaan. Het resultaat is een continue kronkellijn die nergens een raaklijn heeft en waarvan elk onderdeel, hoe klein ook, gelijkvormig is met het geheel. Een echte fractal zeggen we tegenwoordig. Het is niet moeilijk om vast te stellen dat de Koch fractal vol zit met Cantor verzamelingen en dat de Koch fractal aan dezelfde fundamentele transformaties (5) voldoet. Helaas kunnen we niet zeggen dat de Koch fractal door die twee transformaties ondubbelzinnig bepaald is. Dat schept natuurlijk een probleem, maar dat zal verderop opgelost worden. Eerst moeten we iets weten over de voor L en R in aanmerking komende gelijkvormigheidstransformaties.



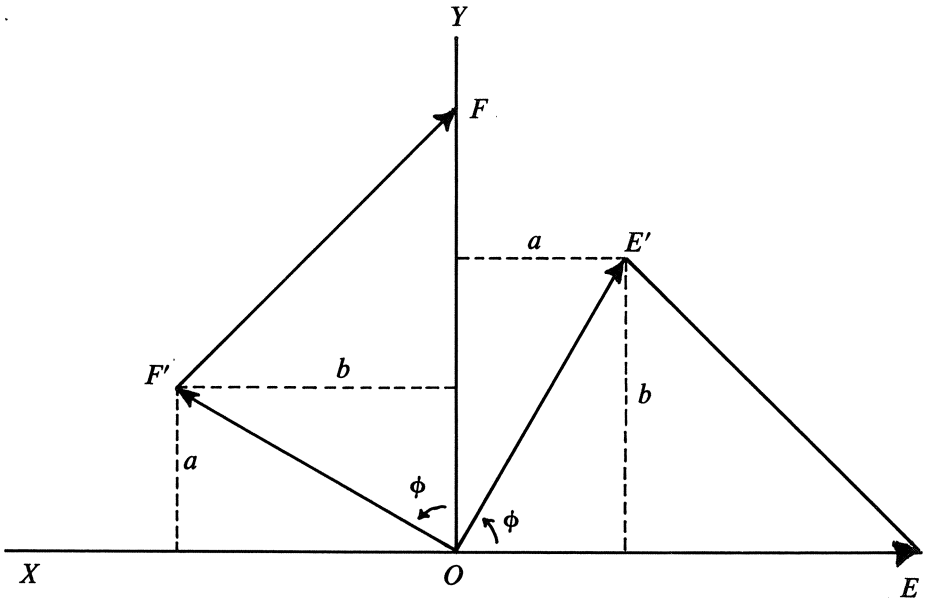
Figuur 10

4. DRAAIVERMENIGVULDIGING

De eenvoudigste manier om de formules van een draaivermenigvuldiging in coördinatenmeetkunde af te leiden, is om te poneren dat een gelijkvormigheidstransformatie een bijzonder geval is van een lineaire transformatie

$$\begin{cases} x' = a_0 + a_1 x + a_2 y \\ y' = b_0 + b_1 x + b_2 y. \end{cases} \tag{1}$$

Bij de draaivermenigvuldiging is er precies één punt dat op zijn plaats blijft. Passen we het coördinatenstelsel zo aan dat de oorsprong het rotatiecentrum, het dekpunt, is, dan is alvast $a_0 = b_0 = 0$. De waarden van de overige coëfficiënten kunnen we aflezen uit fig.11.



Figuur 11

De punten E en F zijn de eindpunten van de eenheidsvectoren langs de coördinaatassen, d.w.z. $E(1,0)$ en $F(0,1)$. De draaivermenigvuldiging om O brengt ze in de posities E' en F'. Heeft E' de coördinaten a en b dan heeft F' de coördinaten $-b$ en a. Dus

$$(1,0) \to (a,b) \quad \text{en} \quad (0,1) \to (-b,a).$$

Dit gesubstitueerd in

$$x' = a_1 x + a_2 y \text{ en } y' = b_1 x + b_2 y$$

geeft meteen

$$a_1 = a,\ b_1 = b,\ a_2 = -b,\ b_2 = a.$$

Een draaivermenigvuldiging om de oorsprong wordt dus beschreven door

$$\begin{cases} x' = ax - by \\ y' = bx + ay. \end{cases} \tag{8}$$

Uit de figuur lezen we af dat de schaalfactor s volgt uit $s = OE'/OE = \sqrt{a^2+b^2}$. Is ϕ de rotatiehoek dan is

$$a = s\cos\phi,\ b = s\sin\phi.$$

Formules als (8) behoren tot het basismateriaal voor de computermeetkunde. Met de rekenwijze van de complexe getallen

$$z = x + iy,\ c = a + ib,\ i^2 = -1$$

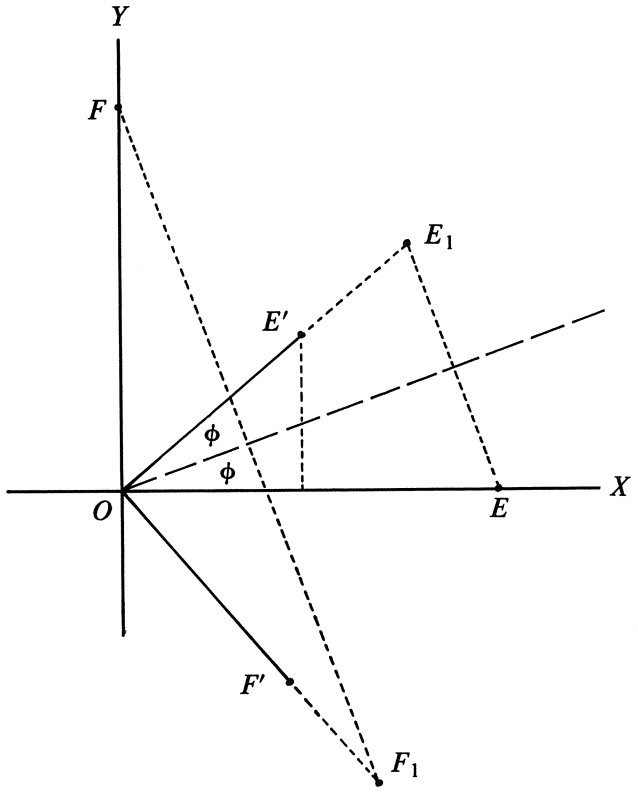
kunnen we (8) veel eenvoudiger schrijven als

$$z' = cz. \tag{8a}$$

Dit wijst er al op hoe nuttig de complexe rekenwijze kan zijn bij meetkundige beschouwingen. Nu tegenwoordig weer meer aandacht aan de meetkunde gegeven wordt zou ook de 'meetkunde van de complexe getallen' als vast onderdeel in het onderwijsprogramma opgenomen moeten worden.

5. Spiegelen

Er zijn twee typen gelijkvormigheidstransformaties, de directe en de indirecte. Bij de directe blijft de oriëntatie behouden, bij de indirecte gaat linksom over in rechtsom. Een indirecte gelijkvormigheidstransformatie kunnen we altijd opvatten als het produkt van een gewone lijnspiegeling en een centrale vermenigvuldiging of translatie. Zien we af van speciale gevallen als de schuifspiegeling, dan kunnen we de situatie beschrijven aan de hand van fig.12 waarbij de oorsprong het dekpunt van de transformatie is. De spiegelingsas gaat door de oorsprong en maakt een hoek ϕ met de positieve X-as.



Figuur 12

Net als bij de afleiding van (8) letten we op de transformatie van de eenheidsvectoren OE en OF. Na transformatie krijgen we OE' en OF' en natuurlijk is $|OE'| = |OF'|$ en $OE' \perp OF'$. Heeft E' de coördinaten (a,b) dan volgen hieruit voor F' de coördinaten $(b, -a)$. Dus

$$(1,0) \rightarrow (a,b) \quad \text{en} \quad (0,1) \rightarrow (b, -a).$$

De beschrijving wordt daarmee

$$\begin{cases} x' = ax + by \\ y' = bx - ay, \end{cases} \tag{9}$$

of in complexe notatie eenvoudigweg

$$z' = c\bar{z}. \tag{9a}$$

Is s de schaalfactor OE'/OE dan kunnen we a en b schrijven als

$$a = s \cos 2\phi \text{ en } b = s \sin 2\phi$$

zodat b.v. $s = \sqrt{a^2 + b^2}$. Voor $s < 1$ is de transformatie een contractie en heet

een *krimpspiegeling*, voor $s>1$ kunnen we van een rekspiegeling spreken.

Voeren we de transformatie (9) tweemaal achter elkaar uit dan is het resultaat een gewone centrale vermenigvuldiging. Dat kunnen we zowel meetkundig bewijzen als met (9) laten zien. Uit (9a) volgt de bewering meteen uit

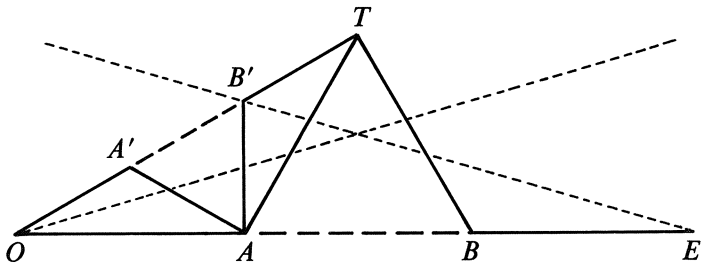
$$z'' = c\bar{z}'$$

zodat

$$z'' = c\bar{c}z.$$

De bespiegeling van Koch

We zijn nu in staat te doorzien dat ook de kromme van Von Koch opgevat kan worden als een door twee transformaties bepaalde fractal. In fig.13 is de eerste fase van de constructie geschetst waarbij het lijnstuk OE vervangen wordt door de gebroken lijn $OATBE$. L is een krimpspiegeling waarvan de as door O gaat en een hoek van 15^o maakt met OA.



Figuur 13

De reductiefactor is $OT/OE = 1/\sqrt{3}$ zodat L de lijn $OATBE$ overvoert in $OA'AB'T$. R is de overeenkomstige krimpspiegeling waarbij de as door E gaat. Duidelijk is dat L^2 en R^2 centrale vermenigvuldigingen t.o.v. O en E met factor 1/3 zijn, In formulevorm:

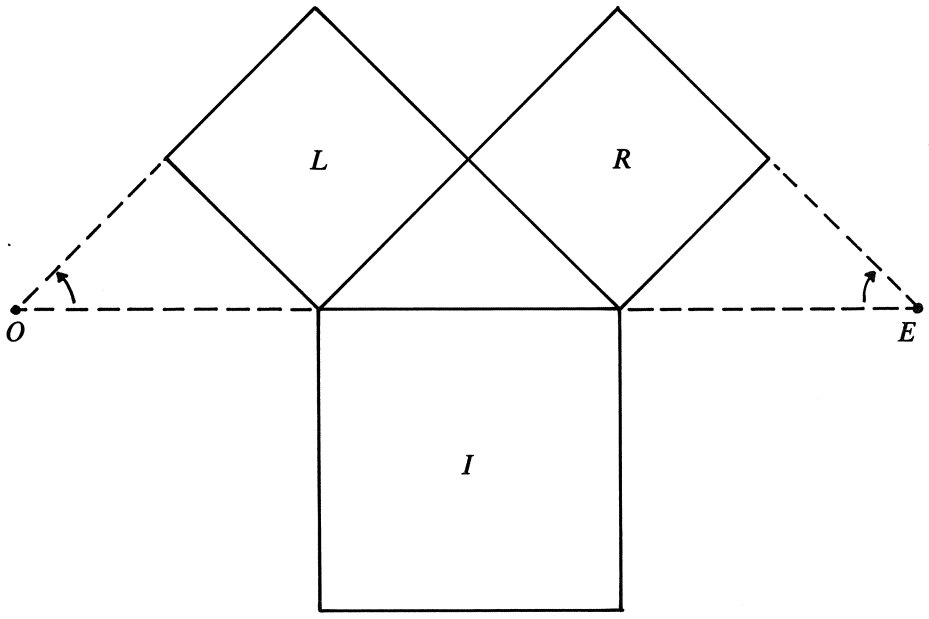
$$L\begin{cases} x' = .5x + .289y \\ y' = .289x - .5y \end{cases} \qquad R\begin{cases} x' = .5x - .289y + .5 \\ y' = -.289x - .5y + .289. \end{cases}$$

Met de notatie van de complexe getallen ziet het er weer eenvoudiger uit, b.v.

$$L: z' = (\frac{1}{2} + \frac{i}{2\sqrt{3}})\bar{z}.$$

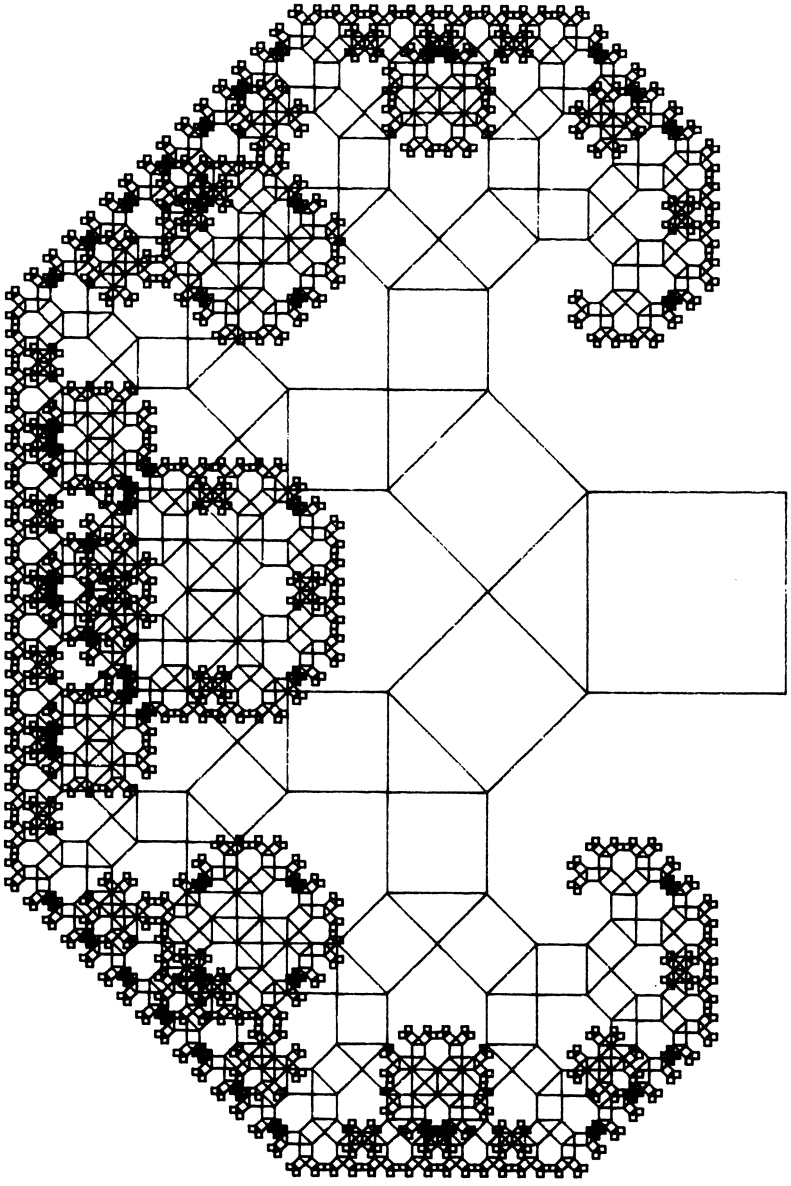
Door L en R wordt de Koch fractal punt voor punt opgebouwd wanneer we als beginpunt O of E kiezen, maar op het beeldscherm van onze computer voegen enige duizenden punten zich al aaneen tot een voor het oog continue kromme.

De boom van Pythagoras
Wanneer we de transformaties (2) loslaten op een vierkant als geschetst in fig.14 krijgen we de bekende boom van Pythagoras van Bosman, indien we alle vierkanten na transformatie laten staan.



Figuur 14

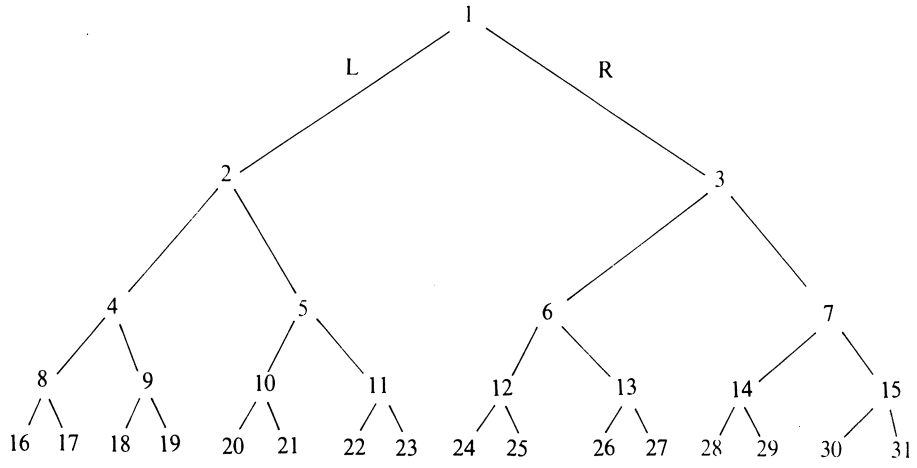
Wanneer die constructie oneindig vaak voortgezet wordt ontstaat er een limietfiguur van tot een punt ineengeschrompelde vierkantjes welke na afsluiting (het eraan toevoegen van de limietpunten) identiek is met de kromme van Levy! In fig.15 is een computerbenadering van de Pythagorasboom afgebeeld.



Figuur 15

6. BOEKHOUDING

Nog eens, L en R zijn contraherende transformaties die we in alle combinaties laten werken op een beginpunt P. De uit P afgeleide punten kunnen we nummeren en berekenen volgens de in fig.16 afgebeelde tweetallige boom.

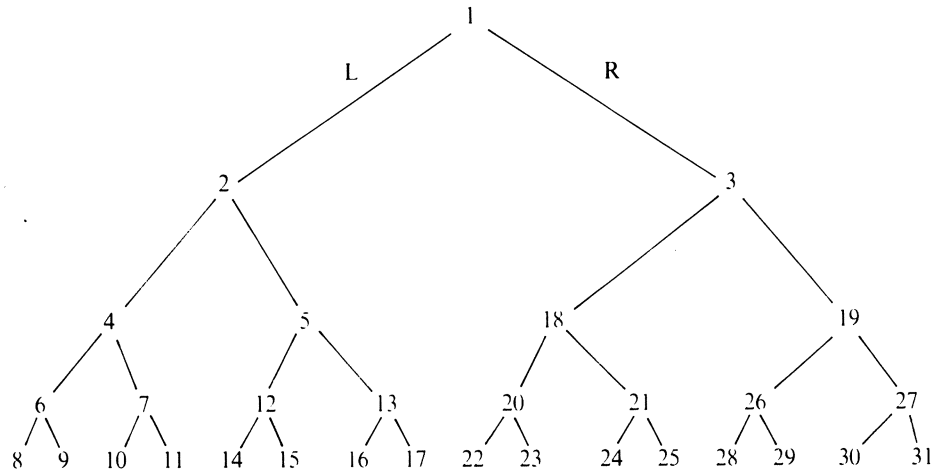


Figuur 16

Aldus is b.v. $21 = RLRL\,1$.

Bij deze methode worden de coördinaten van de punten laag voor laag berekend. In de boekhouding moeten we kunnen beschikken over de gegevens van alle punten van de vorige laag en die kunnen behoorlijk aangroeien. In de praktijk blijkt dit vaak tot een te grote belasting van het computergeheugen te leiden zodat het de moeite loont een andere techniek te bedenken. Heel eenvoudig en doeltreffend is de z.g. Monte Carlo methode. Het principe is als volgt. Is P een (zojuist geconstrueerd) punt van de fractal dan is het volgende punt LP of RP waarbij de keuze L of R toevallig is. In het computerprogramma gebruiken we daarbij de RND-opdracht welke een toevallig getal uit (0,1) levert. Die methode is verwerkt in het programma STOF een heel eenvoudig en heel flexibel programma. De Monte Carlo techniek geeft aanleiding hier — in het onderwijs — wat aandacht aan elementaire kansrekening te besteden. Een heel fraaie systematische methode berust op de nummering van fig.17. Willen we 12 niveau's diep gaan dan, vereist dit slechts $4 \times 12 = 48$ geheugenplaatsen. Deze methode is verwerkt in het computerprogramma STOFB. Ook dit programma is buitengewoon flexibel omdat het de mogelijkheid in zich heeft voor L en R verschillende keuzen te maken. Het is

een van de meest gebruikte programma's om fractals te maken die gebouwd zijn volgens een binaire boomstructuur. De beschreven methode staat bekend als 'backtracking'.



Figuur 17

Voorbeelden
De onderstaande getallenvoorbeelden hebben betrekking op de programma's STOF, STOFT of STOFB. Daarbij zijn L en R hetzij een draaivermenigvuldiging hetzij een krimpspiegeling:

$$L\begin{cases} x' = ax \mp by \\ y' = bx \pm ay \end{cases} \qquad R\begin{cases} x' = cx \mp dy + 1 - c, \\ y' = dx \pm cy - d. \end{cases}$$

1. Twee rotaties
   $a = b = c = .5, \quad d = -.5.$
   $a = b = c = d = .5.$
   $a = .6, \quad b = .45, \quad c = .5, \quad d = 0.$
   $a = 0, \quad b = c = .7, \quad d = 0.$

2. Rotatie en spiegeling
   $a = 2/3, \quad b = 0, \quad c = .5, \quad d = -.2887$
   $a = -.3, \quad b = .5196, \quad c = a, \quad d = -b$
   $a = .4, \quad b = .4, \quad c = a, \quad d = -b$

3. Twee spiegelingen
   $a=.5,\quad b=.2887,\quad c=a,\quad d=-b.$
   $a=0,\quad b=.64,\quad c=a,\quad d=-b.$
   $a=b=c=.5,\quad d=0.$

7. Generalisaties

Wiskundigen dienen zich af te vragen of iets algemener kan, gegeneraliseerd kan worden. Maar toch moet daarbij niet vergeten worden dat 'eenvoud het kenmerk van het ware is' (simplex sigillum veri). Zo kan de hier besproken methode gegeneraliseerd worden tot 'de invariante verzameling van m contraherende afbeeldingen in een metrische ruimte'. Meer concreet en met de computer technisch uitvoerbaar, is de bestudering van ruimtelijke fractals als vrucht van drie ruimtelijke gelijkvormigheidstransformaties. Hier echter beperken we ons tot een generalisatie in het platte vlak, waarbij de transformaties L en R conforme afbeeldingen zijn, d.w.z. gelijkvormigheidstransformaties met een rotatiehoek en verkortingsfactor die afhankelijk van de plaats zijn. Bovendien beschouwen we maar een enkel voorbeeld, maar dat voorbeeld is zo interessant dat er met gemak een boek mee te vullen is. Er is trouwens al veel over geschreven en het fraaie boek *The Beauty of Fractals* van Peitgen en Richter is er voor een groot deel aan gewijd. De formulering in complexe notatie is heel eenvoudig:

$$\begin{cases} L: z' = \sqrt{z-c} \\ R: z' = -\sqrt{z-c} \end{cases} \tag{10}$$

L en R zijn dus de twee inversen van de kwadratische afbeelding

$$z' = z^2+c,$$

of

$$\begin{cases} x' = x^2-y^2+a \\ y' = 2xy+b, \end{cases}$$

in reële schrijfwijze.
Voor elke combinatie van a en b wordt door (10) een fractal bepaald. Dergelijke fractals worden tegenwoordig naar Julia genoemd, een Frans wiskundige die in 1919 de grondslagen vestigde van iteratieve complexe afbeeldingen van de vorm $z_{n+1}=f(z_n)$. Met het bijgevoegde programma JULIAB is men in staat op een microcomputer een, meestal, redelijke benadering van de door (10) bepaalde Julia fractal te verkrijgen. Het programma is niets anders dan de uitwerking van (10) in reële notatie met een op de backtrackmethode van fig.16 berustende boekhouding.

Appendix

```
10 REM ***PYTHAGORASLIJN***
20 REM ***NAAM:LEVY***
30 SCREEN 3 : CLS : PI=3.141593
40 WINDOW (-.5, -1.2)-(1.5,.3)
50 P=12 : REM ***KEUZE ORDE***
60 H=2^(-(P/2)) : A=H*COS(P*PI/4 ): B=H*SIN(P*PI/4)
70 LINE (0,0)-(A,-B) : LINE -(A+B,A-B)
80 X=1 :Y=1
90 FOR N=2 TO 2^P-1
100 M=N : S=1
110 IF M MOD 2=1 THEN S=S+1
120 M=M \2
130 IF M>1 THEN GOTO 110
140 IF S MOD 4=0 THEN X=X+1
150 IF S MOD 4=1 THEN Y=Y+1
160 IF S MOD 4=2 THEN X=X-1
170 IF S MOD 4=3 THEN Y=Y-1
180 LINE -(A*X+B*Y,A*Y-B*X)
190 NEXT N : BEEP : A$=INPUT$(1) : END
```

```
10 REM ***STOFFRACTAL IN TWEETALLIGE OPBOUW MET P NIVEAUS***
20 REM ***ER WORDEN 2^(P+1)+1 PUNTEN AFGEBEELD***
30 REM ***NAAM:STOFT***
40 SCREEN 3 : CLS
50 WINDOW (-.7,-.7)-(1.7,1.1)
60 P=12 : DIM A(P) : REM ***KEUZE ORDE***
70 A=.5 : B=.5 : C=.5 : D=-.5 : REM ***KEUZE PARAMETERS***
80 PSET (0,0) : PSET (1,0)
90 FOR M=0 TO P
100 FOR N=2^M TO 2^(M+1)-1 : L=N
110 FOR K=1 TO M
120 A(K)=L MOD 2 : L=INT(L/2) : NEXT K
130 X=A : Y=B
140 FOR J=1 TO M
150 IF A(J)=0 THEN GOSUB 180 ELSE GOSUB 190
160 NEXT J : PSET (X,Y) : NEXT N : NEXT M
170 BEEP : END
180 Z=X : X=A*X-B*Y : Y=B*Z+A*Y : RETURN : REM ***ROTATIE***
190 Z=X : X=C*X-D*Y+1-C : Y=D*Z+C*Y-D : RETURN : REM ***ROTATIE***
200 END
```

```
10 REM ***CANTOR KAM***
20 REM ***NAAM:KAM***
30 SCREEN 3 : CLS
40 WINDOW (-.3,-.4)-(1.3,.8)
50 DIM A(729),B(729) : A(0)=0 : A(1)=1
60 B=.1 : LINE (0,0)-(1,0) : LINE -(1,-B) : LINE -(0,-B) : LINE -(0,0)
70 FOR P=1 TO 6
80 FOR I=0 TO 2^P-1
90 B(I)=A(I)/3 : B(I+2^P)=1-(1-A(I))/3
100 NEXT I
110 FOR J=1 TO 2^(P+1)-1
120 A(J)=B(J) : NEXT J
130 FOR K=0 TO 2^(P+1)-1 STEP 2
140 LINE (A(K),B*P)-(A(K+1),B*P)
150 LINE (A(K),B*P)-(A(K),B*P-B) : LINE (A(K+1),B*P)-(A(K+1),B*P-B)
160 NEXT K : NEXT P : END
```

```
10 REM ***PYTHAGORASBOOM , BACKTRACKMETHODE***
20 REM ***NAAM:BOOMP***
30 SCREEN 3 : CLS : PI=3.141593
40 WINDOW (-5,-3)-(5,4.5)
50 P=12 : DIM X1(P),Y1(P),X2(P),Y2(P),U1(P),V1(P),U2(P),V2(P)
60 F=PI/5 : C=COS(F) : S=SIN(F) : REM ***KEUZE HOEK F***
70 A1=-C*S : A2=C^2 : B1=A1+A2 : B2=-A1+A2
80 C1=B2 : C2=1-B1 : D1=1-A1 : D2=1-A2
90 X1(0)=0 : Y1(0)=0 : U1(0)=1 : V1(0)=0
100 LINE (0,0)-(0,-1) : LINE -(1,-1) : LINE -(1,0)
110 S=1 : GOSUB 170
120 FOR M=1 TO 2^(P-1)-1 : S=P : N=M
130 IF N MOD 2=0 THEN N=N\2 : S=S-1 : GOTO 130
140 GOSUB 150 : NEXT M : END
150 X1(S-1)=X2(S-1) : Y1(S-1)=Y2(S-1)
160 U1(S-1)=U2(S-1) : V1(S-1)=V2(S-1)
170 FOR J=S TO P
180 X=X1(J-1) : Y=Y1(J-1) : U=U1(J-1) : V=V1(J-1)
190 X3=U-X : Y3=V-Y
200 X1(J)=X+A1*X3-A2*Y3 : Y1(J)=Y+A2*X3+A1*Y3
210 U1(J)=X+B1*X3-B2*Y3 : V1(J)=Y+B2*X3+B1*Y3
220 X2(J)=X+C1*X3-C2*Y3 : Y2(J)=Y+C2*X3+C1*Y3
230 U2(J)=X+D1*X3-D2*Y3 : V2(J)=Y+D2*X3+D1*Y3
240 LINE (X,Y)-(X1(J),Y1(J)) : LINE -(U1(J),V1(J)) : LINE -(U,V)
250 LINE -(X,Y) : LINE-(X2(J),Y2(J)) : LINE -(U2(J),V2(J)) : LINE -(U,V)
260 NEXT J : RETURN : END
```

```
10 REM ***STOFFRACTAL , MONTE CARLO METHODE***
20 REM ***NAAM:STOF***
30 SCREEN 3 : CLS : PI=3.141593 : RANDOMIZE 100
40 WINDOW (-1.1,-1.2)-(2.1,1.2)
50 R1=.6 : A=R1*COS(2*PI/3) : B=R1*SIN(2*PI/3)
60 R2=.6 : C=R2*COS(2*PI/3) : D=-R2*SIN(2*PI/3)
70 X=A : Y=B : REM ***KEUZE BEGINPUNT***
80 FOR K=1 TO  10000
90 IF RND <.5 THEN GOSUB 120 ELSE GOSUB 140
100 PSET (X,Y)
110 NEXT K : END
120 Z=X : X=A*X-B*Y : Y=B*Z+A*Y : REM ***ROTATIE***
130 RETURN
140 Z=X : X=C*X-D*Y+1-C : Y=D*Z+A*Y-D : REM ***ROTATIE***
150 RETURN
160 Z=X : X=A*X+B*Y : Y=B*Z-A*Y : REM ***SPIEGELING***
170 RETURN
180 Z=X : X=C*X+D*Y+1-C : Y=D*Z-A*Y-D : REM ***SPIEGELING***
190 RETURN : END
```

```
10 REM ***STOFFRACTAL BACKTRACK METHODE***
20 REM ***NAAM:STOFB***
30 SCREEN 3 : CLS : PI=3.141593
40 A=.6 : B=.45 : C=.5 : D=0 : P=12 : WINDOW (-.1 ,-.3)-(1.1,.6)
50 DIM X1(P),Y1(P),X2(P),Y2(P) : PSET (0,0) : PSET (1,0)
60 X1(0)=A : Y1(0)=B : PSET (X1(0),Y1(0))
70 S=1 : GOSUB 130
80 FOR M=1 TO 2^(P-1)-1 : S=P : N=M
90 IF N MOD 2=0 THEN N=N\2 : S=S-1 : GOTO 90
100 GOSUB 120 : NEXT M : BEEP
110 A$=INPUT$(1) : END
120 X1(S-1)=X2(S-1) : Y1(S-1)=Y2(S-1)
130 FOR J=S TO P
140 X=X1(J-1) : Y=Y1(J-1)
150 'X1(J)=A*X-B*Y : Y1(J)=B*X+A*Y : REM ***ROTATIE***
160 X1(J)=A*X+B*Y : Y1(J)=B*X-A*Y : REM ***SPIEGELING***
170 'X2(J)=C*X-D*Y+1-C : Y2(J)=D*X+C*Y-D : REM ***ROTATIE***
180 X2(J)=C*X+D*Y+1-C : Y2(J)=D*X-C*Y-D : REM ***SPIEGELING***
190 PSET (X1(J),Y1(J)) : PSET (X2(J),Y2(J))
200 NEXT J : RETURN : END
```

```
10 REM ***JULIA FRACTAL VAN Z:=Z^2+C , BACKTRACK METHODE***
20 REM ***NAAM:JULIAB***
30 SCREEN 3 : CLS
40 WINDOW (-2,-1.5)-(2,1.5)
50 P=12 : DIM X1(P),Y1(P),X2(P),Y2(P)
60 A=0 : B= 1
70 X1(0)=-1.3002 : Y1(0)=.6248
80 PSET (X1(0),Y1(0)) : S=1 : GOSUB 130
90 FOR M=1 TO 2^(P-1)-1 : S=P : N=M
100 IF N MOD 2 =0 THEN N=N\2 : S=S-1 : GOTO 100
110 GOSUB 120 : NEXT M : END
120 X1(S-1)=X2(S-1) : Y1(S-1)=Y2(S-1)
130 FOR J=S TO P
140 X=X1(J-1) : Y=Y1(J-1)
150 R=SQR((X-A)^2+(Y-B)^2)/2 : T=(X-A)/2
160 X1(J)=SQR(R+T) : X2(J)=-X1(J)
170 Y1(J)=SQR(R-T)*SGN(Y-B) : Y2(J)=-Y1(J)
180 PSET (X1(J),Y1(J)) : PSET (X2(J),Y2(J))
190 NEXT J : RETURN : END
```

Personal Computer en Getaltheorie

J. van de Lune
Centrum voor Wiskunde en Informatica
Kruislaan 413, 1098 SJ Amsterdam

0. Inleiding

Ook al valt de getaltheorie zo goed als geheel buiten het kader van het middelbaar onderwijs, het is stellig een bijzonder geschikt onderwerp bij het actief leren omgaan met computers en programma's. Immers, dit onderdeel van de wiskunde is uiterst rijk aan zowel eenvoudige als interessante thema's, waar men zelf (ook als beginner) vrij gemakkelijk programmatuur bij kan schrijven. Een niet onbelangrijke educatieve 'bijkomstigheid' van het daadwerkelijk uitvoeren van gericht numeriek werk is verder dat het de interesse in meer zuiver theoretische beschouwingen in sterke mate kan bevorderen. In de volgende paragrafen zullen we trachten het een en ander toe te lichten aan de hand van een aantal eenvoudige programma's (in GWBASIC). Al deze programma's zijn getest op een Olivetti M24.

Het spreekt vanzelf dat we in een korte voordracht als deze geen theorie in extenso kunnen behandelen. Vandaar dat we hieronder een aantal eenvoudige inleidende boeken over getaltheorie vermelden die dienst kunnen doen als 'naslagwerk'.

De gegeven programma's hebben slechts een illustratief karakter. De geboden stof dient veel meer opgevat te worden als een aantal suggesties die men zelf nader uit dient te werken.

Opmerking. Om de executiesnelheid van een GWBASIC programma op te voeren, doet men er goed aan de in het programma voorkomende variabelen een zo kort mogelijke naam te geven en alle commentaar-regels (REM statements) weg te laten.

Eenvoudige inleidingen tot de getaltheorie:

1. J. AGNEW (1972). *Explorations in Number Theory,* Brooks/Cole.
2. A.H. BEILER (1966). *Recreations in the Theory of Numbers* (The Queen of Mathematics Entertains), Dover Publications.
3. D.M. BURTON (1980). *Elementary Number Theory,* Allyn & Bacon.
4. H. DAVENPORT (1952). *The Higher Arithmetic* (An Introduction to the Theory of Numbers), Hutchinson's University Library.
5. J.E. MAXFIELD, M.W. MAXFIELD (1972). *Discovering Number Theory,* Saunders.
6. O. ORE (1948). *Number Theory and its History,* McGraw-Hill.
7. J.V. USPENSKY, M.A. HEASLET (1939). *Elementary Number Theory,* McGraw-Hill.
8. P. WIJDENES (1937). *Beginselen van de Getallenleer,* Noordhoff.

1. DEELBAARHEID

Met 'getal' bedoelen we in het vervolg 'geheel getal' (element van $\mathbb{Z}$), tenzij expliciet anders vermeld is. Een natuurlijk getal is een positief geheel getal.

DEFINITIE. Het getal a heet een deler van het getal b als de vergelijking

$$ax = b \tag{1.1}$$

oplosbaar is (met $x \in \mathbb{Z}$). Notatie: $a \mid b$. Als hierbij $a \in \mathbb{N}$ dan noemen we a een natuurlijke deler van $b \in \mathbb{Z}$.

Als $a \neq 0$ is dan is $a \mid b$ equivalent met het geheel zijn van $\frac{b}{a}$.

Programma 1 (DELER)

```
10 'Programma 1 (DELER)
20 CLS:PRINT"We testen of A een deler is van B"
30     PRINT"................................":PRINT
40 INPUT"Voer A in";A
50 INPUT"Voer B in";B
60 IF A=INT(A) AND B=INT(B) GOTO 80
70 BEEP:PRINT:PRINT"Foute input !":PRINT:GOTO 40
80 Q=B/A:IF Q=INT(Q) GOTO 100
90 PRINT"A =";A;"is GEEN deler van B =";B:GOTO 110
100 PRINT"A =";A;"is een DELER van B =";B
110 END
```

Opgave. Beveilig Programma 1 tegen de input $a = 0$, rekening houdend met het feit dat toch $0 \mid 0$.

Opgave. Maak Programma 1 geschikt voor DP (Double Precision) arithmetiek.

Van fundamenteel belang in de getaltheorie is de observatie dat bij elk tweetal getallen a en b, met $b \neq 0$, een (uniek) paar getallen q en r bestaat zodanig dat

$$a = q|b| + r \text{ met } 0 \leqslant r < |b|. \tag{1.2}$$

Als $r = 0$ dan is $b|a$. (Door zich strikt te houden aan de veronderstelling $b \neq 0$, is bij menig schrijver 0 geen deler van 0.)

Programma 2 (QR)

```
10 'Programma 2 (QR)
20 INPUT"Voer A in";A:INPUT"Voer B in";B:GOSUB 60
30 PRINT"Q =";Q,"R =";R:GOTO 20
40 'SUBROUTINE. In: A,B. Out: Q,R.
50 'Bepaling van Quotient (Q) en Rest (R) bij deling van A door B.
60 IF B=0 THEN PRINT:PRINT"Foute input ! B =";B:PRINT:BEEP:GOTO 20
70 Q=INT(A/B):R=A-B*Q:RETURN
```

2. Priemgetallen

Definitie. Onder een priemgetal verstaan we een natuurlijk getal p dat groter dan 1 is en alleen 1 en p als natuurlijke delers heeft.

Definitie. Een getal s heet samengesteld als $|s| > 1$ is en een (echte) deler heeft die strikt tussen 1 en $|s|$ ligt.

Merk op dat een samengesteld natuurlijk getal n altijd een deler d heeft met de eigenschap

$$1 < d \leqslant \sqrt{n} \tag{2.1}$$

en dat de kleinste deler $d > 1$ van een samengesteld getal altijd een priemgetal is. De samengesteldheid van een natuurlijk getal is met behulp van deze criteria 'gemakkelijk' aan te tonen: men gaat na of $n > 1$ is en of n een priemdeler d heeft die aan (2.1) voldoet.

Een getal $p > 1$ is priem als het niet samengesteld is. Op deze overwegingen is de volgende priemtest gebaseerd.

Programma 3 (PRIEMTST)

```
10 'Programma 3 (PRIEMTST)
20 CLS:PRINT"PRIEMTEST":PRINT
30 DEFDBL A-Z        'Dit is slechts een optie.
40 BEEP:INPUT"Voer een NATUURLIJK getal N > 1 in";N:PRINT
50 'We testen eerst of N wel geheel is en > 1.
60 IF N=INT(N) AND N>1 GOTO 70 ELSE PRINT"N =";N;"Foute input !":PRINT:GOTO 40
70 IF N=2 GOTO 130 ELSE D=2:Q=N/D:IF Q=INT(Q) GOTO 120
80 IF N=3 GOTO 130 ELSE D=3:Q=N/D:IF Q=INT(Q) GOTO 120
90 'Ga na door welke getallen D we N vervolgens delen !
100 D=1:I=2:M=INT(SQR(N+.5))
110 I=6-I:D=D+I:IF D>M GOTO 130 ELSE Q=N/D:IF Q<>INT(Q) GOTO 110
120 PRINT"N=";N;"is SAMENGESTELD met kleinste priemfactor";D:PRINT:GOTO 40
130 PRINT"N=";N;"is PRIEM.":PRINT:GOTO 40
```

Opgave. Test bovenstaand programma uit door alle priemen < 100 te bepalen. Verifieer dat 10000000019 priem is. Hoe lang duurt de test voor dit getal?

Opgave. Verifieer het volgende programma (PRMLIST) dat beoogt alle priemgetallen p groter dan een gegeven n te bepalen.

Opgave. Schrijf een programma dat de priemgetallen genereert met behulp van de zeef van Eratosthenes.

Programma 4 (PRMLIST)

```
10 'Programma 4 (PRMLIST)
20 CLS:PRINT"Lijst van ALLE PRIEMGETALLEN vanaf N.":PRINT
30       'DEFDBL A-Z
40 NR=0:PRINT"Voer een NATUURLIJK getal N > 1 in.":PRINT
50 BEEP:INPUT N:IF N<>INT(N) OR N<=1 THEN PRINT"Foute input !":GOTO 40
60 IF N=2 THEN NR=1:PRINT"N =";N;"is PRIEM.     ( nr =";NR;")":N=N+1:GOTO 120
70 IF 2*INT(N/2)=N THEN N=N+1
80 IF N=3 GOTO 120
90 D=3:IF N=D*INT(N/D) THEN N=N+2
100 D=1:I=2:M=INT(SQR(N+.5))
110 I=6-I:D=D+I:IF D>M GOTO 120 ELSE Q=N/D:IF Q<>INT(Q) GOTO 110 ELSE GOTO 130
120 NR=NR+1:PRINT"N =";N;"is PRIEM.     ( nr =";NR;")"
130 N=N+2:GOTO 90
```

Opgave. Ga na dat $10^{10}+19$ het kleinste priemgetal $> 10^{10}$ is. Maak een schatting van de tijd die nodig is om aan te tonen dat een getal in de buurt van 10^{16} priem is.

DEFINITIE Onder een priemtweeling verstaan we een paar priemgetallen (p,q) met $q = p+2$. Naast p moet dus ook $p + 2$ priem zijn.

Voorbeelden. (5,7), (11,13), (17,19), (29,31). (Het is niet bekend of er oneindig veel van bestaan.)

Opgave. Verifieer het volgende programma dat beoogt alle priemtweelingen te bepalen.

Programma 5 (TWINPRMS)

```
10 'Programma 5 (TWINPRMS)
20 CLS:PRINT"Lijst van TWEELING-PRIEMGETALLEN vanaf 6N - 1":PRINT
30 H=.5:E=1:T=2:Z=6:K=0
40 BEEP:INPUT"We starten met 6 * N - 1. Wat zal N zijn ?";N:PRINT:A=N
50 A=Z*(A-E)-E:B=A+T
60 A=A+Z:B=B+Z
70 G=A:GOSUB 110:IF W=0 GOTO 60 ELSE G=B:GOSUB 110:IF W=0 GOTO 60
80 K=K+E:PRINT"P1 =";A,"P2 =";B,"(paar nr ";K;")":GOTO 60
90 'Subroutine PRIEMTEST (we testen de primaliteit van G)
100 'We behoeven in dit geval NIET te testen op deelbaarheid door 2 en 3.
110 D=E:I=T:M=INT(SQR(G+H))
120 I=Z-I:D=D+I:IF D>M THEN W=1:RETURN
130 Q=G/D:IF Q=INT(Q) THEN W=0:RETURN ELSE GOTO 120
```

3. KETTINGBREUKEN; PRECISIE

Zij α een reëel getal. We definiëren

$$a_0 := [\alpha] \text{ en } \rho_1 := \alpha - a_0 \tag{3.1}$$

waarbij $[\alpha]$ het grootste gehele getal $\leqslant \alpha$ is. Dus

$$\alpha = a_0 + \rho_1 \text{ met } a_0 \text{ geheel en } 0 \leqslant \rho_1 < 1. \tag{3.2}$$

Als $\rho_1 \neq 0$ is, dan schrijven we $\alpha_1 := 1/\rho_1$ (met $\alpha_1 > 1$) en behandelen vervolgens α_1 soortgelijk als α:

$$a_1 := [\alpha_1] + \rho_2. \text{ Etcetera.} \tag{3.3}$$

Als $\rho_k = 0$ dan stoppen we. De getallen $a_0, a_1, a_2, \ldots$ noemen we de wijzergetallen van de (reguliere) kettingbreuk van α:

$$\alpha = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots}}} \tag{3.4}$$

De wijzergetallen kunnen berekend worden met behulp van een programma zoals

Programma 6 (KETBRTST)

```
10 'Programma 6 (KETBRTST)
20 CLS: PRINT"Wijzergetallen van SQR(N) ZONDER PRECISIE CONTROLE."
30      PRINT"..................................................."
40      PRINT"                                  .....          "
50      PRINT"GEVAARLIJK IN'T GEBRUIK !!!!!": BEEP: BEEP: BEEP: BEEP: BEEP
60      PRINT"...........................": PRINT
70 DEFDBL A-Z
80 INPUT"N =";N:X=SQR(N)
90 PRINT:PRINT"We bepalen de wijzergetallen van X =";X:PRINT
100 FOR K%=0 TO 30
110 A=INT(X):PRINT"(index =";K%;")",,"wijzergetal =";A:X=1/(X-A)
120 NEXT K%
130 BEEP:PRINT"De laatste wijzergetallen KUNNEN ONBETROUWBAAR zijn !!!!!"
140 GOTO 80
```

De kettingbreuk (3.4) van α geven we aan met $\{a_0; a_1, a_2, \ldots\}$.

Bij een rekenproces als dit, spelen precisie kwesties een uiterst belangrijke rol. Men kan met de hand nagaan dat $\sqrt{2} = \{1;2,2,2,2,\ldots\}$. Wat zijn de eerste 25 wijzergetallen van $\sqrt{2}$ die door Programma 6 worden afgeleverd?

Om de berekening van wijzergetallen zeker te stellen, maken we gebruik van de volgende kunstgreep: kies een geschikte $\epsilon > 0$ en bepaal simultaan de wijzergetallen van de kettingbreuken van

$$\beta := \alpha - \epsilon, \quad \alpha, \quad \gamma := \alpha + \epsilon. \tag{3.5}$$

Als nu voor $0 \leqslant k \leqslant N$ geldt dat de drie corresponderende wijzergetallen aan elkaar gelijk zijn, dan zijn de gevonden wijzergetallen van α correct voor $0 \leqslant k \leqslant N$. Zie het volgende programma.

Programma 7 (KETBREUK)

```
10 'Programma 7 (KETBREUK)
20 CLS:PRINT"Kettingbreuk van SQR(N) MET PRECISIE CONTROLE."
30     PRINT"..........................................."
40 DEFDBL A-Z
50 INPUT"N =";N:X=SQR(N)
60 PRINT:PRINT"We bepalen de wijzergetallen van SQR(N) met N =";N:PRINT
70 EPS=10#^(-16#+LOG(X)/LOG(10#))
80 XL=X-EPS:XR=X+EPS
90 FOR K%=0 TO 50
100 AL=INT(XL):A=INT(X):AR=INT(XR)
110 IF AL<>A OR A<>AR GOTO 150
120 PRINT"(index =";K%;")",,"wijzergetal =";A
130 XL=1/(XL-AL):X=1/(X-A):XR=1/(XR-AR)
140 NEXT K%
150 BEEP:PRINT:PRINT"(EINDE van PRECISIE bereikt)"
160 PRINT"AL =";AL;" A =";A;" AR =";AR"
170 GOTO 50
```

Opgave. Ga in de literatuur na hoe kettingbreuk-ontwikkeling toegepast kan worden voor snelle berekening van inversen in een gereduceerd restsysteem (mod m). Merk verder op dat er een sterk verband bestaat tussen het berekenen van de wijzergetallen van $\alpha = a/b$ (met gehele a en b) en het Euclidisch algoritme ter bepaling van de GGD van a en b (zie paragraaf 7).

Programma 8 (INVERSE)

```
10 'Programma 8 (INVERSE)
20 CLS: PRINT"Programma: MULTIPLICATIEVE INVERSE van A (mod M)"
30      PRINT"..................................................":PRINT
40 '! Voor details zie DAVENPORT, Chapter 4, pag. 88 - 89.
50 DEFDBL A-Z:DIM NOEM(32),CF(30)
60 INPUT"Voer de modulus M in (1 < M < Maxint)";M
70 'We gaan eerst na of M niet te klein en niet te groot is.
80 R=M*M:IF 1 < M AND R < R+1 GOTO 100
90 PRINT:PRINT"De modulus M deugt niet ! M =";M:BEEP:BEEP:GOTO 60
100 INPUT"Voer de te INVERTEREN A in";A:IF A < 1 GOTO 100
110 NOEM(1)=0:NOEM(2)=1:N1=M:N2=A
120 FOR I%=1 TO 30
130 NQ=INT(N1/N2):CF(I%)=NQ:N2H=N1-NQ*N2
140 IF N2H > 0 THEN N1=N2:N2=N2H:GOTO 170
150 'We gaan nu na of inderdaad (A,M) = 1
160 IF N2 > 1 THEN PRINT:PRINT"! FOUT: GGD(A,M) =";N2:BEEP:STOP ELSE GOTO 180
170 NEXT I%
180 LJ%=I%-1
190 FOR J%=1 TO LJ%
200 NOEM(J%+2)=NOEM(J%+1)*CF(J%)+NOEM(J%)
210 NEXT J%
220 IF LJ% MOD 2 = 0 THEN INV=NOEM(LJ%+2):GOTO 240
230 INV=-NOEM(LJ%+2)+M
240 PRINT:PRINT"De inverse van A (mod M) =";INV
250 END
```

Uitdaging. Bepaal $[1/(2^{1/n}-1)]$ voor een groot aantal $n \in \mathbb{N}$. Tracht na te gaan of de uitkomsten juist zijn.

Opgave. Schrijf een programma om na te gaan of $k \in \mathbb{N}$ een zuivere macht is (d.w.z. van de vorm m^n met natuurlijke m en n, beide > 1). Bouw een verificatie in DP-gehele-getallen-arithmetiek in (DP staat voor Double Precision).

4. Pseudo random getallen

Opgave. Experimenteer eens met het volgende programma dat (pseudo) random getallen (uniform verdeeld tussen 0 en 1) tracht te genereren.

Programma 9 (RANDOMGR)

```
10 'Programma 9 (RANDOMGR)
20 CLS:PRINT "Eigen (pseudo) RANDOM GENERATOR"
30 PRINT"Houd zelf de KWALITEIT van deze random generator in het oog !"
40 DEFDBL D:DF=10#^4:DF2=DF*DF:BEEP:PRINT
50 PRINT"Voer een POSITIEF GETAL (tussen 0 en 1) in.":PRINT
60 INPUT DN
70 IF DN*DN<=10#^15 THEN DN=2*DN:GOTO 70 ELSE NR=0
80 FOR I=1 TO 20
90 'merk op dat DN^2 uit 16 cijfers bestaat.
100 'schrap de laatste 4 cijfers van DN^2.
110 DNH=INT(DN^2/DF)
120 'schuif nu de komma 8 plaatsen naar links.
130 DG=DNH/DF2
140 'neem het fractionele deel van DG met 8 cijfers achter de komma.
150 DR=DG-INT(DG):DR=INT(DR*DF2)/DF2:NR=NR+1
160 PRINT"(nr ";NR;")","random getal =";DR
170 'schuif de komma in DR 8 plaatsen naar rechts (DN is weer geheel).
180 DN=DR*DF2
190 NEXT I
200 BEEP:PRINT "Druk op CONT ( KEY F5 ) om door te gaan ( Olivetti M 24 )"
210 STOP:GOTO 80
```

Opgave. Hoe kan men zelf een pseudo random generator maken die (goede benaderingen van) onafhankelijke trekkingen uit een normale verdeling aflevert? (Denk aan het gemiddelde van een niet al te klein aantal random getallen, afgeleverd door het vorige programma, of door de machinefunctie RND.) Probeer hiermee (grafisch) Brownse bewegingen te simuleren. Een suggestie is

Programma 10 (KOMEET)

```
10 'Programma 10 (KOMEET)
20 CLS:B=640:E=1:T=2:H=.5:RANDOMIZE
30 SCREEN 3:WINDOW(1,-200)-(B,200):LINE(1,0)-(B,0)
40 S=0:FOR I=1 TO B:S=S+INT(T*RND+H)-E:PSET(I,S):NEXT I:GOTO 40
```

5. Het grootste machinegetal

In theorie zijn er oneindig veel gehele getallen. In de praktijk van het machinaal rekenen is deze gedachte evenwel niet te verwezenlijken. Met behulp van het volgende programma berekenen we het grootste (gehele) getal dat nog exact door de machine gerepresenteerd kan worden (zonder speciale 'multiprecisie' trucs). Merk op dat het grootste machinegetal gedefinieerd is als het

kleinste natuurlijke getal n met (schrik niet!) de 'machine-eigenschap' $n+1 \leqslant n$.

Programma 11 (MAXNDBL)

```
10 'Programma 11 (MAXNDBL)
20 CLS:PRINT"We bepalen de GROOTSTE DP-INTEGER op de Olivetti M 24.":PRINT
30 DEFDBL A-Z
40 MAX=1#:INC=1#:TEST$="n"        ' "n" voor Normaal, "a" voor Abnormaal
50 N=MAX+INC:IF N+1# <= N THEN TEST$="a" ELSE MAX=N
60 'PRINT"MAX =";MAX,"INC =";INC
70 IF TEST$="n" THEN INC=2#*INC:GOTO 50
80 INC=INC/2#:IF INC>=1# GOTO 50 ELSE PRINT"*****"
90 PRINT"MAXIMALE DP-INTEGER is 7 * (10#^16#) +"; MAX - 7 * 10#^16#
100 END
```

Opgave. Schrijf soortgelijke programma's ter bepaling van de grootste IP en SP (Integer- en Single Precision) gehele getallen. Verifieer de uitkomsten met de opgegeven waarden in de manual bij uw PC. Ga na dat al deze maxima van de vorm $2^n - 1$ zijn. Bepaal de corresponderende waarden van n. Wat valt hierbij op, wetende dat uw PC een 16 bits machine is (zoals de Olivetti M24)? Ga na of deze maxima priem zijn.

Opgave. Wat zijn de minimale IP, SP en DP gehele getallen op uw PC?

6. De afstand van x tot $\mathbb{Z}$

In verband met precisie-kwesties kan het van belang zijn te weten hoe ver een reële x van $\mathbb{Z}$ af ligt. De afstand $d(x,\mathbb{Z})$ van x tot $\mathbb{Z}$ definiëren we als

$$d(x,\mathbb{Z}) := \min_{a\in\mathbb{Z}} |x-a|. \tag{6.1}$$

Merk op dat $d(x,\mathbb{Z}) = \min\{\alpha, 1-\alpha\}$ met $\alpha = x-[x] =: \{x\}$, het zogenaamde fractionele deel van x ($\in \mathbb{R}$).

Verifieer het volgende programma ter bepaling van $d(x,\mathbb{Z})$.

Programma 12A (AFSTANDZ)

```
10 'Programma 12A (AFSTANDZ)
20 CLS: PRINT "We bepalen de AFSTAND van X tot ZZ.": PRINT
30 REM Let op de PRECICIE !
40 'DEFDBL A - Z
50 INPUT "Voer X in"; X
60 A = X - INT(X): B = 1 - A
70 IF A > B THEN D = B ELSE D = A        'Alternatief D = ABS(X - INT(X + .5))
80 PRINT: PRINT "De AFSTAND van X tot ZZ bedraagt"; D: PRINT
90 GOTO 50
```

Experiment. Kies een irrationaal reëel getal α en bereken een aantal natuurlijke getallen N met de eigenschap

$$d(N\alpha,\mathbb{Z}) < d(n\alpha,\mathbb{Z}) \text{ voor alle } n < N. \tag{6.2}$$

($N\alpha$ moet dus dichter bij $\mathbb{Z}$ liggen dan alle 'vorige' $n\alpha$.) Tracht een verband op te sporen tussen deze getallen N en de noemers van de kettingbreukbenaderingen van α.

Opgave. Verifieer het volgende programma dat voor reële x het dichtstbij gelegen gehele getal bepaalt.

Programma 12B (NEARINT)

```
10 'Programma 12B (NEARINT)
20 CLS
30 PRINT "We bepalen het GEHELE getal IX dat het dichtst bij X ligt": PRINT
40 REM ALS X = I + .5 dan "is" IX := I + 1. Let op de PRECISIE.
50 'DEFDBL A - Z
60 INPUT "Voer X in"; X: PRINT
70 PRINT "N(X) ="; INT(X + .5): PRINT
80 GOTO 60
```

7. GGD

DEFINITIE. Voor gehele getallen a en b (niet beide gelijk aan 0) is de Grootste Gemene Deler van a en b (net wat het woord suggereert) het grootste getal dat zowel een deler van a als van b is. De GGD van a en b noteren we als (a,b).

Voorbeelden. $(0,1) = 1$, $(5,7) = 1$, $(8,12) = 4$.

Om (a,b) uit te rekenen, kunnen we in principe alle gemeenschappelijke delers van a en b opsporen en dan van al deze delers de grootste nemen. Dit blijkt zelfs voor betrekkelijk kleine a en b al uitermate tijdrovend te zijn. Men stelle zich dit eens voor als a en b beide in de buurt van 10^{16} liggen! Niet in de laatste plaats hierom is het zeer verrassend dat er voor de (snelle!) berekening van (a,b) het al zeer oude Euclidische algoritme bestaat. Dit algoritme is voornamelijk gebaseerd op de volgende eigenschappen:

(1) $(a,b) = (b,a) = (|a|, |b|)$ als a en b niet beide gelijk aan 0 zijn,
(2) als $b>0$ dan is $(b, 0) = b$,
(3) als tussen gehele getallen a,b,q en r met $b\neq 0$ de relatie $a = qb+r$ bestaat dan is $(a,b) = (b,r)$.

Een GGD programma kan er als volgt uitzien.

Programma 13 (GGD)

```
10 'Programma 13 (GGD)
20 'DEFDBL A - Z
30 CLS: PRINT "We bepalen de GGD van M en N.": PRINT
40 INPUT "Voer M in"; M: INPUT "Voer N in"; N
50 IF M=INT(M) AND N=INT(N) GOTO 70
60 BEEP: PRINT "Input fout ! M en/of N is niet geheel.": GOTO 40
70 MA=ABS(M): NA=ABS(N)
80 IF MA > NA THEN Q=MA: R=NA ELSE Q=NA: R=MA
90 IF Q > 0 GOTO 110
100 BEEP: PRINT "Input fout ! BEIDE argumenten M en N zijn gelijk aan 0.": STOP
110 IF R > 0 GOTO 130
120 PRINT "GGD("; M; ","; N; ") ="; Q: GOTO 40
130 HR=Q - R * INT(Q/R)
140 IF HR > 0 THEN Q=R: R=HR: GOTO 130
150 PRINT "GGD("; M; ","; N; ") ="; R: GOTO 40
```

Opgave. Schrijf een programma ter bepaling van alle natuurlijke a die onderling ondeelbaar zijn met m (bij gegeven $m \in \mathbb{N}$).

8. EULER'S ϕ FUNCTIE

In menig onderdeel van de wiskunde speelt de arithmetische functie $\phi(n)$ een belangrijke rol. We definiëren $\phi(n)$ voor $n \in \mathbb{N}$ als het aantal getallen a met de eigenschappen

$$1 \leqslant a \leqslant n \quad \text{en} \quad (a,n) = 1. \tag{8.1}$$

Voorbeelden. $\phi(1) = 1$; $\phi(p) = p - 1$ voor elk priemgetal p.

Als n de standaard priemontbinding

$$n = \Pi p^e \tag{8.2}$$

heeft, dan is

$$\phi(n) = \Pi p^{e-1}(p-1) = n\Pi(1-\frac{1}{p}). \tag{8.3}$$

Hierop is het volgende programma ter berekening van $\phi(n)$ gebaseerd.

Programma 14 (SNELPHI)

```
10 'Programma 14 (SNELPHI)
20 CLS: PRINT "Programma ter bepaling van phi(n).": PRINT
30 'DEFDBL A - Z
40 INPUT "Voer N in"; N
50 IF N >= 1 GOTO 70
60 BEEP: PRINT "Fout in SNELPHI: argument N < 1.": GOTO 40
70 IF N=INT(N) GOTO 90
80 BEEP: PRINT "Fout in SNELPHI: argument is niet geheel.": GOTO 40
90 PHI=1
100 IF N=1 OR N=2 GOTO 230 ELSE NH=N
110 P=2: Q=NH/P: IF Q <> INT(Q) GOTO 140
120 NH=Q: GOSUB 260          ' remove P from NH and compute PHI
130 IF NH=1 GOTO 230
140 P=3: Q=NH/P: IF Q <> INT(Q) GOTO 170
150 NH=Q: GOSUB 260          ' remove P from NH and compute PHI
160 IF NH=1 GOTO 230
170 P=1: INC=2: MAX=INT(SQR(NH+.5))
180 INC=6-INC: P=P+INC: IF P > MAX THEN PHI=PHI*(NH-1): GOTO 230
190 Q=NH/P: IF Q <> INT(Q) GOTO 180
200 NH=Q: GOSUB 260          ' remove P from NH and compute PHI
210 IF NH=1 GOTO 230
220 MAX=INT(SQR(NH+.5)): GOTO 180
230 PRINT "PHI("; N; ")="; PHI: PRINT
240 GOTO 40                  'Alternatief END
250 'Subroutine REMOVE P FROM NH and COMPUTE PHI.
260 Q=NH/P          'N.B. Voor de aanroep pleegden we al NH <-- Q <-- NH/P
270 IF Q <> INT(Q) THEN PHI=PHI*(P-1): RETURN ELSE PHI=PHI*P: NH=Q: GOTO 260
```

Opgave. Vergelijk de snelheid van Programma 14 met een (zelf te maken) programma dat $\phi(n)$ bepaalt door alle a tussen 1 en n te testen op de eigenschap $(a,n) = 1$.

Opgave. Vind enkele ('Phibonacci') getallen n met de eigenschap

$$\phi(n+2) = \phi(n+1) + \phi(n). \tag{8.4}$$

9. De functie van Möbius

Een van de belangrijkste bronnen voor uiterst moeilijke problemen in de (analytische) getaltheorie is wel de functie van Möbius die we als volgt definiëren:

$$\mu(n) := \begin{cases} 0 \text{ als } n \text{ deelbaar is door } p^2 \text{ voor zekere priem } p \\ (-1)^k \text{ als } n = p_1 p_2 ... p_k \text{ met alle } p_j \text{ priem.} \end{cases} \tag{9.1}$$

Voorbeelden. $\mu(1) = 1$; $\mu(2) = -1$; $\mu(6) = 1$; $\mu(30) = -1$; $\mu(98) = 0$.

We geven het volgende programma ter bepaling van $\mu(n)$.

Programma 15 (MUN)

```
10 'Programma 15 (MUN)
20 CLS: PRINT "Programma ter bepaling van MU(N).": PRINT
30 INPUT "Voer N in"; N: IF N >= 1 GOTO 50
40 PRINT "Fout in MUN: argument < 1": GOTO 30
50 IF N=INT(N) GOTO 70
60 PRINT "Fout in MUN: argument is niet geheel": GOTO 30
70 MUN=1: IF N=1 GOTO 190
80 NH=N
90 P=2: GOSUB 220                    ' remove P from NH and compute MUN
100 IF NH=1 OR MUN=0 GOTO 190
110 P=3: GOSUB 220                   ' remove P from NH and compute MUN
120 IF NH=1 OR MUN=0 GOTO 190
130 P=1: INC=2: MAX=INT(SQR(N+.5))
140 INC=6-INC: P=P+INC: IF P > MAX GOTO 180
150 GOSUB 220
160 IF NH=1 OR MUN=0 GOTO 190
170 MAX=INT(SQR(NH+.5)): GOTO 140
180 MUN=-MUN
190 PRINT "MU("; N; ")="; MUN
200 GOTO 30                          'Alternatief END
210 'SUBROUTINE. We verwijderen P uit NH en stellen MUN bij.
220 KOUNT=0
230 Q=NH/P: IF Q <> INT(Q) GOTO 250
240 NH=Q: KOUNT=KOUNT+1: GOTO 230
250 IF KOUNT > 1 THEN MUN=0: RETURN
260 IF KOUNT=1 THEN MUN=-MUN
270 'Bij KOUNT=0 laten we MUN ongemoeid !
280 RETURN
```

Om een indruk te krijgen van het oscillatorisch gedrag van $\mu(n)$ geven we het volgende programma dat een grafiek op het scherm brengt van de functie

$$M(x) := \sum_{n \leqslant x} \mu(n). \tag{9.2}$$

Programma 16 (PLOTSMU)

```
10 'Programma 16 (PLOTSMU)
20 CLS:PRINT"We plotten SIGMA over n van MU(n).": PRINT
30 PRINT"Hoeveel sommen wilt U geplot hebben op een scherm ?"
40 PRINT"Neem een veelvoud van 640"
50 INPUT B: H=.5*SQR(B)
60 'INPUT"Wat is de verwachting van de grootste waarde van |SIGMA MU(n)| ?";H
70 CLS:SCREEN 3:WINDOW(1,-H)-(B,H):LINE(1,0)-(B,0)
80 N=1:S=1:MA=1:MI=1:PSET(1,1)
90 FOR J=1 TO B
100 N=N+1
110 GOSUB 220:S=S+MU                          ':IF S=0 THEN PRINT"n =";N
120 PSET(N,S)
130 NEXT J
140 BEEP:PRINT"Press KEY F5":STOP:H=.5*SQR(N+B)
150 CLS:WINDOW(N,H)-(N+B,-H):PRINT"We gaan door vanaf N =";N;" ( s =";S;")"
160 LINE(N,0)-(N+B,0):GOTO 90
170 'IF S=0 THEN PRINT"!  s = 0 voor k =";N:GOTO 60
180 'IF S>MA THEN MA=S:PRINT"+  max voor k =";N;"ma =";MA;" mi =";MI:GOTO 60
190 'IF S<MI THEN MI=S:PRINT"-  min voor k =";N;"mi =";MI;" ma =";MA:GOTO 60
200 'GOTO 60
210 '* * * * *   subroutine MU(N)   * * * * *
220 MU=1:NH=N
230 P=2:NP=0
240 Q=NH/P:IF Q<>INT(Q) THEN GOTO 250 ELSE NH=Q:NP=NP+1:GOTO 240
250 IF NP>1 THEN MU=0:RETURN
260 IF NP=1 THEN MU=-MU:IF NH=1 THEN RETURN
270 P=3:NP=0
280 Q=NH/P:IF Q<>INT(Q) THEN GOTO 290 ELSE NH=Q:NP=NP+1:GOTO 280
290 IF NP>1 THEN MU=0:RETURN
300 IF NP=1 THEN MU=-MU:IF NH=1 THEN RETURN
310 P=1:I=2:MAX=INT(SQR(NH+.5))
320 I=6-I:P=P+I:IF P>MAX THEN MU=-MU:RETURN
330 Q=NH/P:IF Q<>INT(Q) THEN GOTO 320 ELSE NH=Q
340 Q=NH/P:IF Q=INT(Q) THEN MU=0:RETURN ELSE MU=-MU
350 IF NH>1 THEN GOTO 320 ELSE RETURN
```

10. Een observatie van Erdös

Toen de Hongaarse wiskundige P. Erdös 64 jaar werd, moet hij opgemerkt hebben dat $64 = 2^6$. Het komt dus voor dat het getal 2^n 'net zo begint' als het getal n (alles decimaal geschreven). Nog zo'n voorbeeld is $1024 = 2^{10}$.

Naar aanleiding hiervan moet Erdös de vraag gesteld hebben of dit verschijnsel zich nog vaker voordoet. Directe berekeningen leiden niet zo maar eventjes tot een nieuw voorbeeld van een 'Erdös getal'. We analyseren de situatie als volgt.

Schrijven we het natuurlijke getal n in het tientallige stelsel, dan is het Aantal Cijfers $\mathrm{ac}(n)$ van n gelijk aan

$$\mathrm{ac}(n) = [{}^{10}\log n] + 1. \tag{10.1}$$

Verder merken we op dat het natuurlijke getal a dat door de eerste k cijfers van het natuurlijke getal m wordt gevormd, gelijk is aan

$$a = \left[\frac{m}{10^{\mathrm{ac}(m)-k}}\right] \tag{10.2}$$

hetgeen gemakkelijk in te zien is door kommaverschuiving (schuif in m de komma eerst $\mathrm{ac}(m)$ plaatsen naar links, daarna k plaatsen naar rechts en neem dan de entier).

Opgave. Schrijf een programma dat het eerste cijfer c_1 van $n \in \mathbb{N}$ bepaalt in een g-tallig stelsel (schrijf daarbij de output c_1 gewoon tientallig).

Opgave. Converteer een getal naar een g-tallig stelsel.

Door samenvoeging van deze (theoretische) overwegingen verkrijgen we een waterdicht criterium voor een Erdös getal n (bij grondtal $g = 10$ en basis $b = 2$)

$$n = [\frac{2^n}{10^{\mathrm{ac}(2^n)-\mathrm{ac}(n)}}]. \tag{10.3}$$

Dit criterium is gelijkwaardig met de ongelijkheid

$$n \leqslant \frac{2^n}{10^{\mathrm{ac}(2^n)-\mathrm{ac}(n)}} < n+1 \tag{10.4}$$

of, na het nemen van logaritmen (we schrijven LOG voor $^{10}\log$),

$$\mathrm{LOG}(n) \leqslant n\,\mathrm{LOG}(2) - \mathrm{ac}(2^n) + \mathrm{ac}(n) < \mathrm{LOG}(n+1) \tag{10.5}$$

of, met gebruikmaking van de formule voor ac(n) in (10.1),

$$\mathrm{LOG}(n) \leqslant n\,\mathrm{LOG}(2) - [n\,\mathrm{LOG}(2)] - 1 + [\mathrm{LOG}(n)] + 1 < \mathrm{LOG}(n+1) \tag{10.6}$$

of, tenslotte,

$$0 \leqslant \{n\,\mathrm{LOG}(2)\} - \{\mathrm{LOG}(n)\} < \mathrm{LOG}(n+1) - \mathrm{LOG}(n). \tag{10.7}$$

Hiermee hebben we een handig rekentechnisch instrument verkregen om na te gaan of $n \in \mathbb{N}$ een Erdös getal is of niet.

We geven het volgende programma ter berekening van Erdös getallen (bij grondtal $g = 10$ en basis $b = 2$). Met een soortgelijk (FORTRAN) programma zijn op een CYBER 175/750 de volgende Erdös getallen gevonden

6; 10; 1542; 77075; 113939; 1122772.

Tracht dit te verifiëren met

Programma 17 (ERDOES)

```
10 'Programma 17 (ERDOES)
20 CLS: PRINT"We bepalen ERDOES-GRTALLEN bij de basis 2.": PRINT
30 DEFDBL A - Z
40 DL10=LOG(10#): DL2=LOG(2#)/DL10
50 INPUT"We beginnen met N ="; DN: PRINT
60 DLDN=LOG(DN)/DL10
70 DLDNP1=LOG(DN+1#)/DL10: DNDL2=DN*DL2
80 D1=(DNDL2-INT(DNDL2))-(DLDN-INT(DLDN))
90 D2=DLDNP1-DLDN
100 IF D1 < 0 OR D2 <= D1 GOTO 120
110 PRINT"N ="; DN; " is een Erdoes-getal"
120 DN=DN+1#: DLDN=DLDNP1: GOTO 70
```

Opgave. Verifieer het volgende meer algemene programma ter bepaling van Erdös getallen bij grondtal g en basis b. Ga na dat bij $g = 51$ en $b = 2$ het getal $n = 615013$ voldoet. Wat is het kleinste Erdös getal bij $g = 22$ en $b = 2$? (Het is de spreker niet bekend of er bij elke keuze van g en b een Erdös getal bestaat, ook al ziet het er naar uit van wel.)

Programma 18 (ERDOES2)

```
10 'Programma 18 (ERDOES2)
20 CLS: PRINT "We zoeken EEN Erdoes-getal bij grondtal G en basis B.": PRINT
30 DEFDBL A-Z: EPS=.00001#
40 INPUT "Met welk GRONDTAL G beginnen we";G: INPUT "Welke basis B kiezen we";B
50 LG=LOG(G): LB=LOG(B)/LG: N=2: LN=LOG(N)/LG
60 LNP1=LOG(N+1)/LG:NLB=N*LB
70 D1=NLB-INT(NLB)-(LN-INT(LN)): D2=LNP1-LN
80 IF D1 < 0 OR D2 <= D1 THEN LN=LNP1: N=N+1: GOTO 60
90 PRINT"G =";G,"B =";B,"Erdoes getal"," N =";N
100 D=D2-D1: IF D > EPS THEN G=G+1: GOTO 50
110 PRINT: PRINT"( testwaarde =";CSNG(D);")":PRINT
120 G=G+1: GOTO 50
```

Opgave. Verifieer een aantal (kleine) Erdös getallen 'met de hand'!

11. Het KGV van de eerste n natuurlijke getallen

Een vrij groot deel van de analytische getaltheorie draait om de kwestie hoe groot het KGV van de eerste n natuurlijke getallen is. Om de discussie wat te vergemakkelijken, voeren we een aantal begrippen in. Men doet er goed aan zich hierbij steeds voor ogen te houden hoe men zelf (de natuurlijke logaritme van) het KGV van de getallen $1,2,3,4,..,n$ zou bepalen.

Definitie. Onder de von Mangoldt functie $\Lambda(n)$ verstaan we

$$\Lambda(n) := \begin{cases} \log(p) \text{ als } n = p^k, \ p \text{ priem en } k \in \mathbb{N}. \\ 0 \text{ anders.} \end{cases} \tag{11.1}$$

Definitie. Onder de Chebycheff functie $\psi(x)$ verstaan we

$$\psi(x) := \sum_{n \leqslant x} \Lambda(n). \tag{11.2}$$

In woorden: $\psi(x)$ is de logaritme van het *KGV* van alle natuurlijke getallen die $\leqslant x$ zijn.

Voorbeeld. $\psi(10) = 3\log(2)+2\log(3)+\log(5)+\log(7)$.

Littlewood heeft aangetoond dat de functie

$$\psi(x)-x \tag{11.3}$$

(zowel naar onderen als naar boven) niet begrensd is als $x \to \infty$. Op grond hiervan en het triviale feit dat $\psi(x)$ op elk begrensd interval slechts eindig veel verschillende waarden aanneemt ($\psi(x)$ is een eenvoudige trapfunctie met sprongen in de punten $x = p^k$), kan aangetoond worden dat de vergelijking

$$\psi(x) = x \tag{11.4}$$

oneindig veel reële oplossingen heeft. Aan de hand van een plaatje ziet men gemakkelijk in dat $u \in \mathbb{R}$ een oplossing van (11.4) is, als er een geheel getal n bestaat met de eigenschappen

$$u = \psi(n) \quad \text{en} \quad 0 \leqslant \psi(n) - n < 1. \tag{11.5}$$

Alle oplossingen van (11.4) worden dus gevonden door alle gehele $n \geqslant 0$ op te sporen waarvoor

$$0 \leqslant \psi(n) - n < 1 \tag{11.6}$$

en dan voor de gevonden n's te nemen $u = \psi(n)$.

We voeren dit uit met behulp van het volgende programma.

Programma 19 (PSIXISX)

```
10 'Programma 19 (PSIXISX)
20 'FIXPOINTS of PSI(X).
30 'PSI(X) := SIGMA over n van LAMBDA(n) voor n <= X.
40 CLS: PRINT"We bepalen de positieve DEKPUNTEN van PSI(X).": PRINT
50 O=0: E=1: T=2: D=3: Z=6: H=.5
60 F=O: N=E              Definitie. F := PSI, G := PSI - N, L := LAMBDA
70 FOR II=1 TO 200
80 FOR JJ=1 TO 30000
90 N=N+E: NH=N
100 'we bepalen nu LAMBDA(n) via een SUBROUTINE (GOSUB 330).
110 P=T: GOSUB 260: IF K=E GOTO 170
120 P=D: GOSUB 260: IF K=E GOTO 170
130 P=E: I=T: MAX=INT(SQR(N+H))
140 I=Z-I: P=P+I: IF P <= MAX GOTO 160
150 L=LOG(N): GOTO 170
160 GOSUB 260: IF K=O GOTO 140
170 F=F+L: G=F-N
180 IF G < O OR G > E GOTO 200 ELSE A=A+E          A van Aantal (dekpunten)
190 PRINT"( nr =";A;")","PSI(X) = X   voor   X  =";F
200 NEXT JJ
210 NEXT II
220 END
230 '*************************************************************************
240 'SUBROUTINE ter bepaling van LAMBDA(N)
250 'NH (=N) en P worden aangeleverd. Output L := LAMBDA(N).
260 Q=NH/P: IF Q <> INT(Q) GOTO 280
270 NH=Q: IF NH > E GOTO 260
280 IF NH = N THEN K=O: RETURN
290 K=E: IF NH=E THEN L=LOG(P): RETURN ELSE L=O: RETURN
```

Opgave. Maak een grafiek van $\psi(x) - x$ volgens een programma zoals

Programma 20 (PSIXMINX)

```
10 'Programma 20 (PSIXMINX)
20 'PSI(X) := SIGMA van LAMBDA(n) over n <= X.
30 CLS:O=0:H=.5:E=1:T=2:D=3:Z=6
40 X=1:DX=1/4:BR=640          'BR van BReedte
50 RNG=0:CLS:SCREEN 3:Y=SQR(X+BR/2):WINDOW (X,-Y)-(X+BR,Y):LINE(X,0)-(X+BR,0)
60 PRINT"GRAFIEK van PSI(X) - X  voor";X;"<= X <=";X+BR
70 X=X+DX:RNG=RNG+DX:IF RNG>BR THEN X=X-DX:BEEP:STOP:GOTO 50
80 IF X<>INT(X) GOTO 110 ELSE N=X:GOSUB 150
90 'We kunnen tevens de dekpunten van PSI(X) (door BEEP) signaleren.
100 PSI=PSI+L:F=PSI-X           ':IF O<=F AND F<E THEN BEEP:GOTO 120
110 F=PSI-X
120 PSET(X,F):GOTO 70
130 '*****  SUBROUTINE PSI(X)
140 'Input: N; Output: PSI
150 NH=N
160 P=T:GOSUB 220:IF K=E THEN RETURN
170 P=D:GOSUB 220: IF K=E THEN RETURN
180 P=E: I=T:MAX=INT(SQR(N+H))
190 I=Z-I:P=P+I:IF P>MAX THEN L=LOG(N):RETURN
200 GOSUB 220:IF K=O GOTO 190 ELSE RETURN
210 '*****  SUBROUTINE ter bepaling van LAMBDA(NH)
220 Q=NH/P:IF Q <> INT(Q) GOTO 240
230 NH=Q:IF NH>E GOTO 220
240 IF NH=N THEN K=O:RETURN
250 K=E:IF NH=E THEN L=LOG(P) ELSE L=O
260 RETURN
```

12. Factorisatie volgens Fermat

Als $n = pq$ met p en q priem (en beide groot), dan is het doorgaans tijdrovend om op de 'gewone' manier de factoren p en q te vinden (bijv. met PRIEMTST). Fermat gaf een procedure aan die hiervoor heel geschikt is als p en q weinig van elkaar verschillen. Het is duidelijk dat we ons kunnen beperken tot oneven n. Als n samengesteld is, $n = ab$, met oneven a en $b (a \geqslant b)$ dan zijn

$$x = \frac{a+b}{2} \text{ en } y = \frac{a-b}{2} \tag{12.1}$$

beide geheel en we hebben, zoals men gemakkelijk narekent,

$$x^2 - y^2 = ab = n. \tag{12.2}$$

Omgekeerd, als we gehele x en y hebben zodanig dat $n = x^2 - y^2$, dan hebben we ook tegelijk een ontbinding van n. Immers, dan is $n = (x-y)(x+y)$.
Om zulke x en y te vinden, gaat men na of voor $0 \leqslant x \leqslant \sqrt{n}$ het getal $x^2 - n$ een zuiver kwadraat is. Het volgende programma vertelt enkele details.

Programma 21 (FERMAT)

```
10 'Programma 21 (FERMAT)
20 'Zie voor DETAILS het boek van ORE, p. 54 - 58.
30 'Zie ook het boek van SCHUH, p. 460 e.v.
40 CLS: PRINT"FACTORISATIE volgens FERMAT": PRINT
50 DEFDBL A-H: DEFINT I-N: DEFDBL O-Z
60 PRINT: INPUT "Voer het te factoriseren getal DN in"; DN
70 TIME$="000000": IF DN <= 2 OR DN <> INT(DN) GOTO 60
80 Q=DN/2:IF Q=INT(Q) GOTO 60
90 WN=INT(SQR(DN+.5))
100 'we testen of DN een kwadraat is
110 IF DN=WN*WN THEN PRINT"N =";DN;" is het kwadraat van";WN: GOTO 60
120 K=1: DM=WN+1: S=2*DM+1: X=DM*DM-DN
130 WX=INT(SQR(X+.5))
140 IF X <> WX*WX THEN X=X+S: K=K+1: S=S+2: GOTO 130
150 PRINT "N =";DN,"=";DM+K-WX-1;"*";DM+K+WX-1,,"TIJD =";TIME$
160 GOTO 60       'Alternatief END
```

Opgave. Verifieer enkele ontbindingen van getallen van het type pq met p en q grote priemen die niet al te veel van elkaar verschillen. Onderwerp dezelfde pq ook aan PRIEMTST en vergelijk de run-tijden.

13. RESTSYSTEMEN

DEFINITIE. Onder een volledig restsysteem (mod m) verstaan we m getallen die onderling niet congruent zijn (mod m).

Voorbeelden. $\{0,1,2,3,4\}$ (mod 5); $\{6,-5,8,15,-8,125\}$ (mod 6).

DEFINITIE. Onder een gereduceerd restsysteem (mod m) verstaan we $\phi(m)$ getallen die niet congruent zijn (mod m).

Voorbeelden. $\{1,2,3,4,5,6\}$ (mod 7); $\{1,3,5,7\}$ (mod 8).

We herinneren er aan dat als R een gereduceerd restsysteem (mod m) is en als $(a,m) = 1$ dan is ook $aR := \{ar \mid r \in R\}$ een gereduceerd restsysteem (mod m).

Opgave. Verifieer dit met de hand voor enkele niet al te grote $m \in \mathbb{N}$ en merk op dat de 'volgorde' der elementen een grillige verandering ondergaat.

De bedoeling van het volgende experiment is te laten zien dat deze grilligheid goeddeels schijn is. Er zit wel degelijk systeem in de volgorde-verandering van R bij vermenigvuldiging met a (zoals mij getoond werd door P. Schwerzel te Amsterdam). Men plaatse de punten $r \in R$ op een cirkel zo dat ze een regelmatige r hoek vormen

$$P_k = \exp(\frac{r}{m}\pi i), \quad r \in R. \tag{13.1}$$

Verbind nu het punt P_r rechtlijnig met het punt P_{ar}. Het volgende programma laat zien wat er gebeurt. (Neem als voorbeeld de input $m = 103$ en $a = 2$ of algemener $m = p$ met p priem en a een primitieve wortel van p.)

Programma 22 (SCHWERZEL)

```
10 'Programma 22 (SCHWERZEL)
20 CLS: PRINT "* * * * *     OMHULLENDE FIGUREN      * * * * *"
30       PRINT "Betreft  A * { 0, 1, 2, 3, ..., P-1 }  (mod P)": PRINT
40 BEEP: INPUT "Voer een geheel getal P in (tussen 100 en 500)"; P: PRINT
50 BEEP: PRINT "Voer een niet al te groot positief geheel getal A in"
60 INPUT "Sla de kleintjes vooral niet over !"; A
70 TPI=8*ATN(1#)
80 DH=TPI/P
90 CLS: PRINT "SCHWERZEL"
95 SCREEN 3: WINDOW (-1.2*4/3,-1.2)-(1.2*4/3,1.2)
100 PRINT "P ="; P, "A ="; A
110 FOR I=1 TO P
120 H1=I*DH: F=A*I MOD P: H2=F*DH
130 LINE (COS(H1),SIN(H1))-(COS(H2),SIN(H2))
140 NEXT I
150 PRINT: PRINT "Press F5": PRINT: STOP: GOTO 20        'Alternatief END
```

Hoe hangt de 'veelvoudigheid' van de ontstane (inhullende) figuur van a af?

14. Orde van a (mod m)

Definitie. De orde van a (mod m), met $(a,m) = 1$, is het kleinste natuurlijke getal ω met de eigenschap

$$a^{\omega} \equiv 1 \pmod{m}. \tag{14.1}$$

We merken op dat de orde van a (mod m) altijd een deler is van $\phi(m)$.

Het schijnt dat er geen efficiënt algoritme bestaat voor de berekening van de orde van a (mod m). Desondanks geven we het volgende

Programma 23 (ORDE)

```
10 'Programma 23 (ORDE)
20 CLS: PRINT "Programma ter bepaling van de ORDE van A (mod M)": PRINT
30 'DEFDBL D, M, N: DEFINT I - L
40 BEEP: INPUT "Voer A in"; A: INPUT "Voer de modulus M in"; M
50 IF M=1 GOTO 40
60 N=A: GOSUB 170      'Bepaal GGD(M,N)
70 IF GGD=1 GOTO 90
80 PRINT "GGD("; A; ","; M; ") ="; GGD; " > 1": GOTO 40
90 H=1
100 'De volgende loop heeft opzettelijk integer lengte !
110 FOR K=1 TO M-1: H=H*A: H=H-M*INT(H/M): IF H=1 THEN GOTO 130 ELSE NEXT K
120 NEXT K
130 PRINT "De orde van  "; A; "  modulo  "; M; " is  "; K
140 GOTO 40       'Alternatief END
150 ' SUBROUTINE GGD(M,N).
160 ' Input: M en N, beide > 0. Output: GGD
170 IF M > N THEN Q=M: R=N ELSE Q=N: DR=M
180 RH=Q-R*INT(Q/R)
190 IF RH = 0 THEN GGD=R: RETURN
200 Q=R: R=RH: GOTO 180
```

Dit programma is alleen geschikt voor niet al te grote waarden van m.

15. Primitieve wortels

Het getal a heet een primitieve wortel (of generator) van m als $(a,m) = 1$ en de orde van a (mod m) gelijk is aan $\phi(m)$. Anders gezegd: een generator van een gereduceerd restsysteem (mod m) is een getal a met de eigenschap dat de rij

$$a^1, a^2, a^3, \ldots, a^{\phi(m)}$$

weer een gereduceerd restsysteem (mod m) is. M.a.w., elk element van een gereduceerd restsysteem is te schrijven als een macht van a (alles modulo m).

Opgave. Ga na dat $m = 8$ geen primitieve wortel heeft. Bepaal primitieve wortels (mod 5) en (mod 7).

We geven het volgende simplistische programma voor de berekening van primitieve wortels.

Programma 24 (PRWORTEL)

```
10 'Programma 24 (PRWORTEL)
20 CLS:PRINT"Programma: PRIMITIEVE WORTEL van P"
30 PRINT"Dit programma is alleen geschikt voor kleine priemen !"
40 PRINT"..................................................":PRINT
50 DEFINT I-N
60 PRINT:INPUT"Van welk priemgetal P > 3 zoeken we primitieve wortels";IP:PRINT
70 IF IP > 3 GOTO 90
80 BEEP:PRINT "FOUT: input PRIEM < 5": GOTO 60
90 M=IP
100 FOR I=2 TO IP-1
110 N=I
120 GOSUB 240
130 IF GGD > 1 GOTO 200
140 RT=INT(SQR(I+.5)):IF RT*RT=I GOTO 200
150 RH=CSNG(I)
160 FOR K=2 TO IP-2
170 RH=RH*I:RH=RH-IP*INT(RH/IP):IF RH=1 GOTO 200
180 NEXT K
190 PRINT I;"  is een primitieve wortel van  ";IP
200 NEXT I
210 GOTO 60
220 '*******************************************
230 'SUBROUTINE GGD(M,N). Input: M,N.  Output: GGD
240 Q=M:R=N
250 RA=Q-R*INT(Q/R):IF RA = 0 THEN GGD=R:RETURN
260 Q=R: R=RA: GOTO 250
```

Ook dit programma is alleen geschikt voor niet al te grote priemen.

Opgave. Maak een tabel van de kleinste primitieve wortels van de eerste 100 priemgetallen. (Elk priemgetal heeft inderdaad een primitieve wortel.) Hoeveel primitieve wortels heeft een gereduceerd restsysteem (mod p) als p priem is? (Antwoord: $\phi(\phi(p)) = \phi(p-1)$.)

Opgave. Implementeer Theorem 4.8, p.83, uit LEVEQUE. Dit kan een aanzienlijke versnelling geven van de berekening van primitieve wortels (mod p). Hier is een voorstel:

Programma 25 (PRWSNEL)

```
10 'Programma 25 (PRWSNEL)
20 CLS:PRINT"Kleinste PRIMITIEVE WORTELS van alle ONEVEN priemen vanaf P": PRINT
30 DEFDBL A-Z:DIM PD(20)
40 BEEP:INPUT"P =";P:IF P=2*INT(P/2) THEN P=P-1
50 GOSUB 490            'Priemtest
60 IF T=1 THEN PM1=P-1:KNT=0 ELSE P=P+2:GOTO 50
70 'eerst alle priemdelers PD van PM1=P-1 zoeken en in PD(.) plaatsen
80 D=2:IF PM1=D*INT(PM1/D) THEN KNT=KNT+1:PD(KNT)=D:GOSUB 260
90 IF PM1=1 THEN JJL%=KNT:GOTO 160
100 D=3:IF PM1=D*INT(PM1/D) THEN KNT=KNT+1:PD(KNT)=D:GOSUB 260
110 IF PM1=1 THEN JJL%=KNT:GOTO 160
120 D=1:INC=2:MAX=SQR(PM1)
130 INC=6-INC:D=D+INC:IF D > MAX THEN KNT=KNT+1:PD(KNT)=PM1:JJL%=KNT:GOTO 160
140 Q=PM1/D:IF Q=INT(Q) THEN KNT=KNT+1:PD(KNT)=D:GOSUB 260
150 IF PM1>1 GOTO 130 ELSE JJL%=KNT
160 'PRINT"Aantal priemdelers van P-1 is";JJL%
170 FOR II%=2 TO 100                    'ii% is kandidaat voor een pr. w.
180 FOR JJ%=1 TO JJL%                   'bij vaste ii% alle jj% testen
190 IEXP=(P-1)/PD(JJ%):IMOD=P:B=II%
200 GOSUB 300: IF POWRES = 1 THEN GOTO 230
210 NEXT JJ%
220 PRINT"KLEINSTE PR. WORTEL van P =";P;" is ";II%:P=P+2:PM1=P-1:KNT=0:GOTO 50
230 NEXT II%
240 PRINT"GEEN pr. w. gevonden":P=P+2:PM1=P-1:GOTO 50
250 '***** Subroutine: CLEAR D out of PM1
260 Q=PM1/D:IF Q=INT(Q) THEN PM1=Q:GOTO 260 ELSE RETURN
270 STOP
280 '***** Subroutine POWRES: retourneert het residu van NB ^ IEXP (mod IMOD)
290 'INPUT"basis =";B:INPUT"exponent =";IEXP:INPUT"modulus =";IMOD
300 D=IMOD*IMOD:IF D >= D+1 THEN BEEP:PRINT" DE MODULUS IS TE GROOT":STOP
310 IF IEXP=0 THEN POWRES=1:RETURN
320 IF IEXP=1 THEN POWRES=B-IMOD*INT(B/IMOD):RETURN
330 IF IMOD=1 THEN POWRES=0:RETURN
340 'we breken de exponent binair af
350 DB=B:DB2=DB*DB:HEXP=IEXP:DM=IMOD
360 IF HEXP=2*INT(HEXP/2) GOTO 380
370 RES=DB2-DM*INT(DB2/DM):POW=B-IMOD*INT(B/IMOD):HEXP=INT(HEXP/2):GOTO 390
380 RES=DB2-DM*INT(DB2/DM):POW=1:HEXP=HEXP/2
390 IF HEXP=1 GOTO 450
400 IF HEXP=2*INT(HEXP/2) GOTO 430
410 DPROD=RES*POW
420 POW=DPROD-DM*INT(DPROD/DM)
430 DPROD=RES*RES
440 RES=DPROD-DM*INT(DPROD/DM):HEXP=INT(HEXP/2):GOTO 390
450 DPROD=RES*POW
460 POWRES=DPROD-DM*INT(DPROD/DM)
470 RETURN
480 '***** Subroutine PRIEMTEST
490 IF P=2 GOTO 560
500 D=2:IF D*INT(P/D)=P THEN T=0:RETURN
510 IF P=3 GOTO 560
520 D=3:IF D*INT(P/D)=P THEN T=0:RETURN
530 D=1:INC=2:MAX=SQR(P+.5)
540 INC=6-INC:D=D+INC:IF D > MAX THEN GOTO 560
550 Q=P/D:IF Q=INT(Q) THEN T=0:RETURN ELSE GOTO 540
560 'PRINT"P =";P;" is PRIEM":T=1:RETURN
570 T=1:RETURN
```

16. IS DE KLEINE STELLING VAN FERMAT OMKEERBAAR?
We herinneren er aan dat we $a \equiv b \pmod{m}$ schrijven, als m een natuurlijk getal is en $m \mid a-b$. Dit is gelijkwaardig met de uitspraak dat a en b bij deling door m dezelfde rest heben. We zeggen dan dat a en b 'congruent modulo m' zijn.

Voorbeelden. $a \equiv b \pmod{1}$ voor alle a en b; $2^{12} \equiv 13 \pmod{13}$.

We herinneren aan de kleine stelling van Fermat: Voor elk priemgetal p en elke a met $(a,p) = 1$ geldt

$$a^{p-1} \equiv 1 \ (mod\ p). \tag{16.1}$$

Voorbeeld. $2^{102} \equiv 1 \pmod{103}$.

Merk op dat deze stelling een bijzonder geval is van de stelling van Euler die zegt dat

$$a^{\phi(n)} \equiv 1 \pmod{n} \text{ als } (a,n) = 1. \tag{16.2}$$

Door bovendien te observeren dat

$$a^{\phi(n)} = a^{\phi(n)-1} \star a \tag{16.3}$$

vinden we een alternatieve methode om inversen in een gereduceerd rest-systeem (mod n) uit te rekenen.

Opgave. Toon aan dat voor elk oneven getal $n > 1$ geldt dat $(n-1)^{n-1} \equiv 1 \pmod{n}$. (N.B. $(n-1,n) = 1$.)

Conclusie: bij elke oneven n bestaat er wel een a met de eigenschappen

$$(a,n) = 1 \text{ en } a^{n-1} \equiv 1 \pmod{n}. \tag{16.4}$$

Uit de stelling van Fermat volgt dat $n \in \mathbb{N}$ wel samengesteld moet zijn als er een a bestaat zodanig dat

$$1 < a < n \text{ en } a^{n-1} \not\equiv 1 \pmod{n}. \tag{16.5}$$

Om numeriek na te kunnen gaan of er bij gegeven a en n voldaan is aan (16.4), geven we het volgende algemene programma ter bepaling van $a^k \pmod{n}$.

Programma 26 (POWRES)

```
10 'Programma 26 (POWRES)
20 CLS:PRINT"Programma (POWER RESIDUE) PR:= B^E (mod M)":PRINT
30 DEFDBL A-Z
40 BEEP:INPUT"Basis B =";B:INPUT"Exponent E =";IEXP:INPUT"Modulus M =";IMOD
50 D=IMOD*IMOD:IF D >= D+1 THEN BEEP:PRINT" De modulus M is TE GROOT":GOTO 40
60 GOSUB 120
70 PRINT"B =";B,"Exp =";IEXP,"Mod=";IMOD,"POWRES =";POWRES:PRINT
80 GOTO 40
90 END
100 'Subroutine POWER RESIDUE: retourneert NB ^ IEXP (mod IMOD)
110 'Geef de variabelen KORTERE NAMEN !
120 IF IEXP=0 THEN POWRES=1:RETURN
130 IF IEXP=1 THEN POWRES=B-IMOD*INT(B/IMOD):RETURN
140 IF IMOD=1 THEN POWRES=0:RETURN
150 'we breken de exponent binair af !
160 DB=B:DB2=DB*DB:HEXP=IEXP:DM=IMOD
170 IF HEXP=2*INT(HEXP/2) GOTO 190
180 RES=DB2-DM*INT(DB2/DM):POW=B-IMOD*INT(B/IMOD):HEXP=INT(HEXP/2):GOTO 200
190 RES=DB2-DM*INT(DB2/DM):POW=1:HEXP=HEXP/2
200 IF HEXP=1 GOTO 260
210 IF HEXP=2*INT(HEXP/2) GOTO 240
220 DPROD=RES*POW
230 POW=DPROD-DM*INT(DPROD/DM)
240 DPROD=RES*RES
250 RES=DPROD-DM*INT(DPROD/DM):HEXP=INT(HEXP/2):GOTO 200
260 DPROD=RES*POW
270 POWRES=DPROD-DM*INT(DPROD/DM)
280 RETURN
```

Opgave. Verifieer dat $2^{p-1} \equiv 1 \pmod{p^2}$ voor $p = 1093$. Schrijf een programma om nog zo'n priem p te vinden.

Opgave. Merk op dat in de berekening van $a^k \pmod{m}$ de exponent k binair 'ontrafeld' wordt. Waarom kan men de modulus m in programma POWRES niet te groot nemen (zie regel 50)?

Van historici vernemen we dat in de oudheid de Chinezen meenden het volgende priemcriterium te bezitten

$$n \text{ priem } \leftrightarrow n \mid 2^n - 2. \tag{16.6}$$

We gaan dit eens natrekken met behulp van het volgende programma (dat gebruik maakt van Programma 26 (POWRES)).

Programma 27 (CHINA)

```
10 'Programma 27 (CHINA)
20 CLS:PRINT"Programma: CHINA":PRINT
30 DEFDBL A - Z
40 INPUT"Voer de Basis B in";B
50 INPUT"Met welke (ONEVEN !) N > 3 beginnen we";N:PRINT
60 IF N<=3 OR 2*INT(N/2)=N THEN BEEP:GOTO 50 ELSE N=N-2
70 N=N+2
80 D=3:Q=N/D:IF Q=INT(Q) GOTO 120
90 D=1:I=2:MAX=SQR(N+.5)
100 I=6-I:D=D+I:IF D>MAX GOTO 70          'N moet samengesteld zijn !
110 Q=N/D:IF Q<>INT(Q) GOTO 100
120 E=N:M=N                               'E = Exponent; M = Modulus
130 GOSUB 180
140 IF POWRES <> B GOTO 70
150 PRINT"( B =";B;")  N =";N;"is een 'CHINEES' getal. ( D =";D;"N/D =";N/D;")"
160 GOTO 70
170 'Subroutine POWER RESIDUE: retourneert het residu van B ^ E (mod M)
180 B2=B*B
190 IF E=2*INT(E/2) GOTO 210
200 R=B2-M*INT(B2/M):POW=B-M*INT(B/M):E=INT(E/2):GOTO 220
210 R=B2-M*INT(B2/M):POW=1:E=E/2
220 IF E=1 GOTO 280
230 IF E=2*INT(E/2) GOTO 260
240 PROD=R*POW
250 POW=PROD-M*INT(PROD/M)
260 PROD=R*R
270 R=PROD-M*INT(PROD/M):E=INT(E/2):GOTO 220
280 PROD=R*POW
290 POWRES=PROD-M*INT(PROD/M)
300 RETURN
```

Opgave. Verifieer dat $n = 341 = 13*17$ het kleinste (samengestelde) natuurlijke getal is dat aan (16.6) voldoet. Uit hoeveel decimale cijfers bestaat 2^{341}?

Natuurlijke getallen n die aan (16.6) voldoen, worden pseudopriemen genoemd (bij de basis 2). Als men pseudopriemen gaat bepalen, dan krijgt men eerst de indruk dat ze allemaal oneven zijn. In 1951 werd door onze landgenoot Beeger aangetoond dat er oneindig veel even pseudopriemen bestaan. Het kleinste

$$161038 = 2.73.1103$$

werd in 1950 door Lehmer gevonden. Verifieer de pseudo-primaliteit van dit getal (met behulp van Programma 26 (POWRES)).

Opgave. Ga na dat $n = 91$ de kleinste pseudopriem is bij de basis 3. (Voor verdere literatuur verwijzen we naar ROTKIEWICZ.)

Het was Carmichael die in het begin van deze eeuw ontdekte dat er samengestelde n bestaan waarvoor $a^{n-1} \equiv 1 \pmod{n}$ voor alle a met $(a,n) = 1$. Deze getallen n worden Carmichael getallen genoemd. Het kleinste is 561. Ga dit na met behulp van (een enigszins aangepaste versie van) Programma POWRES. (Voor een snellere procedure en verdere theorie verwijzen we naar ORE.)

17. Variaties op de stelling van Wilson

De stelling van Wilson luidt: Het natuurlijke getal $n > 1$ is dan en slechts dan priem als $(n-1)!+1$ deelbaar is door n.

Opgave. Vind de kleinste priem p waarvoor $(p-1)!+1$ niet priem is.

Het zal blijken dat het voorkomt dat $a!+1$ deelbaar is door n voor zekere natuurlijke $a < n-1$. We zullen een aantal van zulke samengestelde n bepalen:
$n = 25$, $n = 121$, $n = 169$, $n = 437$, $n = 551$, $n = 667$, $n = 721$, $n = 1037$.

Programma 28 (WILSON1)

```
10 'Programma 28 (WILSON1)
20 CLS
30 PRINT "We zoeken SAMENGESTELDE natuurlijke getallen N met de eigenschap:"
40 PRINT: PRINT "                N | X! + 1 voor zekere natuurlijke X < N-1."
50         PRINT "                ........................................."
60 PRINT: INPUT "We beginnen met N ="; N: N=N-1: PRINT
70 N=N+1:GOSUB 150:IF T=1 GOTO 70 ELSE F=1
80 FOR I=2 TO N-2
90 F=F*I: F=F-N*INT(F/N)
100 IF F+1 <> N GOTO 120
110 PRINT "N ="; N; " X ="; I; "Een DELER van N is"; D
120 NEXT I
130 GOTO 70
140 '*****  Subroutine PRIEMTEST
150 IF N=2 THEN T=1: RETURN
160 D=2: Q=N/D: IF Q=INT(Q) THEN T=0: RETURN
170 IF N=3 THEN T=1: RETURN
180 D=3: Q=N/D: IF Q=INT(Q) THEN T=0: RETURN
190 D=1: I=2: MAX=SQR(N+.5)
200 I=6-I: D=D+I: IF D > MAX THEN T=1: RETURN
210 Q=N/D: IF Q=INT(Q) THEN T=0: RETURN ELSE GOTO 200
```

Opgave. Verifieer dat $p^2 \mid (p-1)!+1$ voor $p = 5$, $p = 13$ en $p = 563$.

Opgave. Vind een aantal priemen p met de eigenschap

$$a!+1 = 0 \pmod{p} \text{ voor zekere } 1 < a < p-1. \tag{17.1}$$

18. Een experiment met irrationale getallen

Zij $\alpha \in \mathbb{R}$ gegeven. We beschouwen de rij $[n\alpha]$ met $n \in \mathbb{N}$ en letten speciaal op de pariteit van $[n\alpha]$. Een natuurlijk middel om dit aspect te volgen is de som

$$S_N(\alpha) := \sum_{n=1}^{N} (-1)^{[n\alpha]}. \tag{18.1}$$

Merkwaardigerwijs komt het bij bepaalde (ook irrationale) $\alpha \in \mathbb{R}$ voor dat $S_N(\alpha)$ voor geen enkele waarde van N negatief is. Anders gezegd: $[n\alpha]$ is dan bij voorkeur even. Zonder bewijs vermelden we dat dit het geval is als alle wijzergetallen met even index (van de reguliere kettingbreuk) van α even zijn. Als voorbeeld neme men $\alpha = \sqrt{2}-1 = \{0;2,2,2,...\}$ of $\alpha = \sqrt{5}-2$.

Ga dit na met behulp van het volgende programma. (Experimenteer ook

eens met enkele eenvoudige breuken $\alpha = a/b$ met natuurlijke a en b.)

Programma 29 (ALTSUM)

```
10 'Programma 29 (ALTSUM)
20 CLS:PRINT"Programma ALTERNATING SUM"
30     PRINT".....................":PRINT
40 DEFDBL A-Z:O=0#:E=1#
50 A=SQR(2#):PRINT"We analyseren alpha =";A:PRINT
60 N=0:S=0:MI=0:MA=0
70 N=N+E:I=INT(N*A)
80 IF 2*INT(I/2)=I THEN S=S+E ELSE S=S-E
90 IF S<MI THEN MI=S:GOTO 110
100 IF S>MA THEN MA=S ELSE GOTO 70
110 PRINT"N =";N,"MIN SUM =";MI,"MAX SUM";MA
120 GOTO 70
```

Experimenteer ook eens met

Programma 30 (ALTSUM2)

```
10 'Programma 30 (ALTSUM2)
20 CLS:PRINT"ALTERNATING SUM 2":PRINT
35 DEFDBL A-Z
40 BREED=1280:SCREEN 3:WINDOW (0,-20)-(BREED,20):LINE(0,0)-(BREED,0)
60 A=SQR(2#)-1:PRINT"We analyseren alpha =";A:PRINT
70 N=0:S=0:E=1#
80 N=N+1:IF N>BREED THEN BEEP:STOP
90 I=INT(N*A):IF 2*INT(I/2)=I THEN S=S+E ELSE S=S-E
100 PSET(N,S):PSET(N,0):GOTO 80
```

Experiment. Ga na voor welke N de som $S_N(\alpha)$ maximaal is in die zin dat

$$S_k(\alpha) < S_N(\alpha) \text{ voor alle } k < N \tag{18.2a}$$

en

$$S_{N+1}(\alpha) < S_N(\alpha). \tag{18.2b}$$

Bereken de noemers van de kettingbreuk-benaderingen van α en tracht experimenteel een verband te vinden tussen de plaatsen waar deze maxima optreden en tracht daaruit (voor bijvoorbeeld $\alpha = \sqrt{2} - 1$) de waarde af te leiden van

$$\limsup_{N \to \infty} S_N(\alpha)/\log(N). \tag{18.3}$$

Soortgelijke experimenten kan men doen door $[n\alpha]$ bijvoorbeeld modulo 4 te beschouwen en na te gaan of zich dan ook van dit soort 'scheefheids' verschijnselen voordoen.

19. PRIEMEN IN DE RING VAN GAUSS

Onder $\mathbb{Z}(i)$, de ring van Gauss, verstaan we de verzameling van alle complexe getallen van de vorm $a + bi$ met $a,b \in \mathbb{Z}$. Een $z \in \mathbb{Z}(i)$ heet een deler van $w \in \mathbb{Z}(i)$ als er een $v \in \mathbb{Z}(i)$ bestaat zodanig dat $zv = w$. Een getal $u \in \mathbb{Z}(i)$ heet een unit als u een deler is van 1; d.w.z. als er een $v \in \mathbb{Z}(i)$ bestaat zodanig dat $uv = 1$. Men gaat gemakkelijk na dat

$$\{1, i, -1, -i\}$$

de verzameling van alle units in $\mathbb{Z}(i)$ is. (Een $u \in \mathbb{Z}(i)$ is dus een unit dan en slechts dan als $|u| = 1$.)

Een $z \in \mathbb{Z}(i)$ heet irreducibel als z geen unit is en als voor elke decompositie $z = vw$, met $v,w, \in \mathbb{Z}(i)$, geldt dat of v of w een unit is. Een $p \in \mathbb{Z}(i)$ heet priem als $p \neq 0$ is, geen unit is en uit $p|vw$ altijd volgt dat of $p|v$ of $p|w$.

Opgave. Vergelijk de zojuist ingevoerde begrippen met de er op lijkende begrippen in $\mathbb{Z}$. Ga na dat in $\mathbb{Z}(i)$ de begrippen irreducibel en priem gelijkwaardig zijn.

Men kan (met behulp van een Euclidisch algoritme) aantonen dat in $\mathbb{Z}(i)$ evenals in $\mathbb{Z}$ eenduidige priemontbinding geldt.

Opgave. Ga na dat $6 \in \mathbb{Z}(i\sqrt{5})$ verschillende priemontbindingen heeft door te observeren dat $6 = 2 \times 3 = (1+i\sqrt{5})(1-i\sqrt{5})$.

Een $z \in \mathbb{Z}(i)$ heet reducibel (samengesteld) als z niet irreducibel is.

Voor elk element van $\mathbb{Z}(i)$ geldt dat het

*) of gelijk aan 0,

*) of een unit,

*) of samengesteld,

*) of priem is.

Dus als $z \in \mathbb{Z}(i)$ niet gelijk aan 0, geen unit en niet samengesteld is, dan is z priem.

Het is triviaal om uit te maken of $z = 0$. Units zijn ook gemakkelijk te herkennen. Als we nu ook nog samengesteldheid van z kunnen uitsluiten, dan is z priem. Samengesteldheid tonen we als volgt aan.

Een $z = a + bi$ is samengesteld als $a^2 + b^2 = (c^2 + d^2)(e^2 + f^2)$ met beide factoren >1. Tracht dit zelf aan te tonen.

Opgave. $z = a + bi$ is priem dan en slechts dan als $a - bi$ priem is.

Opgave. Als $z = a + bi$ en $a + b$ is even en >2 dan is z samengesteld.

Samenvattend komen we zo tot de volgende priemtest in $\mathbb{Z}(i)$.

Programma 31 (PTESTZI)

```
10 'Programma 31 (PTESTZI)
20 CLS:PRINT"PRIEMTEST IN Z(I). We testen de PRIMALITEIT van X + Y*I":PRINT
30 DEFINT A-N
40 W2=SQR(2#)
50 INPUT "X="; M: INPUT "Y="; N
60 X=ABS(M): Y=ABS(N): GOSUB 150
70 IF T$ = "Y" GOTO 120
80 IF XPY = 0 THEN PRINT M; " + i *"; N; "is de NUL": GOTO 50
90 IF XPY = 1 THEN PRINT M; " + i *"; N; "is een UNIT": GOTO 50
100 PRINT "Z ="; M; " + i *"; N; " is NIET priem. ";
110 PRINT "FACTOREN zijn "; A; "+ i *"; B; " en "; U; "+ i * "; V: GOTO 50
120 PRINT "Z ="; M; " + i *"; N; " is PRIEM "
130 GOTO 50
140 '*****  Subroutine PRIEMTEST voor Z(I). Input: X,Y. Output T$.
150 XPY=X+Y: IF XPY < 2 THEN T$="N": RETURN
160 IF ABS(X)=1 AND ABS(Y)=1 THEN T$="Y": RETURN
170 U=XPY/2: V=(X-Y)/2: IF U=INT(U) THEN T$="N": A=1: B=1: RETURN
180 RZ=SQR(X*X+Y*Y+.5): R=SQR(RZ): R2=R*R
190 LA=R/W2
200 FOR A=2 TO LA
210 RA=A: RA2=RA*RA: XA=X*RA: YA=Y*RA
220 FOR B=0 TO A-1
230 RB=B: RN=RA2+RB*RB
240 U=(XA+Y*RB)/RN
250 IF U <> INT(U) GOTO 290
260 V=(YA-X*RB)/RN
270 IF V <> INT(V) THEN GOTO 290
280 T$="N": RETURN
290 NEXT B
300 NEXT A
310 FOR A=LA+1 TO INT(R)
320 'RA etc zijn REALS (wegens PRECISIE) !
330 RA=A: RA2=RA*RA: XA=X*RA: YA=Y*RA
340 FOR B=0 TO SQR(R2-RA2+.5)
350 RB=B: RN=RA2+RB*RB
360 U=(XA+Y*RB)/RN: IF U <> INT(U) GOTO 390
370 V=(YA-X*RB)/RN: IF V <> INT(V) GOTO 390
380 T$="N": RETURN
390 NEXT B
400 NEXT A
410 T$="Y"
420 RETURN
```

Opgave. Toon aan dat er oneindig veel priemen in $\mathbb{Z}(i)$ zijn.

Opgave. Zij $2\mathbb{Z}$ de verzameling van alle even getallen. Merk op dat $2\mathbb{Z}$ geen units bevat. Bevat $2\mathbb{Z}$ irreducibele elementen? Bevat $2\mathbb{Z}$ priemelementen? Beschouw de volgende ontbindingen in $2\mathbb{Z}$: $36 = 2\times18 = 6\times6$. Constateer een verschil tussen irreducibiliteit en primaliteit in het systeem $2\mathbb{Z}$.

20. KWADRAATRESTEN

Een getal a heet een kwadraatrest (kortweg rest) modulo m als $(a,m) = 1$ en de vergelijking

$$x^2 \equiv a \pmod{m} \tag{20.1}$$

oplosbaar is. (Voor de eenvoud zullen we voor m steeds een oneven priemgetal p nemen.) M.a.w., a is een kwadraatrest (mod m) als a een kwadraat is in een gereduceerd restsysteem (mod m).

Voorbeeld. $a = -1$ is een kwadraatrest (mod 13) wegens $5^2 = 25 \equiv -1 \pmod{13}$.

Met behulp van de stelling van Wilson kan men heel gemakkelijk inzien dat $a = -1$ altijd een kwadraatrest (mod p) is als $p = 4n+1$.

DEFINITIE. Het symbool van Legendre $(\frac{a}{p})$ definiëren we voor $(a,p) = 1$ als volgt

$$(\frac{a}{p}) = \begin{cases} 1 \text{ als } a \text{ een kwadraatrest (mod } p) \text{ is} \\ -1 \text{ als } a \text{ geen kwadraatrest (mod } p) \text{ is.} \end{cases} \tag{20.2}$$

Om uit te maken of a een rest (mod p) is, hoeven we dus 'slechts' $(\frac{a}{p})$ te bepalen. Hiervoor gaf Euler het volgende criterium:

$$(\frac{a}{p}) \equiv a^{(p-1)/2} \pmod{p}. \tag{20.3}$$

Het Legendre symbool kan nu snel berekend worden met behulp van de routine POWRES. Hier volgt het programma.

Programma 32 (LEGENDRE)

```
10 'Programma 32 (LEGENDRE)
20 CLS: PRINT"LEGENDRE SYMBOOL (Basis ! Priem)":PRINT
30 PRINT"We hebben GEEN PRIEMTEST ingebouwd !":PRINT
40 DEFDBL D: DEFINT I-N
50 INPUT "Basis ="; NB: INPUT "Priem ="; IP
60 IEXP=(IP-1)/2: IMOD=IP
70 GOSUB 120
80 IF POWRES=1 THEN LEG=1 ELSE LEG=-1
90 PRINT "(";NB;",";IP;") = "; LEG:PRINT
100 GOTO 50
110 END
120 '*****  Subroutine POWER RESIDUE: residue of NB ^ IEXP (mod IMOD)
130 'IF IEXP=0 THEN POWRES=1: RETURN
140 IF IEXP=1 THEN POWRES=NB MOD IMOD: RETURN
150 'IF IMOD=1 THEN POWRES=0: RETURN
160 DB=NB: DB2=DB*DB: HEXP=CSNG(IEXP): DM=CDBL(IMOD)
170 IF HEXP=2*INT(HEXP/2) GOTO 190
180 RES=DB2-DM*INT(DB2/DM): POW=NB MOD IMOD: HEXP=INT(HEXP/2): GOTO 200
190 RES=DB2-DM*INT(DB2/DM): POW=1: HEXP=HEXP/2
200 IF HEXP=1 GOTO 260
210 IF HEXP=2*INT(HEXP/2) GOTO 240
220 DPROD=RES*POW
230 POW=DPROD-DM*INT(DPROD/DM)
240 DPROD=RES*RES
250 RES=DPROD-DM*INT(DPROD/DM): HEXP=INT(HEXP/2): GOTO 200
260 DPROD=RES*POW
270 POWRES=DPROD-DM*INT(DPROD/DM)
280 RETURN
```

Opgave. Hoeveel kwadraatresten zitten er in een gereduceerd restsysteem (mod p)?
(Antwoord: $(p-1)/2$ voor oneven priemen p.)

21. HET TELLEN VAN ROOSTERPUNTEN
Onder een roosterpunt zullen wij hier verstaan een punt (a,b) in het vlak met geheeltallige coordinaten a en b.

Hoeveel roosterpunten liggen er binnen een cirkel om (0,0) met straal $x > 0$? Gebruikmakend van een aantal evidente symmetrie-eigenschappen van de cirkel komen we tot het volgende programma om deze vraag te beantwoorden.

Programma 33 (CIRKEL)

```
10 'Programma 33 (CIRKEL)
20 CLS:PRINT"Telling van roosterpunten in een cirkel om (0,0) met straal R > 0
30 DEFDBL D: DEFINT I-N: DEFDBL O-Z
40 PRINT: INPUT "Voer R (> 0) in"; R
50 IF R <= 0 GOTO 40
60 K=INT(R/SQR(2#)): IR=INT(R): DN=4#*IR+1+4*K^2:K=K+1:D=0
70 IF K>IR GOTO 90 ELSE Q=R*R
80 FOR I=K TO IR: D=D+INT(SQR(Q-I^2)): NEXT I
90 PRINT"R =";R:PRINT"Het aantal roosterpunten bedraagt"; DN+8*D: PRINT
100 PRINT"BENADERING van PI is: aantal roosterpunten / R^2 =";(DN+8*D)/R/R
110 PRINT:GOTO 40
```

Opgave. Maak met behulp van dit programma een schatting van het getal π.

DEFINITIE. Onder $\tau(n)$ verstaan we het aantal natuurlijke delers van $n \in \mathbb{N}$.

Voor $n \in \mathbb{N}$ beschouwen we de hyperbool $y = \frac{n}{x}$, met $x > 0$. Aan de hand van een grafiek make men zich zelf duidelijk dat het aantal roosterpunten op deze kromme gelijk is aan $\tau(n)$. Met deze opmerking in gedachten is het duidelijk dat het aantal roosterpunten in het gebied tussen deze kromme en de beide assen gelijk is aan

$$T(n) := \sum_{0<a\leqslant n} \tau(a). \tag{21.1}$$

Gebruikmakend van de symmetrie van de hyperbool om de lijn $y = x$ komen we tot het volgende programma om $T(n)$ snel uit te rekenen (het tekenen van een plaatje kan geen kwaad!).

Programma 34 (SIGMATAU)

```
10 'Programma 34 (SIGMATAU)
20 CLS:PRINT"Roosterpunt-telling onder hyperbool"
30     PRINT"..................................": PRINT
40 DEFDBL D:DEFINT I-N
50 PRINT:INPUT"Voer een GEHELE N (> 0) in";DN:IF DN<>INT(DN) THEN BEEP:GOTO 50
60 IF DN < 1 GOTO 50
70 IW=INT(SQR(DN+.5)):DK=IW+1:D=0
80 FOR I=1 TO IW:D=D+INT(DN/I):NEXT I
90 D=2*D-IW^2
100 PRINT"R=";DN:PRINT"Het aantal roosterpunten bedraagt"; D
110 PRINT"BENADERING van GAMMA =";(1+(D-DN*LOG(DN))/DN)/2
120 GOTO 50
```

De theorie leert dat

$$T(n) = n\log(n)+(2\gamma-1)n+B(\sqrt{n}) \tag{21.2}$$

waarbij γ de constante van Euler is en $B(x)$ een functie die bij deling door x begrensd is als $x\to\infty$. Tracht nu een schatting te maken van γ met behulp van (21.2) en bovenstaand programma.

22. IS EENDUIDIGE PRIEMONTBINDING VANZELFSPREKEND?

Zoals bekend is priemontbinding in $\mathbb{N}$ eenduidig. Lange tijd heeft men (ook getaltheoretici) zich hierover niet zo druk gemaakt, wellicht omdat men de zaak volkomen vanzelfsprekend vond. (Er wordt wel beweerd dat sommige filosofen deze mening nog steeds toegedaan zijn.) Wel, als het hier inderdaad om een vanzelfsprekende zaak zou gaan, dan zou de eenduidige-priemontbindings-eigenschap toch zo goed als zeker in vrijwel elk algebraïsch systeem moeten gelden. Dat dit niet het geval is, zullen we aan de hand van (een hele klasse van) voorbeelden laten zien. In Davenport's boek, p. 21, wordt het volgende (aan Hilbert toegeschreven) voorbeeld beschreven. Zij H de verzameling van alle natuurlijke getallen van de vorm $4n+1$. Onder een pseudopriem pp verstaan we een getal $pp \in H$ dat niet te schrijven is als het produkt van twee (of meer) elementen uit $H\setminus\{1\}$. De rij van pseudopriemen in H begint als volgt

$$5,9,13,17,21,29,33,37,...$$

Wegens $441 = 9\times49 = 21\times21$, heeft 441 ($\in H$) minstens twee verschillende priemontbindingen (in H). (In DAVENPORT wordt overigens in plaats van 441 het voorbeeld $693 = 9\times77 = 21\times33$ gegeven.) Dus, eenduidige priemontbinding is toch niet zo vanzelfsprekend.

Wat is er nu zo bijzonder aan de rol van het getal 4 in de definitie van H? Kort gezegd, zo goed als niets bijzonders, zoals hieronder zal blijken.

DEFINITIE. Zij v (ontleend aan Vier) een natuurlijk getal. Dan verstaan we onder het bij v behorende Hilbert-systeem H_v de verzameling

$$H_v := \{1,1+v,1+2v,1+3v,1+4v,...\}. \tag{22.1}$$

Onder een H_v-priem verstaan we een getal in H_v dat niet te schrijven is als het produkt van elementen uit $H_v\setminus\{1\}$.

Het is duidelijk dat $H_1 = \mathbb{N}$, zodat we in H_1 eenduidige H_1-priemontbinding hebben. Het is niet moeilijk in te zien dat we ook in H_2 eenduidige H_2-priemontbinding hebben. Ga dit na. Hoe staan de zaken er voor als $v > 2$?

Soortgelijk als voor $\mathbb{N}$ kunnen we een H_v-priemtest voor H_v schrijven. Zie de betreffende subroutine in het volgende programma HILBER2V. De strekking van dit programma is het kleinste element van H_v te vinden dat minstens twee verschillende H_v-priemontbindingen heeft. Zodra gevonden, dan wordt v met 1 opgehoogd en ondergaat het volgende Hilbert-systeem dezelfde test.

Programma 35 (HILBER2V)

```
10 'Programma 35 (HILBER2V)
20 CLS: PRINT "Programma HILBERTV"
30 PRINT"We zoeken het kleinste HILBERT-getal in 1+V, 1+2V, 1+3V, ...":PRINT
40 DIM F(2),P(200):INPUT"Voer een V > 2 in";V:L=5
50 E=1:H=.5:O=0
60 N=E:K=0
70 N=N+V:Z=N:GOSUB 260:IF T=E THEN K=K+E ELSE GOTO 70
80 IF K<=L THEN P(K)=N:GOTO 70
90 PRINT"( v =";V;")","P(";L;") =";P(L):L1=L
100 FOR I=1 TO L:PL2=P(L)^2:IF PL2+1>PL2 THEN GOTO 110 ELSE L=L-1:NEXT I
110 IF L1-L > 4 THEN STOP
120 N=(V+1)*(2*V+1)
130 N=N+V:Z=N:GOSUB 260:IF T=E THEN GOTO 130 ELSE K=O:M=SQR(N+H)
140 FOR I=E TO L
150 D=P(I):IF D>M GOTO 130
160 Q=N/D:IF Q<>INT(Q) GOTO 190
170 Z=Q:GOSUB 260:IF T=O THEN GOTO 190 ELSE K=K+E:F(K)=D
180 IF K>E GOTO 220
190 NEXT I
210 BEEP:L=L+5:GOTO 60
220 PRINT"n= ";N;" is een HILBERT-GETAL bij V =";V
230 PRINT"d1 =";F(1),"q1 =";N/F(1)
240 PRINT"d2 =";F(2),"q2 =";N/F(2):PRINT:V=V+E:GOTO 60
250 '*****  Subroutine PRIEMTEST in een HILBERT-systeem.
260 F=E:X=SQR(Z+H)
270 F=F+V:IF F>X THEN T=E:RETURN
280 G=Z/F:IF G=INT(G) THEN T=O:RETURN ELSE GOTO 270
```

Numeriek blijkt dat H_v voor 'alle' $v > 2$ geen eenduidige H_v-priemontbinding heeft. We geven enkele voorbeelden als testwaarden voor het bovenstaande programma.

$$v = 3 \quad n = 100 = 4\times25 = 10\times10$$

$$v = 4 \quad n = 441 = 9\times49 = 21\times21$$

$$v = 5 \quad n = 336 = 6\times56 = 16\times21$$

$$v = 6 \quad n = 3025 = 25\times121 = 55\times55$$

$$v = 7 \quad n = 792 = 8\times99 = 22\times36$$

$$v = 8 \quad n = 1089 = 9\times121 = 33\times33$$

$$v = 9 \quad n = 1540 = 10\times154 = 28\times55$$

$$v = 10 \quad n = 4641 = 21\times21 = 51\times91.$$

Dat voor $v > 2$ het systeem H altijd elementen bevat die niet eenduidig in H_v-priemen te ontbinden zijn werd onlangs door de spreker vermoed en kort daarna door Te Riele bewezen. Het bewijs is zo eenvoudig dat we het hier presenteren. Bij gegeven $v \in \mathbb{N}$ zoeken we twee ('echte') priemgetallen van de vorm $nv - 1$ en $mv - 1$. (Dit kan volgens de stelling van Dirichlet die zegt dat er in elke aritmetische rij a, $a+b$, $a+2b$,... met $(a,b) = 1$ oneindig veel priemgetallen voorkomen.) Noem deze priemgetallen p en q.
N.B. p en q behoren niet tot H_v. Maar de getallen pq, p^2 en q^2 behoren wel tot H_v evenals

$$n := pq\times pq = p^2\times q^2.$$

Men gaat gemakkelijk na dat pq, p^2 en q^2 (Hilbert-)priem zijn in H_v zodat onze bewering bewezen is.

Het volgende programma berekent bij gegeven v de zojuist besproken getallen p, q en n.

Programma 36 (HILBCOMP)

```
10 'Programma 36 (HILBCOMP)
20 CLS:PRINT"We bepalen een HILBERT-getal in 1+V, 1+2V, 1+3V, ...":PRINT
30 DIM F(2)
40 O=0:E=1:H=.5:T=2:D=3:Z=6
50 INPUT"We beginnen met v =";V
60 N=-E:K=0
70 N=N+V:GOSUB 130:IF W=0 THEN GOTO 70 ELSE K=K+E:F(K)=N
80 IF K<T THEN GOTO 70
90 FE#=CDBL(F(E)):FT#=CDBL(F(T))
100 PRINT"v =";V,"p1 =";F(E),"p2 =";F(T),"n =";(FE#*FT#)^T
110 V=V+1:GOTO 60
120 '***** PRIEMTEST *****
130 IF N=T THEN W=1:RETURN
140 IF T*INT(N/T)=N THEN W=0:RETURN
150 IF N=D THEN W=1:RETURN
160 IF D*INT(N/D)=N THEN W=0:RETURN
170 I=2:F=1:X=INT(SQR(N+H))
180 I=Z-I:F=F+I:IF F>X THEN W=1:RETURN
190 Q=N/F:IF Q=INT(Q) THEN W=0:RETURN ELSE GOTO 180
```

Opgave. Genereer de kleinste 'echte' priem in H_v.

Programma 37 (PARITHM)

```
10 'Programma PARITHM
20 CLS:O=0:E=1:T=2:D=3:Z=6
30 INPUT"We beginnen met V =";V
40 PRINT"FORMAT (V, P, K)"
50 M=0:S=T
60 N=E:K=0
70 N=N+V:K=K+E:GOSUB 110:IF W=0 GOTO 70 ELSE S=S+K
80 IF K>M THEN M=K:PRINT"(";V;",";N;",";K;")";" K GEMIDDELD =";S/V
90 V=V+E:GOTO 60
100 '***** PRIEMTEST *****
110 IF T*INT(N/T)=N THEN W=0:RETURN
120 IF D*INT(N/D)=N THEN W=0:RETURN
130 I=T:F=E:X=INT(SQR(N))
140 I=Z-I:F=F+I:IF F>X THEN W=E:RETURN
150 Q=N/F:IF Q=INT(Q) THEN W=0:RETURN ELSE GOTO 140
```

Verdere aanbevolen literatuur.

L.É. DICKSON, *History of the theory of Numbers*, 3 Vols, Chelsea (1971).
H. GRIFFIN, *Elementary Theory of Numbers*, McGraw-Hill (1954).
E. GROSSWALD, *Topics from the theory of Numbers*, Birkhäuser (1984).
R.K. GUY, *Unsolved Problems in Number Theory*, Springer (1981).
G.H. HARDY & E.M. WRIGHT, *An Introduction to the Theory of Numbers*, Clarendon Press (1968).
HUA LOO KENG, *Introduction to Number Theory*, Springer (1982).
E. LANDAU, *Elementary Number Theory*, Chelsea (1966).
W.J. LEVEQUE, *Fundamentals of Number Theory*, Addison-Wesley (1977).

H.G. MEIJER, *Cursus Getaltheorie*, Mathematisch Centrum, Rapport ZC 78 (1971).
L.J. MORDEL, *Diophantine Equations*, Academic Press (1969).
T. NAGELL, *Introduction to Number Theory*, John Wiley (1951).
W. NARKIEWICZ, *Number Theory*, World Scientific (1983).
D. NIVEN, *Irrational Numbers*, John Wiley (1967).
O. PERRON, *Irrationalzahlen*, Göschens Lehrbücherei, Band 1, Walter de Gruyter (1947).
H. RADEMACHER, *Lectures on Elementary Number Theory*, Krieger Publishing Company (1977).
K.H. ROSEN, *Elementary Number Theory and its Applications*, Addison-Wesley (1984).
A. SCHOLZ & B. SCHOENEBERG, *Einführung in die Zahlentheorie*, Sammlung Göschen, Band 1131, Walter de Gruyter (1961).
H.M. STARK, *An Introduction to Number Theory*, Markham (1970).
D. SHANKS, *Solved and Unsolved Problems in Number Theory*, Chelsea (1978).
H.N. SHAPIRO, *Introduction to the Theory of Numbers*, John Wiley (1983).
B.M. STEWART, *Theory of Numbers*, Macmillan (1964).

Numeriek Rekenen op een PC

H.J.J. te Riele
Centrum voor Wiskunde en Informatica
Kruislaan 413, 1098 SJ Amsterdam

0. Inleiding

Veel technisch-wetenschappelijke problemen kunnen wiskundig gemodelleerd worden tot b.v. een gewone of partiële differentiaalvergelijking, een eigenwaardeprobleem, een vergelijking waarvan een wortel bepaald moet worden, een (oppervlakte-)integraal, enz. Dit soort wiskundige problemen kunnen vaak niet met *analytische* middelen worden opgelost. In zulke gevallen is men genoodzaakt zijn toevlucht te nemen tot *numerieke* methoden én een snelle computer om de vaak zeer gecompliceerde berekeningen uit te voeren. Hierbij wordt veelvuldig gebruik gemaakt van standaardprogrammatuur die op veel grote computersystemen beschikbaar is (zoals de NAG- en de IMSL-Libraries; zie b.v. [1]). De belangrijkste versies van deze programmabibliotheken zijn geschreven in de programmeertaal FORTRAN, maar NAG levert ook Pascal- en ALGOL68-versies.

Hoewel de belangrijke 'real-life' problemen vaak alleen op de grootste en snelste computers kunnen worden doorgerekend, kunnen ook Personal Computers een nuttige rol spelen bij het oplossen van numerieke problemen: als eerste kennismaking met de problematiek van het numeriek rekenen of ook als middel om snel een eerste indruk te krijgen van het probleem dat (op een snelle computer) dient te worden opgelost. Langzamerhand verschijnen er dan ook PC-versies van de grote programmabibliotheken, zoals de NAG PC-50 Library; een selectie van 50 van de meest gebruikte FORTRAN-routines.

In deze bijdrage aan de vakantiecursus 1987 zullen we aan de hand van voorbeelden enkele standaard numerieke problemen de revue laten passeren. Daarbij maken we gebruik van een op diskette verkrijgbare bibliotheek van numerieke routines in Pascal, zoals beschreven in [2] en [3].

Als PC gebruiken we een Olivetti M24 Personal Computer met MS-DOS als Operating System. Voor het compileren en executeren van de Pascal-program-

ma's gebruiken we Turbo Pascal, zoals beschreven in [4].

In alle voorbeeldprogramma's uit [3] wordt gebruik gemaakt van een file MODFILE.PAS waarin

(i) de PROCEDURE glopen voor het openen van een externe file en het uitvoeren van een 'reset' wordt gedefinieerd;
(ii) het type double en de functie sngl worden gedefinieerd.

Voor Turbo Pascal op IBM en IBM-compatible PCs ziet MODFILE.PAS er als volgt uit:

```
TYPE double = real; string10 = string[10];
FUNCTION sngl(x:real):real; BEGIN sngl := x END;
PROCEDURE glopen(VAR infile:text; filename:string10);
    BEGIN assign(infile,filename); reset(infile) END;
```

Met behulp van de include-opdracht

```
(*$I MODFILE.PAS *)
```

wordt bij compilatie van een voorbeeldprogramma de file MODFILE.PAS automatisch in het Pascal-programma ingevoegd en meegecompileerd.

Als voorbeeld van het werken met Turbo Pascal beschouwen we het volgende programmaatje, dat nagaat wat het kleinste positieve gehele getal i is waarvoor de M24 het getal $1+2^{-i}$ niet meer van 1 kan onderscheiden:

```
PROGRAM ch(input,output);
(* check accuracy *)
LABEL 10;
VAR a: real;
    i: integer;
BEGIN
   i:=0;
   a:=1.0;
10:
   a:=a/2.0;
   i:=i+1;
   IF 1+a<>1 THEN GOTO 10;
   writeln(i:3,a:30:20)
END.
```

Dit programmaatje staat op file CHACC.PAS. Na het laden van de compiler met het command TURBO verschijnt het volgende 'menu' op het scherm:

```
Logged drive: A
Active directory: \

Work file:
Main file:

Edit      Compile  Run   Save

Dir       Quit  compiler Options

Text:      0 bytes
Free: 63485 bytes
```

Toetsen we nu R, de eerste letter van Run, in, dan antwoordt de computer met:

```
Work file name:
```

Na intoetsen van CHACC.PAS zien we verschijnen:

```
Loading A:\CHACC.PAS
Compiling
  14 lines

Code:        000B paragraphs (    176 bytes), 0D1D paragraphs free
Data:        0003 paragraphs (     48 bytes), 0FD9 paragraphs free
Stack/Heap: 8B19 paragraphs (569744 bytes)

Running
 40         0.0000000000090949470
```

Conclusie: de computer kan $1+2^{-40}$ niet meer van 1 onderscheiden, dus de nauwkeurigheid van de berekeningen bedraagt ongeveer 11 à 12 decimale cijfers.

1. VOORBEELDEN VAN HET GEBRUIK VAN EEN NUMERIEKE PC-LIBRARY

1.1. De integraal van een functie

PROBLEEM. Te berekenen (een benadering van) de integraal

$$I = \int_a^b f(x)dx,$$

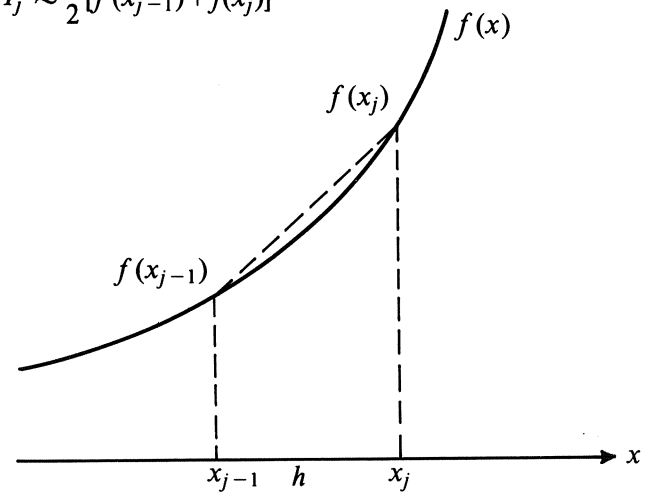
waarbij f een gegeven functie en $[a,b]$ een gegeven interval is.

METHODE. Een simpele, veel gebruikte methode is de *herhaalde trapeziumregel.* Hierbij wordt het interval $[a,b]$ in N gelijke stukken $[x_0,x_1]$, $[x_1,x_2]$, $\cdots$, $[x_{N-1},x_N]$ verdeeld, waarbij $a=x_0$, $b=x_N$; $x_i=a+ih$, $i=0,1,\cdots,N$; $h=(b-a)/N$. Er geldt dan

$$I = \sum_{j=1}^{N} I_j, \quad \text{met} \quad I_j = \int_{x_{j-1}}^{x_j} f(x)dx.$$

We benaderen I_j nu met behulp van de trapeziumregel (zie fig. 1):

$$I_j \approx \frac{h}{2}[f(x_{j-1})+f(x_j)]$$



FIGUUR 1

Hieruit volgt voor I de benadering:

$$I \approx \frac{h}{2}[f(a)+f(b)]+h\sum_{j=1}^{N-1} f(x_j).$$

Als de functie f tweemaal continu differentieerbaar is, dan is de gemaakte fout evenredig met h^2, voor $h\to 0$ $(N\to\infty)$. Het voordeel van de herhaalde trapeziumregel is, dat deze verfijnd kan worden door het aantal punten te verdubbelen zonder dat het eerder gedane werk verloren gaat. Als we met I^h de benadering van I aanduiden die verkregen wordt door de herhaalde

trapeziumregel met stap h toe te passen, dan is het niet moeilijk in te zien dat

$$I^{h/2} = \tfrac{1}{2}I^h + \frac{h}{2}\sum_{j=1}^{N} f(a + \frac{2j-1}{2}h), \quad \text{met} \quad N = (b-a)/h.$$

Kiezen we b.v. $f(x)=4x^3$, $[a,b]=[0,1]$, dan is $I=1$ en we vinden:

$h(N)$	I^h	$I^h - I$	quotiënt opeenvolgende fouten
1 (1)	2	1	
			4
1/2 (2)	1.25	0.25	
			4
1/4 (4)	1.0625	0.0625	
			4
1/8 (8)	1.015625	0.015625	
			4
1/16 (16)	1.00390625	0.00390625	

Zoals te verwachten is, geeft halvering van de stap h een verkleining van de fout met een factor 4 (in feite is de fout $I^h - I$ exact gelijk aan h^2).

Pascal routines

PROCEDURE trapzd (*a,b*: real; VAR *s*: real; *n*: integer);

Deze procedure berekent de *n*de verfijningsstap van de herhaalde trapeziumregel waarbij in *s* het resultaat van de $(n-1)$de stap als invoer wordt meegegeven en ook in *s* het resultaat van de *n*de stap wordt afgeleverd. Voor $n=1$ wordt de eenvoudigste benadering van I in *s* afgeleverd. Bij verdere aanroepen, voor $n=2,3,\cdots$, worden 2^{n-2} inwendige punten in de herhaalde trapeziumregel toegevoegd. Het programma waarin trapzd wordt aangeroepen, moet een functie func (*x*: real): real meegeven die door trapzd moet worden geintegreerd.

PROCEDURE qtrap(*a,b*: real; VAR *s*: real);

Gebruikmakend van de procedure trapzd levert deze procedure in *s* een benadering van de integraal van de functie func van *a* tot *b* af, met een relatieve nauwkeurigheid van ongeveer *eps*. Dit is een constante in de procedure, die van te voeren moet worden ingesteld.

Voorbeeldprogramma

Berekening van $\int_0^{\pi/2} x^2(x^2-2)\sin x\,dx$ met behulp van procedure qtrap met een nauwkeurigheid van ongeveer 10^{-6}.

Programma :

```
PROGRAM d4r2(input,output);
(* driver for routine QTRAP *)
CONST
   pio2=1.5707963;
VAR
   glit : integer;
   a,b,s : real;

FUNCTION func(x : real) : real;
(* Test function *)
BEGIN
   func := sqr(x)*(sqr(x)-2.0)*sin(x)
END;

FUNCTION fint(x : real) : real;
(* Integral of test function *)
BEGIN
   fint := 4.0*x*(sqr(x)-7.0)*sin(x)-
         (sqr(sqr(x))-14.0*sqr(x)+28.0)*cos(x);
END;

(*$I MODFILE.PAS *)
(*$I TRAPZD.PAS *)

(*$I QTRAP.PAS *)

BEGIN
   a := 0.0;
   b := pio2;
   writeln ('Integral of func computed with QTRAP');
   writeln;
   writeln ('Actual value of integral is',fint(b)-fint(a):18:12);
   qtrap(a,b,s);
   writeln ('Result from routine QTRAP is',s:18:12);
END.
```

Output :

```
Integral of func computed with QTRAP

Actual value of integral is    -0.479158841040

 Call of QTRAP with eps = 1.00E-06
    Call of TRAPZD  1 s=     0.9057727802
    Call of TRAPZD  2 s=    -0.0209449905
    Call of TRAPZD  3 s=    -0.3614613180
    Call of TRAPZD  4 s=    -0.4495837621
    Call of TRAPZD  5 s=    -0.4717563216
    Call of TRAPZD  6 s=    -0.4773076746
    Call of TRAPZD  7 s=    -0.4786960160
    Call of TRAPZD  8 s=    -0.4790431327
    Call of TRAPZD  9 s=    -0.4791299138
    Call of TRAPZD 10 s=    -0.4791516093
    Call of TRAPZD 11 s=    -0.4791570332
    Call of TRAPZD 12 s=    -0.4791583893
    Call of TRAPZD 13 s=    -0.4791587282

Result from routine QTRAP is    -0.479158728240
```

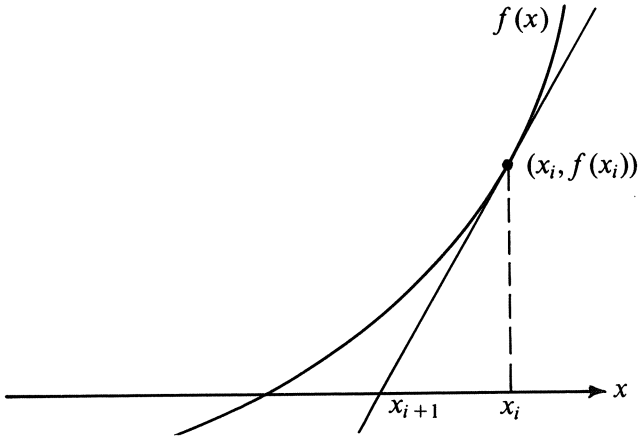
1.2. Bepaling van een oplossing van de vergelijking $f(x)=0$

PROBLEEM. Te berekenen een (benadering van een) oplossing van de vergelijking

$$f(x) = 0,$$

waarbij f een gegeven functie is.

METHODE. De bekendste methode is die van Newton-Raphson. Hierbij is het wel noodzakelijk dat we naast f ook f' kunnen evalueren. Meetkundig beschouwd bestaat de methode hieruit dat de raaklijn in een bepaald punt $(x_i, f(x_i))$ van de kromme $y=f(x)$ wordt getrokken en dat het snijpunt met de x-as als nieuwe schatting x_{i+1} van de oplossing van de vergelijking $f(x)=0$ wordt opgevat (zie fig. 2).



FIGUUR 2

Het snijpunt van de raaklijn $y=f(x_i)+(x-x_i)f'(x_i)$ met de x-as is $x_i-f(x_i)/f'(x_i)$, dus het Newton-Raphson-proces kan worden beschreven door

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}, \quad \text{mits} \quad f'(x_i)\neq 0, \quad i=0,1,\cdots.$$

Hierbij moet de startwaarde x_0 dicht genoeg bij een wortel van de vergelijking $f(x)=0$ liggen, opdat het (iteratieve) proces convergeert. *Als* het proces convergeert, dan convergeert het kwadratisch, d.w.z. dat het aantal goede cijfers in iedere iteratiestap ongeveer verdubbeld wordt. Anders uitgedrukt: als $x=x^*$ een oplossing is van de vergelijking $f(x)=0$ en $\epsilon_i := x_i - x^*$, dan geldt:

$$\epsilon_{i+1} \approx \epsilon_i^2 \frac{f''(x^*)}{2f'(x^*)}.$$

Bij het zoeken naar een eerste benadering van een oplossing van $f(x)=0$ is het gebruikelijk om eerst twee punten a en b te zoeken waarvoor $f(a)$ en $f(b)$

verschillend teken hebben. Dit betekent dan dat er tussen a en b minstens één nulpunt van f ligt, mits f continu is op $[a,b]$.

Pascal routines

FUNCTION rtsafe ($x1, x2$, xacc: real): real;

Deze function bepaalt m.b.v. de Newton-Raphson-methode een wortel van de vergelijking $f(x)=0$ in het interval $[x1, x2]$. Het iteratieproces wordt afgebroken als $|f(x_i)/f'(x_i)|<$xacc. Programma's waarin rtnewt wordt aangeroepen, moeten een procedure funcd (x,f,df: real) bevatten die de functiewaarde $f(x)$ in f en de afgeleide $f'(x)$ in df aflevert. Er is een 'veiligheidsklep' ingebouwd die voorkomt dat een Newton-Raphson-stap een waarde oplevert, die *buiten* het op dat moment ingeperkte interval, waar een oplossing moet liggen, ligt. In deze situatie wordt de Newton-Raphson-stap niet uitgevoerd, maar vervangen door een *bisectie*-stap: $f(x)$ wordt uitgerekend voor x in het midden van het betrokken interval en het iteratieproces wordt, afhankelijk van het teken van deze nieuw berekende waarde, op de linker- of rechterhelft van het betrokken interval voortgezet.

Voorbeeldprogramma
Bepaling van alle wortels van de Bessel-functie $J_0(x)$ in het interval [1, 10]. Dit interval wordt eerst in 10 gelijke stukken verdeeld en op alle intervallen waar een tekenwisseling wordt geconstateerd, wordt een wortel bepaald.

Programma :

```
PROGRAM d9r9(input,output);
(* driver for routine RTSAFE *)
CONST
   n=10;
   nbmax=20;
   x1=1.0;
   x2=10.0;
TYPE
   glnbmax = ARRAY [1..nbmax] OF real;
VAR
   i,nb : integer;
   root,xacc : real;
   xb1,xb2 : glnbmax;

(*$I MODFILE.PAS *)
(*$I BESSJ0.PAS *)

(*$I BESSJ1.PAS *)

FUNCTION fx(x: real): real;
BEGIN
   fx := bessj0(x)
END;

PROCEDURE funcd(x: real; VAR fn,df: real);
BEGIN
   fn := fx(x);
   df := -bessj1(x);
   writeln(' call of funcd; x,f,df: ',
           x:10:6,fn:10:6,df:10:6)
END;

(*$I ZBRAK.PAS *)

(*$I RTSAFE.PAS *)

BEGIN
   nb := nbmax;
   zbrak(x1,x2,n,xb1,xb2,nb);
   writeln;
   writeln('roots of bessj0:');
   FOR i := 1 to nb DO BEGIN
      xacc := (1.0e-6)*(xb1[i]+xb2[i])/2.0;
      root := rtsafe(xb1[i],xb2[i],xacc);
      writeln(' root',i:2,':',root:9:6,
              ' f(root)=',fx(root):9:6)
   END
END.
```

Output :

```
roots of bessj0:
 call of funcd; x,f,df:    1.900000  0.281819 -0.581157
 call of funcd; x,f,df:    2.800000 -0.185036 -0.409709
 call of funcd; x,f,df:    2.350000  0.028778 -0.530467
 call of funcd; x,f,df:    2.404250  0.000299 -0.519272
 call of funcd; x,f,df:    2.404825  0.000000 -0.519148
 root 1: 2.404826 f(root)= 0.000000
 call of funcd; x,f,df:    5.500000 -0.006844  0.341438
 call of funcd; x,f,df:    6.400000  0.243311  0.181638
 call of funcd; x,f,df:    5.950000  0.136570  0.286222
 call of funcd; x,f,df:    5.950000  0.136570  0.286222
 call of funcd; x,f,df:    5.725000  0.067986  0.321137
 call of funcd; x,f,df:    5.513294 -0.002310  0.340676
 call of funcd; x,f,df:    5.520074 -0.000001  0.340265
 root 2: 5.520078 f(root)=-0.000000
 call of funcd; x,f,df:    8.200000  0.122215 -0.257999
 call of funcd; x,f,df:    9.100000 -0.114239 -0.232431
 call of funcd; x,f,df:    8.650000  0.001012 -0.271567
 call of funcd; x,f,df:    8.653727  0.000000 -0.271452
 root 3: 8.653728 f(root)= 0.000000
```

1.3. Bepaling van de wortels van een polynoom

PROBLEEM. Te berekenen de n wortels van een $n$de graads polynoom $P_n(x)$ met complexe coëfficiënten.

METHODE. De methode die we hier zullen bespreken, is de methode van Laguerre. Om deze te kunnen toepassen, is het nodig om complex te kunnen rekenen: als het polynoom zelf complexe coëfficiënten heeft, is dit natuurlijk niet te vermijden. Achtergrond van Laguerre's methode zijn de volgende relaties:

$$P_n(x) = (x-x_1)(x-x_2)\cdots(x-x_n)$$

$$\ln|P_n(x)| = \ln|x-x_1|+\ln|x-x_2|+\cdots+\ln|x-x_n|$$

$$\frac{d}{dx}\ln|P_n(x)| = \frac{1}{x-x_1}+\frac{1}{x-x_2}+\cdots+\frac{1}{x-x_n} = \frac{P'_n}{P_n} =: G$$

$$-\frac{d^2}{dx^2}\ln|P_n(x)| = \frac{1}{(x-x_1)^2}+\frac{1}{(x-x_2)^2}+\cdots+\frac{1}{(x-x_n)^2}$$

$$= \left[\frac{P_n'}{P_n}\right]^2 - \frac{P_n''}{P_n} =: H.$$

De wortel x_1 wordt nu verondersteld op afstand a van een schatting x van x_1 te liggen, terwijl *alle andere wortels* op afstand b liggen:

$$a = x-x_1, \quad b = x-x_i,\ i=1,2,3,\cdots,n.$$

Dan geldt voor G en H:

$$\frac{1}{a}+\frac{n-1}{b} = G \quad \text{en} \quad \frac{1}{a^2}+\frac{n-1}{b^2} = H,$$

en hieruit volgt voor a:

$$a = \frac{n}{G \pm \sqrt{(n-1)(nH-G^2)}},$$

waarbij dát teken wordt gekozen dat de grootste waarde voor de noemer oplevert. Aangezien $nH-G^2<0$ kan zijn, kan a complex zijn.

De methode werkt iteratief: voor een beginschatting x wordt a uitgerekend en $x-a$ is dan de volgende schatting. Dit wordt voortgezet totdat a voldoende klein is geworden. De dan gevonden benadering x^* van de wortel kan vervolgens uit het polynoom $P_n(x)$ worden verwijderd door $P_n(x)$ te delen door $x-x^*$. Dit wordt *deflatie* genoemd. Als quotiënt krijgen we dan een polynoom van de graad $n-1$ en daar kan Laguerre's methode dan weer op worden toegepast. Het is van groot belang er rekening mee te houden dat de coëfficiënten van het quotiëntpolynoom minder nauwkeurig zijn dan die van het oorspronkelijke polynoom. Dit kan de nauwkeurigheid van de wortel van het quotiëntpolynoom nadelig beïnvloeden. Om de invloed van foutvorming t.g.v. deflatie te verminderen, is het gebruikelijk alle wortels die na het toepassen van deflatie m.b.v. Laguerre's methode zijn gevonden bij te 'polijsten' met behulp van, weer, Laguerre's methode, maar dan toegepast op het oorspronkelijke polynoom $P_n(x)$.

Pascal routines

```
PROCEDURE laguer (a: glcarray; m: integer;
  VAR x: gl2array; eps: real; polish: boolean);
```

In het programma waarin laguer wordt aangeroepen, moeten de types

```
TYPE glcarray = ARRAY [1 . . 2*m+2] of real;
     gl2array = ARRAY [1 . . 2] of real;
```

gedefinieerd zijn. Procedure laguer berekent, voor een in gl2array x gegeven beginschatting, voor in glcarray a gegeven $m+1$ complexe coëfficiënten $a(i)$ van het polynoom

$$P_m(x) = \sum_{i=1}^{m+1} a(i)x^{i-1},$$

een wortel van $P_m(x)$ met behulp van de methode van Laguerre. De relatieve nauwkeurigheid is *eps*. Voor normaal gebruik krijgt polish de waarde false. Als polish de waarde true krijgt, wordt x verbeterd tot machineprecisie, aangenomen dat x al een goede beginschatting van een wortel is.

```
PROCEDURE zroots (a: glcarray; m: integer;
  VAR roots: glcarray; polish: boolean).
```

Dit is de 'driver' routine die laguer voor iedere wortel aanroept, de deflatie

uitvoert, de wortels polijst m.b.v. dezelfde routine laguer en tenslotte de wortels sorteert naar hun reële deel. Als polijsten gewenst is, moet polish de waarde true, anders false krijgen.

Voorbeeldprogramma
Het polynoom $P_4(x)=x^4-(1+2i)x^2+2i$ heeft de vier wortels $x=\pm 1$, $x=\pm(1+i)$. Met behulp van procedure zroots worden eerst de 4 wortels van $P_4(x)$ berekend. Vervolgens worden alle wortels met een factor 1.01 'verstoord' en daarna met polish = true door zroots gepolijst.

Programma :

```
PROGRAM d9r11(input,output);
(* driver for routine ZROOTS *)
CONST
   m=4;
   twomp2=10;   (* twomp2=(2*m+2) *)
TYPE
   glcarray = ARRAY [1..twomp2] OF real;
   gl2array = ARRAY [1..2] OF real;
VAR
   i : integer;
   polish : boolean;
   a,roots : glcarray;

(*$I MODFILE.PAS *)
(*$I LAGUER.PAS *)

(*$I ZROOTS.PAS *)

BEGIN
   a[1] := 0.0; a[2] := 2.0;
   a[3] := 0.0; a[4] := 0.0;
   a[5] := -1.0; a[6] := -2.0;
   a[7] := 0.0; a[8] := 0.0;
   a[9] := 1.0; a[10] := 0.0;
   writeln('Roots of the polynomial x^4-(1+2i)*x^2+2i');
   polish := false;
   zroots(a,m,roots,polish);
   writeln;
   writeln('Unpolished roots:');
   writeln('root #':14,'real':13,'imag.':13);
   FOR i := 1 to m DO BEGIN
      writeln(i:11,' ':5,roots[2*i-1]:14:9,roots[2*i]:14:9)
   END;
   writeln;
   writeln('Corrupted roots:');
   FOR i := 1 to m DO BEGIN
      roots[2*i-1] := roots[2*i-1]*(1+0.01*i);
      roots[2*i] := roots[2*i]*(1+0.01*i)
   END;
   writeln('root #':14,'real':13,'imag.':13);
   FOR i := 1 to m DO BEGIN
      writeln(i:11,' ':5,roots[2*i-1]:14:9,roots[2*i]:14:9)
   END;
   polish := true;
   zroots(a,m,roots,polish);
   Writeln;
   writeln('Polished roots:');
   writeln('root #':14,'real':13,'imag.':13);
   FOR i := 1 to m DO BEGIN
      writeln(i:11,' ':5,roots[2*i-1]:14:9,roots[2*i]:14:9)
   END
END.
```

Output :

```
Roots of the polynomial x^4-(1+2i)*x^2+2i

Unpolished roots:
        root #          real          imag.
          1       -1.000000000  -1.000000000
          2       -1.000000000   0.000000000
          3        1.000000000   1.000000000
          4        1.000000000   0.000000000

Corrupted roots:
        root #          real          imag.
          1       -1.010000000  -1.010000000
          2       -1.020000000   0.000000000
          3        1.030000000   1.030000000
          4        1.040000000   0.000000000

Polished roots:
        root #          real          imag.
          1       -1.000000000  -1.000000000
          2       -1.000000000   0.000000000
          3        1.000000000   1.000000000
          4        1.000000000   0.000000000
```

1.4. Oplossen van een stelsel lineaire vergelijkingen

PROBLEEM. Te bepalen de oplossingsvector $x=(x_1, x_2, \cdots, x_n)^T$ van de matrix-vector-vergelijking

$$Ax = b,$$

waarbij $A=(a_{ij})$ een matrix van orde n is en $b=(b_1, b_2, \cdots, b_n)^T$ een gegeven rechterlidvector.

METHODE. Stel dat we de matrix A kunnen schrijven als het produkt van twee matrices L en U:

$$A = LU$$

waarbij $L=(l_{ij})$ een onderdriehoeksmatrix is met enen op de hoofddiagonaal en $U=(u_{ij})$ een bovendriehoeksmatrix. Een dergelijke decompositie kan direct gebruikt worden bij het oplossen van het stelsel

$$Ax = (LU)x = L(Ux) = b$$

door eerst de vector y te bepalen waarvoor geldt

$$Ly = b$$

en vervolgens de vector x uit de vergelijking

$$Ux = y.$$

De laatste twee stelsels kunnen, omdat L en U *driehoeksmatrices* zijn, direct met behulp van voorwaartse, resp. achterwaartse substitutie worden opgelost.

Voor een stelsel van de orde 3 ziet de decompositie $A = LU$ er als volgt uit:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}.$$

De elementen van L en U kunnen worden berekend volgens de z.g. methode van Crout. Ieder element a_{ij} van A kan worden geschreven als het inwendig produkt van rij i van L en kolom j van U:

$$a_{ij} = \sum_{k=1}^{\min(i,j)} l_{ik} u_{kj}, \quad i,j = 1,2, \cdots, n.$$

Als we nu deze n^2 vergelijkingen opschrijven in de volgorde: te beginnen met a_{11} en vervolgens van boven naar beneden de overige elementen van de eerste kolom van A, vervolgens op dezelfde wijze de elementen van de tweede kolom van A en zo verder tot en met de laatste kolom van A, dan blijkt dat er steeds *precies één onbekende* in elke vergelijking voorkomt. Om precies te zijn: dit is u_{ij} als we de bovenstaande vergelijking met a_{ij} beschouwen voor $i \leqslant j$, of l_{ij} voor $i > j$. Op het moment dat een element u_{ij} of l_{ij} is uitgerekend, kan dit op de overeenkomstige plaats van a_{ij} worden opgeborgen omdat deze a_{ij} in de verdere berekeningen van de LU-decompositie niet meer nodig zijn. Na de berekening van de LU-decompositie is het array waarin A stond opgeborgen, overschreven met de elementen van L en U, als volgt:

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ l_{21} & u_{22} & u_{23} \\ l_{31} & l_{32} & u_{33} \end{bmatrix}.$$

De vergelijkingen waaruit de l_{ij} en u_{ij} zijn berekend, luiden in het orde 3 geval:

$$\left.\begin{aligned} u_{11} &= a_{11} \\ l_{21} &= a_{21} / u_{11} \\ l_{31} &= a_{31} / u_{11} \end{aligned}\right\} \text{1ste kolom}$$

$$\left.\begin{aligned} u_{12} &= a_{12} \\ u_{22} &= a_{22} - l_{21} u_{12} \\ l_{32} &= (a_{32} - l_{31} u_{12}) / u_{22} \end{aligned}\right\} \text{2de kolom}$$

$$\left.\begin{aligned} u_{13} &= a_{13} \\ u_{23} &= a_{23} - l_{21} u_{13} \\ u_{33} &= a_{33} - l_{31} u_{13} - l_{32} u_{23} \end{aligned}\right\} \text{3de kolom.}$$

Een verfijning van deze procedure zorgt er voor dat bepaalde rijen in het proces worden verwisseld, waardoor de delingen door u_{ii} (bij de berekening van de $i$de kolom van matrix L, van $i = 1,2, \cdots, n-1$) met zo groot mogelijke

waarden worden uitgevoerd. Dit proces heet 'partieel pivoten' en verhoogt de numerieke stabiliteit van het proces. Van belang is nog op te merken dat als eenmaal een LU-decompositie van A gevonden is, deze gebruikt kan worden voor het oplossen van het stelsel $Ax = b$ voor *verschillende* rechterleden b.

Pascal routines

PROCEDURE ludcmp (VAR *a*: glnpbynp; *n*, *np*: integer;
VAR indx: glindx; VAR *d*: real);

Deze procedure overschrijft een gegeven $n \times n$ matrix a, opgeslagen in een grotere matrix met fysieke dimensies $np \times np$, door de LU-decompositie van een rij-gewijze permutatie van a zelf. De output-vector indx bevat informatie over de permutatie t.g.v. het partieel pivoteren. De output-variabele d krijgt de waarde +1 of -1 afhankelijk van het feit of het *aantal* rij-verwisselingen even of oneven was.

PROCEDURE lubksb (*a*: glnpbynp; *n*, *np*: integer;
indx: glindx; VAR *b*: glnarray);

Deze procedure lost op het stelsel $ax = b$, met als invoer niet a maar de LU-decompositie van a, verkregen met behulp van ludcmp. De door ludcmp opgeleverde permutatievector staat in indx. De gegeven rechterlidvector b wordt door lubksb overschreven met de oplossingsvector x.

Voorbeeldprogramma
Enkele stelsels lineaire vergelijkingen worden m.b.v. procedures ludcmp en lubksb opgelost. De data staan gegeven op file MATRX1.DAT.

Programma:

```
PROGRAM d2r3(input,output,dfile);
(* driver for routine LUBKSB *)
LABEL 10,99;
CONST
   np=20;
TYPE
   glnpbynp=ARRAY [1..np,1..np] OF real;
   glnarray=ARRAY [1..np] OF real;
   glindx=ARRAY [1..np] OF integer;
VAR
   j,k,l,m,n : integer;
   p : real;
   a,b,c : glnpbynp;
   indx : glindx;
   x : glnarray;
   dfile : text;

(*$I MODFILE.PAS *)
(*$I LUDCMP.PAS *)

(*$I LUBKSB.PAS *)

BEGIN
   glopen(dfile,'matrx1.dat');
10:   readln(dfile);
   readln(dfile);
   readln(dfile,n,m);
   readln(dfile);
   FOR k := 1 to n DO BEGIN
      FOR l := 1 to n-1 DO read(dfile,a[k,l]);
      readln(dfile,a[k,n])
   END;
   readln(dfile);
   FOR l := 1 to m DO BEGIN
      FOR k := 1 to n-1 DO read(dfile,b[k,l]);
      readln(dfile,b[n,l])
   END;
(* save matrix a for later testing *)
   FOR l := 1 to n DO BEGIN
      FOR k := 1 to n DO BEGIN
         c[k,l] := a[k,l]
      END
   END;
(* do lu decomposition *)
   ludcmp(c,n,np,indx,p);
(* solve equations for each right-hand vector *)
   FOR k := 1 to m DO BEGIN
      FOR l := 1 to n DO BEGIN
         x[l] := b[l,k]
      END;
```

Vervolg programma:

```
      lubksb(c,n,np,indx,x);
(* test results with original matrix *)
      writeln('right-hand side vector:');
      FOR l := 1 to n-1 DO write(b[l,k]:12:6);
      writeln(b[n,k]:12:6);
      writeln('solution vector:');
      FOR l := 1 to n-1 DO write(x[l]:12:6);
      writeln(x[n]:12:6);
      writeln ('result of matrix applied to sol''n vector');
      FOR l := 1 to n DO BEGIN
         b[l,k] := 0.0;
         FOR j := 1 to n DO BEGIN
            b[l,k] := b[l,k]+a[l,j]*x[j]
         END
      END;
      FOR l := 1 to n-1 DO write(b[l,k]:12:6);
      writeln(b[n,k]:12:6);
      writeln('**********************************')
   END;
   IF eof(dfile) THEN GOTO 99;
   writeln('press RETURN for next problem:');
   readln;
   GOTO 10;
99:   close(dfile)
END.
```

Input:

```
NEXT PROBLEM:
Size of matrix (NxN), Number of solutions:
5 2
Matrix A:
1.4 2.1 2.1 7.4 9.6
1.6 1.5 1.1 0.7 5.0
3.8 8.0 9.6 5.4 8.8
4.6 8.2 8.4 0.4 8.0
2.6 2.9 0.1 9.6 7.7
Solution vectors:
1.1 1.6 4.7 9.1 0.1
4.0 9.3 8.4 0.4 4.1
```

Output:

```
right-hand side vector:
    1.100000     1.600000     4.700000     9.100000     0.100000
solution vector:
   -5.313081     5.735673    -2.507607    -1.058741     0.999381
result of matrix applied to sol'n vector
    1.100000     1.600000     4.700000     9.100000     0.100000
**********************************
right-hand side vector:
    4.000000     9.300000     8.400000     0.400000     4.100000
solution vector:
   31.601022   -28.594809    13.389403     2.780322    -3.008798
result of matrix applied to sol'n vector
    4.000000     9.300000     8.400000     0.400000     4.100000
**********************************
```

1.5. Het oplossen van een stelsel eerste orde gewone differentiaalvergelijkingen
PROBLEEM. Bepaal de vectorfunctie $y(x)$, op een gegeven interval, uit het stelsel differentiaalvergelijkingen:

$$\frac{dy}{dx} = f(x,y)$$

$$y(x) = (y_1(x),y_2(x), \cdots ,y_n(x))^T, \quad f(x,y) = (f_1(x,y), \cdots ,f_n(x,y))^T$$

met beginvoorwaarde $y(x_0)=y_0$.

METHODE. Een veel-gebruikte methode is de 4de orde Runge-Kutta-methode die kan worden beschreven in de volgende vorm:

$$k_1 := hf(x_n, y_n)$$
$$k_2 := hf(x_n+h/2, y_n+k_1/2)$$
$$k_3 := hf(x_n+h/2, y_n+k_2/2)$$
$$k_4 := hf(x_n+h, y_n+k_3)$$
$$y_{n+1} := y_n+\tfrac{1}{6}(k_1+2k_2+2k_3+k_4)+O(h^5).$$

Deze formule vraagt 4 evaluaties van de rechterlidfunctie $f(x,y)$ per stap h. Merk op dat als f niet van y afhangt, de RK4-formule reduceert tot een kwadratuurformule, en wel de z.g. regel van Simpson.

Pascal routines

```
PROCEDURE rk 4 (y, dydx: glnarray; n: integer; x, h: real;
  VAR yout: glnarray);
```

Deze procedure berekent m.b.v. de 4de orde Runge-Kutta-methode de oplossing van de differentiaalvergelijking $dy/dx=f(x,y)$ in het punt $x+h$, met als beginvoorwaarde de oplossing in het punt x. De beginwaardevector staat in het n-dimensionale array y en de oplossing wordt afgeleverd in het n-dimensionale array yout. In het programma dat rk4 aanroept, moet een procedure derivs worden gedefinieerd die, gegeven x en de n-vector y, de n-vector dydx in het punt (x,y) aflevert.

```
PROCEDURE rkdumb (vstart: glnarray; nvar: integer; x1, x2:real;
  nstep: integer);
```

Deze procedure berekent, gebruikmakend van rk4, de oplossing van de differentiaalvergelijking $dv/dx=f(x,v)$ in de punten $x1+jh$, $j=1,2, \cdots$, nstep, waarbij $h=(x2-x1)/\text{nstep}$. De beginvector staat gegeven in het n-dimensionale array vstart. Ook rkdumb roept de bij rk4 genoemde procedure derivs aan. De waarden $x1+jh$, $j=1,2, \cdots$, nstep, komen terecht in een globaal array xx[1 . . 200], en de berekende vectoren in een globaal array y[1 . . nvar, 1 . . 200]; het maximale aantal toegestane stappen is hierbij (dus) 200.

Voorbeeldprogramma
Integratie van het stelsel gewone differentiaalvergelijkingen

$$dy_1/dx = -y_2$$
$$dy_2/dx = y_1 - y_2/x$$
$$dy_3/dx = y_2 - 2y_3/x$$
$$dy_4/dx = y_3 - 3y_4/x$$

over het interval [1, 20], in 150 stappen. De beginvoorwaarden zijn gekozen passend bij de oplossing $y_i(x)=J_{i-1}(x)$, $i=1,2,3,4$, de eerste 4 Besselfuncties.

Programma:

```
PROGRAM d1Sr2(input,output);
(* driver for routine RKDUMB *)
CONST
   nvar=4;
   nstep=150;
TYPE
   glnarray = ARRAY [1..nvar] OF real;
VAR
   i,j : integer;
   x1,x2 : real;
   vstart : glnarray;
   xx : ARRAY [1..200] OF real;
   y : ARRAY [1..nvar,1..200] OF real;

(*$I MODFILE.PAS *)
(*$I BESSJ0.PAS *)

(*$I BESSJ1.PAS *)

(*$I BESSJ.PAS *)

PROCEDURE derivs(x : real; y : glnarray;
      VAR dydx : glnarray);
(* The arrays y and dydx must carry the dimension given to
them in the calling routine. *)
BEGIN
   dydx[1] := -y[2];
   dydx[2] := y[1]-(1.0/x)*y[2];
   dydx[3] := y[2]-(2.0/x)*y[3];
   dydx[4] := y[3]-(3.0/x)*y[4]
END;

(*$I RK4.PAS *)

(*$I RKDUMB.PAS *)

BEGIN
   x1 := 1.0;
   vstart[1] := bessj0(x1);
   vstart[2] := bessj1(x1);
   vstart[3] := bessj(2,x1);
   vstart[4] := bessj(3,x1);
   x2 := 20.0;
   rkdumb(vstart,nvar,x1,x2,nstep);
   writeln('x':8,'integrated':17,'bessj3':10,
           'error':10);
   FOR i := 1 to (nstep DIV 10) DO BEGIN
      j := 10*i;
      writeln(xx[j]:10:4,'  ',y[4,j]:12:6,
              bessj(3,xx[j]):12:6,
              y[4,j]-bessj(3,xx[j]):12:6)
   END
END.
```

Output:

x	integrated	bessj3	error
2.1400	0.152015	0.152016	-0.000001
3.4067	0.374321	0.374321	0.000000
4.6733	0.410069	0.410067	0.000002
5.9400	0.132714	0.132711	0.000003
7.2067	-0.211150	-0.211151	0.000000
8.4733	-0.265609	-0.265606	-0.000003
9.7400	-0.005253	-0.005249	-0.000004
11.0067	0.227857	0.227857	0.000000
12.2733	0.152674	0.152669	0.000005
13.5400	-0.107472	-0.107475	0.000003
14.8067	-0.206084	-0.206081	-0.000003
16.0733	-0.029527	-0.029521	-0.000006
17.3400	0.171945	0.171946	-0.000001
18.6067	0.132634	0.132628	0.000006
19.8733	-0.079679	-0.079684	0.000005

LITERATUUR

1. J. PHILLIPS (1986). *The NAG Library: A Beginner's Guide,* Clarendon Press, Oxford.
2. W.H. PRESS, B.P. FLANNERY, S.A. TEUKOLSKY, W.T. VETTERLING (1986). *Numerical Recipes, The Art of Scientific Computing,* Cambridge University Press, Cambridge etc.
3. W.T. VETTERING, S.A. TEUKOLSKY, W.H. PRESS, B.P. FLANNERY (1985). *Numerical Recipes Example Book* (Pascal). Cambridge University Press, Cambridge etc.
4. *Turbo Pascal* (1985), version 3.0, Reference Manual, Borland International Inc.

De Micro-Computer in het Wiskunde-Onderwijs

P.J. van Blokland
Vrije Leergangen Vrije Universiteit, Afdeling Wiskunde,
Postbus 261, 1110 AG Diemen

D. Kok
Vrije Universiteit, Subfaculteit Wiskunde en Informatica,
De Boelelaan 1081, 1081 HV Amsterdam.

0. Inleiding

Computers kunnen een belangrijke rol in het wiskunde-onderwijs spelen en zullen dat in de toekomst ook zeker gaan doen. In Nederland begint, met name onder invloed van het NIVO-project, de verspreiding van de computers over de verschillende scholen een zekere omvang te krijgen. Voeg dit bij het gegeven dat al deze computers werken onder MS-DOS, een voor de ontwikkeling van software stimulerende omstandigheid, en een gematigd optimisme t.a.v. computergebruik in de klas is op zijn plaats.
We benaderen het gebruik van de computer in de klas hier op twee manieren:

a. *De computer als hulpmiddel om knelpunten in het wiskunde-onderwijs aan te pakken.* Gedacht kan dan worden aan:
   - leren werken met variabelen;
   - leren werken met functie-voorstellingen waarin een parameter voorkomt;
   - koppelen van het begrip afgeleide aan de richtingscoëfficient van de raaklijn aan de grafiek van een functie;
   - het maken van een voorstelling van ruimtelijke figuren.

b. *De computer die door zijn nieuwe mogelijkheden tot nu toe ontoegankelijke wiskundige onderwerpen binnen het bereik van de middelbare schoolwiskunde brengt of de kans biedt bekende stof op een veel realistischer wijze te presenteren.* Men kan hierbij denken aan:
   - verwerken van een omvangrijke hoeveelheid statistische gegevens;
   - manipuleren van grote matrices;

- oplossen van differentiaal-vergelijkingen;
- met numerieke methoden oplossen van vergelijkingen en uitrekenen van integralen.

Er verschijnen krachtige programma's waarmee allerlei technische problemen (numeriek maar soms ook al analytisch) kunnen worden opgelost. Daarmee dringt de vraag zich op welke rol technische vaardigheden in ons onderwijs nog moeten spelen. Deze vraag hoeft vandaag nog niet beantwoord te worden, maar haar ontlopen is niet goed meer mogelijk.

In opdracht van het NIVO heeft een groepje mensen (naast de auteurs van deze voordracht behoorden daar ook toe G.P. Bakema, H. Krabbendam en H.B. Verhage) een aantal thema's onderscheiden die bij elkaar een overzicht bieden van zinvol computergebruik in het wiskunde-onderwijs [1].
We kwamen tot de volgende lijst:
Functies en grafieken;
Voortgezet rekenen;
Statistiek en kansberekening;
Meetkunde;
Shortliners;
Spreadsheet;
Simulatie;
Computer-algebra;
Matrix-rekening en besliskunde.

In deze voordracht zullen we elk van deze thema's kort bespreken. Vervolgens zullen we nader ingaan op de thema's 'Functies en grafieken' en 'Simulatie'. Daarbij zal ook door ons bij die thema's ontwikkelde soft-ware gedemonstreerd worden.

1. Thema's voor zinvol computergebruik

1.1. Funkties en grafieken

De functielijn is een van de belangrijkste leerstoflijnen binnen het vak wiskunde in het voortgezet onderwijs. Op deze lijn bevinden zich heel wat punten die een zorgvuldig doordacht leerproces nodig maken, willen leerlingen de aan de orde zijnde begrippen en inzichten zich kunnen eigen maken. In sommige van deze (knel-)punten kan het benutten van computer-programmatuur het leerproces soepeler en meer doeltreffend doen verlopen. Het zijn vooral de grafische mogelijkheden van de computer die in deze programma's gebruikt zijn.
We kunnen twee soorten software onderscheiden:

- Software die leerlingen stimuleert kwalitatief met grafieken om te gaan.
- Software die dienst kan doen als een stuk gereedschap in het analyse-onderwijs.

In de onderbouw van het voortgezet onderwijs is het belangrijk dat de leerling een juist begrip van grafieken opbouwt. Mede met het oog op de toepassingsgerichte wiskunde in de bovenbouw is het dynamische aspect van grafieken essentieel. De volgende, door de SLO ontwikkelde software richt zich met name op het ondersteunen van dit aspect.

- Het programma 'Badkuip' [2] laat de leerlingen zelf grafieken opbouwen.
- Het programma 'Globgraf' laat leerlingen oefenen in het verwoorden van in beeld gebrachte processen.
- Het programma 'Flesvuller' handelt over de relatie tussen de vorm van een fles en de daarbij behorende 'vul'-grafiek.

Het is de bedoeling dat deze software binnenkort ook voor de IBM-computers beschikbaar komt.
In Engeland zijn er voor de BBC-computer soortgelijke programma's gemaakt. We noemen hier alleen 'Traffic'. De autoweg is de context waarbinnen leerlingen een goed begrip van tijd-afstand-grafieken kunnen ontwikkelen.

Het aan de Vrije Universiteit ontwikkelde VU-grafiek is een krachtig functiepakket en daarmee een breed inzetbaar hulpmiddel bij het analyse onderwijs. Diverse knelpunten kunnen ermee aangepakt worden.
We noemen er enkele:

- Herkennen van de essentiële punten in de grafiek.
- Ontwikkelen van het begip 'afgeleide functie'.
- Op intuïtieve wijze onderzoeken of een limiet bestaat.
- Onderzoeken van 'families van functies'.
- Inzoomen op details van de grafiek versus uitzoomen naar een globaal overzicht.
- Met numerieke methoden een snijpunt bepalen.

OP VU-grafiek komen we verderop in deze voordracht nader terug.

1.2. Voortgezet rekenen en algebra
In het onderbouwprogramma van het voortgezet onderwijs krijgen rekenactiviteiten noodgedwongen een steeds groter accent. De computer heeft enkele specifieke eigenschappen die bij dit voortgezet rekenonderwijs van dienst kunnen zijn. We noemen:

- De mogelijkheid tot visualisering van rekenoperaties.
- De mogelijkheid om de leerling onmiddellijk feed-back te geven.
- De grafische mogelijkheden: het snel tekenen van roosters en 100-velden.
- Animatie. Leerlingen vinden het maken van rijtjes opgaven vaak vervelend. Computer-spelletjes bieden een motiverende omgeving, waarbinnen leerlingen misschien tot extra oefening verleid kunnen worden.

Ook de rekenmachine is in dit verband belangrijk. Vroeger moest de leerling beschikken over de combinatie 'rekenen met tafels - pen en papier - kennis van

de rekenalgoritmen' om de gemiddelde rekenopgave te kunnen maken. Tegenwoordig volstaat de combinatie 'rekenen met tafels - rekenmachine'. Het is nuttig te beseffen dat hier slechts sprake is van een verandering in technologisch opzicht, die overigens wel zijn consequenties heeft voor de dagelijkse onderwijspraktijk.

Het NIVO-OMO-SCO-project heeft o.a. het programma 'Schatten' opgeleverd, dat er veelbelovend uitziet.
Een ander programma dat belangrijk lijkt, is van Joost Klep, werkzaam bij de SLO. Het draagt de naam 'Een wereld rond getallen'. Hier kan de leerling zelf kiezen volgens welk model hij de tafelsommen wil benaderen: getallenlijn, stroken, rechthoekmodel of groepjes. Met name dit aspect lijkt ons heel belangrijk, nl. dat de leerling zijn benadering hier niet hoeft in te leveren bij een alleen maar sturende computer.

Er zijn nog erg weinig programma's die hulp kunnen bieden bij het algebraonderwijs. We bedoelen nu niet 'drill and practice'-programma's, die zijn er wel, maar juist programma's die gericht zijn op begripsvorming. Thomas en Tall hebben veelbelovende resultaten geboekt met een programma dat leerlingen hielp een goed variabele-begrip te ontwikkelen. We kennen van dit experiment alleen maar schriftelijke verslagen. Er is een programma, 'SOLVE' geheten, draaiend op de BBC, dat leerlingen zich bewust maakt hoe het oplossen van vergelijkingen in zijn werk gaat. Ons zijn van dit programma geen klasse-ervaringen bekend. Hetzelfde geldt voor 'ZOOM', een programma waarmee de leerling kan verkennen hoe 'vol' de getallenlijn wel is. Lerarenintuïtie brengt ons ertoe dit programma de moeite van het uitproberen waard te achten.

Overigens geeft bovenstaande alinea wel aan hoe onzeker de toestand rond het computergebruik in de wiskundeles nog altijd is.

1.3. Statistiek en kansrekening
In het onderbouwprogramma hebben kansrekening en statistiek altijd een bescheiden plaats ingenomen, ondanks het feit dat deze onderwerpen voor een algemene vorming erg belangrijk zijn. Veel verder dan het bepalen van modus en mediaan van eenvoudige getallen reeksen kwam men niet. Spreidingsmaten kwamen al helemaal niet aan bod. Bovendien bleek uit incidentele experimenten, o.a. die van het I.O.W.O., dat voor veel leerlingen het kansbegrip heel moeilijk was.
Deze situatie is de laatste tijd wat aan het veranderen. In het nieuwe wiskunde-A programma vormen statistiek en kansrekening een belanrijk onderdeel. Bovendien is geprobeerd de statistiek wat meer uit te werken in de richting van het kritisch kijken naar en interpreteren van langs statistische weg verkregen gegevens. Voor het nieuwe HAVO programma lijkt een soortgelijke ontwikkeling waarschijnlijk.
De computer kan bij deze ontwikkelingen een belangrijke ondersteunende

rol vervullen. Het altijd zo omvangrijke rekenwerk kan worden geautomatiseerd. De bijvoorbeeld uit enquêtes verkregen gegevens kan men door de cmputer laten verwerken. De resultaten kunnen gepresenteerd worden via:

- frequentietabellen en histogrammen;
- puntenwolken en regressielijnen;
- kruistabellen.

In het onderwijs kan het accent meer komen te liggen op het formuleren van vermoedens en het interpreteren en controleren van door computerverwerking verkregen resultaten.

VU-STAT [3] is een statistisch pakket waarmee enquêtes kunnen worden ingevoerd en verwerkt. In de praktijk bleken leerlingen van Havo-3 in staat zelf kleine onderzoekjes op te zetten en die met het pakket te verwerken.

Men kan de computer ook gebruiken om aan gegevens voor statistisch onderzoek te komen. In het programma 'TIMES' (draaiend op de BBC, onderdeel van 'Teaching with a micro', math 3) bepaalt de micro de reactie-snelheid van de gebruiker op bijvoorbeeld het in beeld verschijnen van een bepaalde letter. Die tijden kunnen vervolgens statistisch onderzocht worden

Een veel gebruikte mogelijkheid van de computer is het via de zgn. 'randomgenerator' simuleren van kans-spelen. De wet van de grote getallen kan zo gecontroleerd worden.

1.4. Turtle geometry en ruimtemeetkunde

De rol die meetkunde kan spelen in het wiskunde-onderwijs krijgt de laatste tijd weer alle aandacht. Nu de grafische mogelijkheden van de micro-computer zo verbeterd zijn, moet ook nagegaan worden in hoeverre dit apparaat een bruikbaar leermiddel is bij een vernieuwd meetkunde-onderwijs.
Uit het boek 'Turtle Geometry' van Abelson en Di Sessa [4] blijkt dat, anders dan vaak gesuggereerd wordt, Turtle geometry ook aanknopingspunten biedt voor het huidige analyse-onderwijs. Via Turtle-graphics krijgen leraar en leerling de beschikking over een instrument om meetkundige figuren of functies grafisch te representeren. Er zijn aanwijzingen dat leerlingenpractica met Turtle-graphics een meer onderzoekende houding bij leerlingen kunnen stimuleren. Recente ontwikkelingen in LOGO maken het mogelijk dat transformaties, vlakvullingen, projectie methoden en vectormeetkunde onderwerpen zijn, waaruit die practica zouden kunnen bestaan.

Ruimtemeetkunde is weer terug in het bovenbouwprogramma van het voortgezet onderwijs. Nu nog alleen op het VWO, maar binnen afzienbare tijd ook op de HAVO.
De computer kan een krachtige ondersteuning bieden bij een belangrijk onderdeel van het leerproces in de ruimtemeetkunde: het maken van een voorstelling van een ruimtelijke figuur.

Met het programma 'Ruimfig' [5] van het OW&OC hebben Heleen Verhage en Martin Kindt op bescheiden wijze geëxperimenteerd. Daarbij speelden de ideeën van Lauwerier [6] op de achtergrond een belangrijke rol.
Er zou eigenlijk een CAD-achtige omgeving (CAD = Computer Aided Design) gemaakt moeten worden, waarbinnen heel eenvoudig ruimtelijke figuren getekend, gedraaid, doorsneden etc. kunnen worden. De grafische mogelijkheden van de huidige micro-computer zijn goed genoeg voor zo'n programma. Zo'n hulpmiddel kan het onderwijs in de ruimtemeetkunde best gebruiken.

1.5. Short-liners
Met short-liners bedoelt men programma's die niet meer dan 20 regels tellen. Een listing past dus op een scherm. Deze korte programma's kunnen toch heel krachtig zijn. In zowel de USA als in Engeland wordt het belang van short-liners voor het wiskunde-onderwijs onderkend. Zo publiceerde de 'Mathematical Association' in 1986 een boekje met 132 Short Programs, geschreven in BASIC. Bij dit boek hoort een schijfje voor de BBC. Ook LOGO biedt trouwens een goede programmeeromgeving voor shortliners.
De waarde van short-liners is tweeledig. Het schrijven van zo'n programma vereist om te beginnen een zorgvuldige analyse van het onderwerp, men moet een algoritme begrijpen om het te kunnen benutten in een programma. Verder levert het draaien van zo'n programma een beter inzicht op in wat er precies gebeurt.
Hierbij wordt er vanuit gegaan dat het programma door de leraar in samenwerking met de klas wordt geschreven.
Ook in Nederland vindt men al een begin van het gebruik van shortliners: namelijk in de OW&OC-pakketten voor Wiskunde-A. Zo wordt met een shortliner bijvoorbeeld onderzocht wat de afgeleide is van $f(x)=\sin(2x)$. Meer informatie vind men in 'Hewet en de micro' [7].
In dit verband moet ook de naam van Lauwerier genoemd worden die een indrukwekkende hoeveelheid korte, veelal grafische programma's heeft geschreven. In de Nieuwe Wiskrant doet hij van dat werk regelmatig verslag.
Het lijkt ons de moeite waard om binnen de Nederlandse situatie de mogelijke rol van korte programma's na te gaan.

1.6. Spreadsheets
Bij het onderep 'Spreadsheets' gaan we nu eens niet uit van het wiskunde-curriculum, maar van een stuk gereedschap dat zijn nut op vele plaatsen (voornamelijk buiten het onderwijs) al bewezen heeft.
Het spreadsheet is een typisch rekenprogramma, zeg maar de opvolger van de zakrekenmachine.
Je kunt stellen dat het tot het normale repertoire van een wiskundeleraar zou moeten behoren om te weten wat een spreadsheet is en hoe zo'n programma werkt. Bovendien is het 'programmeren' in een spreadsheet een aardig alternatief voor veel korte rekenprogrammaatjes.
Het spreadsheet kan op vele plaatsen in de wiskundelessen worden gebruikt.

Enkele voorbeelden:

- het maken van een tabel met functiewaarden;
- het opbouwen van een formule met tussenvariabelen (in hulpcellen of hulpkolommen);
- eenvoudige statistische bewerkingen als gemiddelde en standaard deviatie;
- rijen, reeksen, limietprocessen;
- numeriek differentiëren en integreren;
- matrixbewerkingen.

Bij wiskunde-A wordt al veel gebruik gemaakt van numerieke methoden en een spreadsheet kan daarbij een goed hulpmiddel zijn.

Binnen het wiscom-project van de vakgroep OW&OC wordt thans gewerkt aan het ontwikkelen van lesvoorbeelden voor wiskunde bij PCCALC, de spreadsheet van het NIVO-startpakket [8]. Deze voorbeelden zijn gespreid over lbo, mavo, havo en vwo. Rond september 1987 zullen ze, samengebracht in een boek met een bijhorend schijfje, verschijnen.

1.7. Simulaties

Modelvorming neemt een steeds belangrijker plaats in het wiskunde onderwijs in. Bekende voorbeelden uit het nieuwe wiskunde-A programma zijn: modellen voor exponentiële en geremde groeiprocessen, prooi-roofdiersystemen.
De computer is een voor de hand liggend hulpmiddel om modellen snel door te kunnen rekenen. Er zijn speciale simulatietalen ontwikkeld waarin modellen eenvoudig geformuleerd kunnen worden en die goede voorzieningen hebben om de resultaten van de berekeningen overzichtelijk (o.a. grafisch) weer te geven. VU-dynamo is bijvoorbeeld zo'n taal.

Op diverse plaatsen in het onderwijs zou meer aandacht aan het werken met modellen besteed kunnen worden.
In de middenbouw kan dit door bij een gegeven model met de constantes te manipuleren en de verschillende grafieken die dit oplevert te bestuderen.
In de bovenbouw is het mogelijk dat leerlingen leren zelf een model op te stellen bij een probleem en dat met behulp van een simulatietaal door de computer laten verwerken.
Het interpreteren van de uitkomsten is daarbij eveneens een belangrijke aktiviteit voor de leerlingen.

Het onderwerp differentiaalvergelijkingen zou bij deze aanpak in een heel ander licht komen te staan. De nadruk zou niet langer liggen op oplossingstechnieken, maar veel meer op het formuleren en interpreteren van modellen.
Zoals gemeld komen we op VU-dynamo in deze voordracht nog terug.

1.8. Computer algebra
Sinds enkele jaren is ook voor micro-computers een computeralgebra-systeem beschikbaar, muMATH geheten. Dit systeem bestaat uit een pakket functies voor formule-evaluatie en formule-manipulatie.
Tot de mogelijkheden van muMATH behoren:

- rekenen met rationale getallen ($\frac{2}{3} + \frac{2}{5}$) en symbolisch geschreven irrationale getallen (e, π, wortels, etc.);
- bewerkingen met vectoren en met matrices;
- rekenen met variabelen, bijvoorbeeld substitutie van waarden voor variabelen in expressies;
- rekenen met expressies, bijvoorbeeld in de zin van haakjes wegwerken of ontbinden in factoren;
- analytisch oplossen van vergelijkingen, bijvoorbeeld van vierkants-vergelijkingen;
- analytisch differentiëren en primitiveren;
- analytisch integreren;
- het op analytische wijze berekenen van limieten, bijvoorbeeld met de stelling van de l'Hospital.

In het artikel 'Experiments with muMATH in Austrian High Schools' stond:

> "Since symbolic manipulation is an essential component of mathematical education in today's high schools the question arises, how far CA-systems can be integrated into teaching mathematical subjects."

De schrijvers verwachten een invloed op het leerplan

> "that will be equally drastic to what happened when the pocket calculator replaced logarithm tables."

De toekomst waarin onze leerlingen een rekendoosje bezitten dat exact kan rekenen en analytisch met formules kan omgaan, is hooguit 10 jaar van ons verwijderd.

Kennismaking met een pakket als muMATH [9] lijkt dus voor iedere wiskundeleraar een must. De directe toepassingen bij alles wat met rekenen, algebra en analyse te maken heeft, liggen voor het oprapen [10].
In de literatuur wordt de verwachting uitgesproken dat over niet al te lange tijd computeralgebra-systemen met grafische mogelijkheden op de markt zullen verschijnen.

Aldus worden we geconfronteerd met een situatie waar geen leraar om gevraagd heeft. Er wordt ons al een antwoord gegeven terwijl we niet eens de tijd kregen om de goede vragen te stellen.
Alhoewel, we hebben nog een paar jaar de tijd voor onze leerlingen met zo'n apparaatje in hun tas het lokaal binnenstappen.
Welke vaardigheden zullen dan belangrijk zijn?
'Symbolisch schatten' in elk geval.

1.9. Matrixrekening en besliskunde
Met de invoering van wiskunde-A is de matrixrekening vanuit een geheel nieuwe optiek in het wiskunde-onderwijs terecht gekomen. Een matrix wordt opgevat als een model om situaties wiskundig mee te beschrijven. Het accent ligt op het opstellen van modellen en de interpretatie van de uitkomsten. Het valt te verwachten dat in de nabije toekomst ook in de onderbouw en op de HAVO matrices in een praktische toepassingscontext behandeld zullen worden. Elementen van besliskunde zoals lineair programmeren, komen zowel bij wiskunde-A als straks op de Havo aan de orde.
In deze vakken is het gebruik van de computer een geïntegreerd onderdeel van het curriculum (automatische Gegevens Verwerking). In de praktijk is gebleken dat in het bijzonder matrixrekening en lineair programmeren tot computergebruik uitnodigen.
De computer wordt dan als gereedschap ingezet voor de verwerking van grote matrices of voor het uitvoeren van algoritmen voor lineaire programmering. Zie bijvoorbeeld het artikel van H.C. Tijms in Euclides [11].

De vraag kwam al eerder naar voren: welke invloed zal de computer op de inhoud van het leerplan hebben. In het buitenland worden genoemd: meer aandacht voor numerieke wiskunde, maar zeker meer aandacht voor discrete wiskunde. Misschien zal in de komende jaren de besliskunde wel een belangrijker plaats gaan innemen, dan de bescheiden positie die ze nu heeft verworven.

Tot zover de globale behandeling van de thema's zoals die werden onderscheiden in het Raamplan NIVO-nascholing wiskunde.

We worden nu concreter.
De afgelopen jaren is er aan de faculteit Wiskunde en Informatica van de Vrije Universiteit door de vakgroep Didactiek software ontwikkeld die bruikbaar moest zijn binnen het wiskunde-onderwijs op de middelbare school. We willen U enige informatie verschaffen over de pakketten VU-dynamo en VU-grafiek.

2. Computer simulaties en VU-dynamo

Modelvormning neemt een steeds belangrijker plaats in het wiskunde-onderwijs in, zowel op het niveau van het voortgezet onderwijs als in het HBO en Universitair onderwijs. Steeds meer wordt ingezien dat wiskundige modelvorming niet alleen binnen vakken als natuurkunde, scheikunde, biologie, economie etc. aandacht behoort te krijgen, maar dat het net als andere basisvaardigheden zijn natuurlijke plaats in het wiskunde-curriculum verdient.

De computer kan bij het doorrekenen van het model een grote rol spelen. VU-dynamo is een computersimulatietaal speciaal voor die simulaties die continu in de tijd verlopen. Natuurlijk kunnen dezelfde simulaties ook in BASIC of PASCAL geschreven worden, maar met een simulatietaal gaat het veel eenvoudiger. Vooral het maken van grafieken gaat gemakkelijker. Naast dit type

simulatietalen zijn er ook simulatietalen die gericht zijn op stochastische problemen, zoals wachttijd problemen.

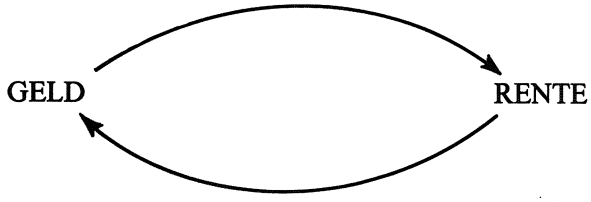
In DYNAMO is het wereld-model van de club van Rome beschreven. Dit model staat in het boek WORLD DYNAMICS van Forrester [12]. Dit is de eerste poging geweest om een samenhangend model te maken van de wereld economie, bevolkingsgroei, milieu en voedsel, vanuit een dynamisch feedback perspectief. Het gaat juist om de verbanden tussen de verschillende delen die tesamen het geheel vormen.
Het boek 'Introduction to Computer Simulation' van Nancy Roberts e.a. [13] laat zien hoe je dit op middelbare school niveau kan behandelen.

Andere modellen die zijn gemaakt in deze taal, zijn:

- een model over het verband tussen heroïne, misdaad en politie-activiteiten (Hoofdstuk 24 van [13]);
- een ecologisch model over herten, leeuwen en voedsel (KAIBAB, zie Hoofdstuk 18 van [13]);
- een economisch model van de prijsvorming van producten als jute, koffie, tin, om de lange termijn fluctuaties in de prijsvorming te verklaren en om de effecten van eenvoudige stabilisatiepolitiek te bestuderen. (Zie Simuleren volgens de SYSTEM DYNAMICS methode van A.W. Abcouwer in [14]).

2.1. Geld en rente

Eerst een voorbeeld om de taal dynamo te introduceren: het verband tussen geld en rente.



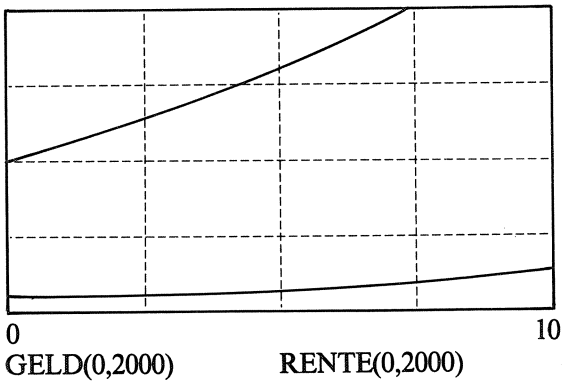
FIGUUR 1.

Hoe meer geld, des te meer rente. Maar ook: hoe meer rente, des te meer geld. Een diagram als hierboven heet een oorzaak-gevolg diagram. Als de twee pijlen elkaar versterken spreken we van een positieve loop. Als er een positieve en een negatieve pijl is, dan noemen we het een negatieve loop.

Het dynamo-programma is als volgt:

```
L GELD = GELD + DT * RENTE
R RENTE = GELD * RENTEPERCENTAGE / 100
C RENTEPERCENTAGE = 10
N GELD = 1000
SPEC DT = 1 / EINDTIJD = 10
PLOT GELD, RENTE (0,2000)
```

L staat voor LEVEL, R voor RATE, N voor INITIAL, C voor CONSTANTE, SPEC voor SPECIFICATIE. Eerst berekent dynamo de beginwaarden. Daarna worden de levels berekend. Op basis van de levels worden de overige variabelen bepaald. Daarna wordt weer een periode DT opgeschoven. Steeds worden op grond van de oude waarden de nieuwe waarden berekend. Draaien van dynamo levert het volgende plaatje.



FIGUUR 2.

`L GELD = GELD + DT * RENTE`
In woorden staat hier dat de nieuwe hoeveelheid geld gelijk is aan de oude hoeveelheid geld plus de rente.

`R RENTE = GELD * RENTEPERCENTAGE / 100`
De rente staat hier eigenlijk voor de rente per jaar. De rente (per jaar) is gelijk aan de hoeveelheid geld keer het rentepercentage.

`N GELD = 1000`
De N is van iNitial. Dynamo gebruikt `GELD = 1000` als startwaarde voor de berekening.

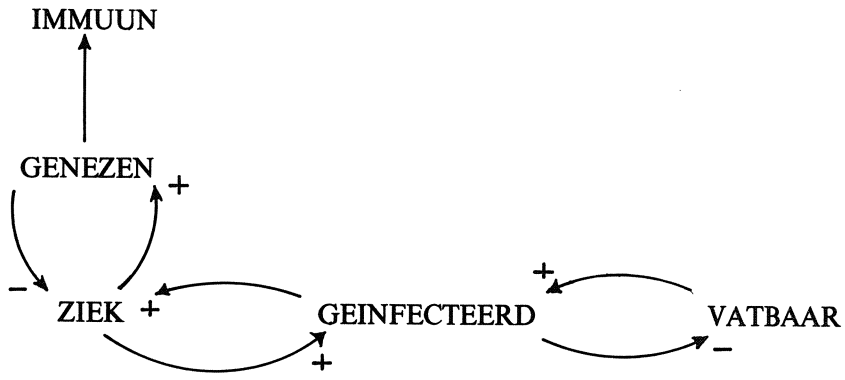
`SPEC DT = 1 / EINDTIJD = 10`
In deze regel staat hoe dynamo het moet berekenen. De reken-periode is 1. De berekening heeft betrekking op een periode van 10 jaar. Indien de rente per half jaar werd uitgekeerd dan moet je `DT = 0.5` nemen.

2.2. Een griepmodel

Als tweede voorbeeld kiezen we een eenvoudig griepmodel.
De maatschappelijke kosten van een griep-epidemie in de VS worden geschat op minstens 1.000.000.000 gulden. Het kan dus best een zinvolle investering zijn om bepaalde maatregelen te nemen tegen de griepepidemie, zoals inenten. Wij maken een heel eenvoudig model. Daartoe maken wij een paar veronderstellingen.

- Beginsituatie is een populatie van 1000 mensen met daarin 1 zieke.
- Als een persoon eenmaal ziek is, kan hij andere mensen aansteken.
- Een persoon kan de ziekte maar één keer krijgen.
- Iemand die ziek is, is gemiddeld 5 dagen ziek.

De populatie bestaat dus uit drie deelgroepen: de vatbaren, de zieken en de mensen die immuun zijn. Het oorzaak-gevolg diagram staat hieronder.

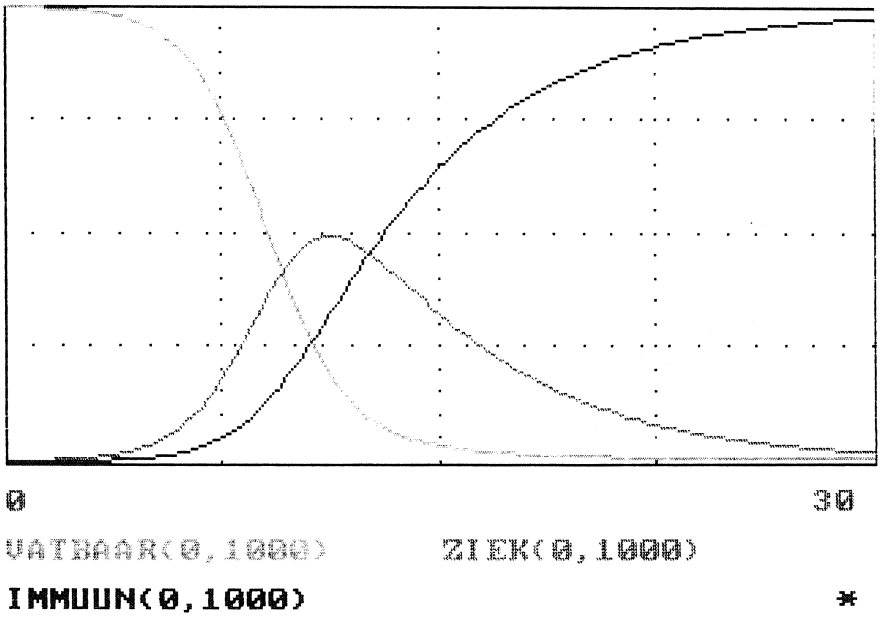


FIGUUR 3.

Het model bevat drie levels (immuun, ziek en vatbaar) en twee rates (genezen en geïnfecteerd worden).
PRCON is de kans dat een contact tussen een zieke en een gezonde in ziekte resulteert.
CONTACT is het aantal contacten tussen zieken en gezonden. GEMCON is het gemiddeld aantal personen dat 1 persoon per dag ontmoet.
De gemiddelde duur van een ziekte is 5 dagen (GENASR = 5).

Het VU-dynamo programma staat hieronder.

```
l vatbaar = vatbaar + dt * geïnfecteerd
n vatbaar = vatbaarn
c vatbaarn = 999
r geïnfecteerd = prcon * contact
c prcon = 0.2      ; kans dat een contact tot een
  infectie leidt
a contact =  gemcon * (vatbaar / totaal) * ziek
c gemcon = 5
NOTE GEMCON := Gemiddeld aantal mensen dat
  1 persoon per dag ontmoet
n totaal = vatbaar + ziek + immuun
l ziek = ziek + dt (geïnfecteerd - genezen)
n ziek = ziekn
c ziekn = 1
r genezen = ziek / genasr
c genasr = 5       ; gemiddelde duur ziekte
l immuun = immuun + dt * genezen
n immuun = immuunn
c immuunn = 0
spec dt = 0.25/ eindtijd = 30/ prtper = 1/ bewaren
plot vatbaar, ziek, immuun (0,1000)
plot contact / geïnfecteerd
print vatbaar / ziek / immuun
```

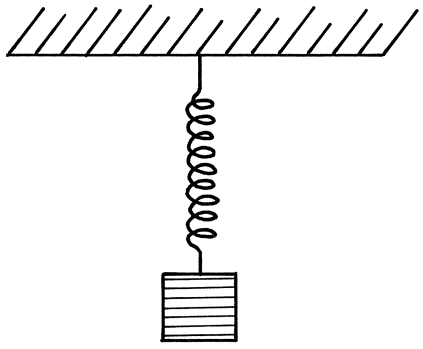


FIGUUR 4.

Toelichting op het programma.
Een nieuw taalelement in dit programma is de A van Auxiliary hetgeen hulpvergelijking betekent. Door middel van A-vergelijkingen kunnen we hulpvergelijkingen introduceren die het model inzichtelijker maken.

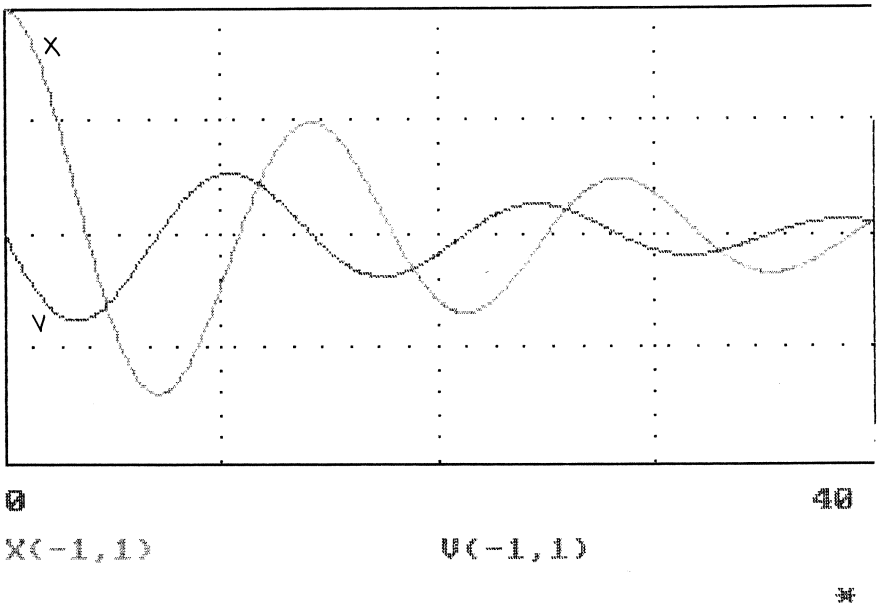
2.3. De veer
Een belangrijk toepassingsgebied is ook de natuurkunde. Een eenvoudig voorbeeld daarvan is de veer waaraan een massa hangt. Deze veer wordt uitgetrokken en daarna losgelaten.

```
l x = x + dt * v
l v = v + dt * a
a a = f / m
c m = 5
a f = -x
n x = 1
n v = 0
spec dt = 0.1 / eindtijd = 40 / cont
plot x,v
```



FIGUUR 5.

Bij de SPEC is de optie CONT toegevoegd. Deze optie zorgt ervoor dat een nauwkeuriger numerieke methode wordt toegepast (vierde orde Runge-Kutta).
De gedempte veer krijg je door de vergelijking $A\ F=-X$ te vervangen door $A\ F=X-R*V$ en de vergelijking $C\ R=-0.5$ toe te voegen.

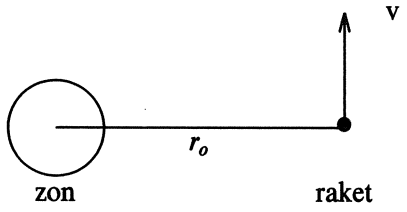


FIGUUR 6.

2.4. De planeetbewegingen

Een dynamo programma om de baan van een raket om de zon te simuleren. De zon staat in de oorsprong en daar omheen draait een raket met een massa van 100 kg. De beginsnelheid en de beginafstand kunnen gevarieerd worden.

```
l x = x + vx * dt
l y = y + vy * dt
a r = sqrt (x ^ 2 + y ^ 2)
a f = -g * m1 * m2 / (r * r)
a fx = f * x / r
a fy = f * y / r
l vx = vx + fx * dt / m2
l vy = vy + fy * dt / m2
c m1 = 6.6 e 24
c m2 = 100
c g = 6.7 e -11
c v0 = 3000
C r0 = 2 E 7
n x = r0
n y = 0
n vx = 0
n vy = v0
spec dt = 300 / eindtijd = 30000 / cont
xyplot x,y
```



FIGUUR 7.

Via de *xyplot* kunnen we de baan van de raket zien. Op eenvoudige wijze kan dit programma worden uitgebreid tot het twee of drie lichamen probleem.

2.5. Systeemdynamica in het voortgezet onderwijs

Op dit moment is er vrijwel geen ruimte binnen het examenprogramma om ook hier aandacht aan te besteden. Qua moeilijkheid is het op het niveau van bovengenoemde contexten wel degelijk haalbaar in de bovenbouw van het VWO (en HAVO?). Ook al omdat deze simulatietaal een zinvolle rol in andere vakken (natuurkunde, scheikunde, economie, biologie) kan spelen zou het aan te bevelen zijn om in de vierde klas van het VWO ruimte te scheppen voor experimenten met Systeemdynamica. Het geheel van een simulatietaal, spreadsheet en een statistisch pakket, zou ons inziens een goede invulling zijn van de informatica in het bovenbouw wiskunde-onderwijs.

Een andere mogelijheid om systeemdynamica in te voeren in de bovenbouw van het VWO zou zijn om het te koppelen aan het onderwerp

differentiaalvergelijkingen.

Als systeemdynamica in het onderwijs komt, dan is het zeker van belang om behalve aan de wiskundige kant veel aandacht te besteden aan modelvorming. Over modelvorming valt meer te zeggen dan dat het maar een model is en dat de werkelijkheid toch veel ingewikkelder is.

Het is te verwachten dat de leerlingen later de wiskunde vaak in een computerachtige context moeten toepassen. Het zou een goede zaak zijn om ze daarop voor te bereiden.

3. VU-GRAFIEK

VU-grafiek is een voorbeeld van wat we zouden willen aanduiden met de term 'instrumentele software'. Met de term instrumenteel willen we aanduiden dat VU-grafiek kan worden gebruikt als een stuk gereedschap bij de analyse. De bruikbaarheid is dus niet beperkt tot een bepaald onderdeel van (het onderwijs in) de analyse. VU-grafieken is in dit opzicht te vergelijken met pen en papier of met een rekenmachine: nuttig om voortdurend onder handbereik te hebben.

Bij de opzet van het programma hebben we ons sterk laten inspireren door het pakket 'Graphic Calculus' van David Tall [15], een programma dat alleen op de BBC draaide.

Op dit moment bestaat VU-grafiek uit de volgende onderdelen:

1. Zoek het funktie-voorschrift
2. Grafieken teken
3. Uitvergroten
4. Differentiëren
5. Integreren
6. Oplossen van vergelijkingen (nulpunten en $f(x) = 0$)
7. Taylor polynomen
8. Differentiaal-vergelijkingen
9. Functies van meer veranderlijken

P. Parametervoorstellingen van krommen

T. Niveau-lijnen

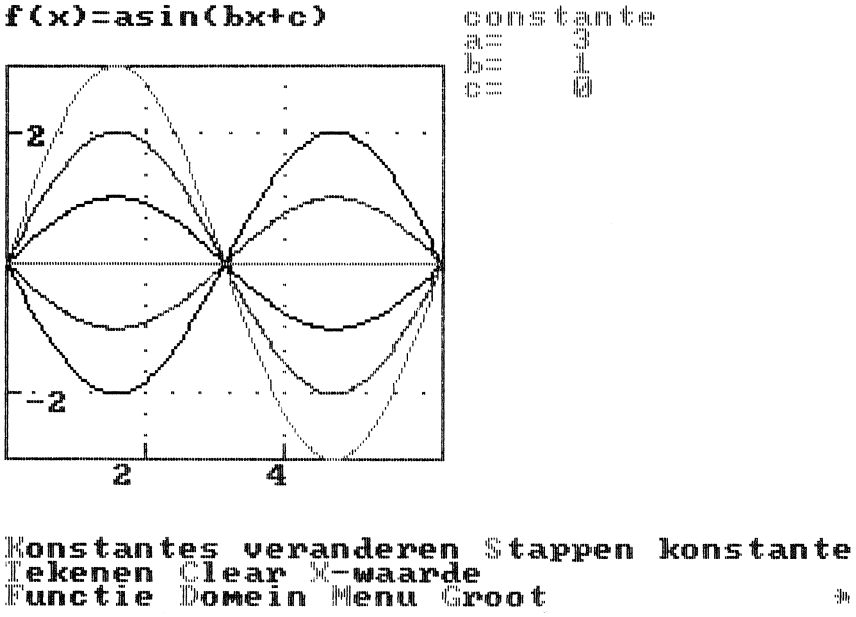
In deze voordracht willen we aan de hand van het genoemde pakket een aantal aspecten van computergebruik in de wiskundeles illustreren.

1. De computer als electronisch schoolbord of dito tekentafel.
2. De computer die een andere didactische aanpak mogelijk maakt.
3. De computer die ons stimuleert vragen te stellen over de inhoud van het leerplan.

3.1. De computer als electronisch schoolbord of dito tekentafel

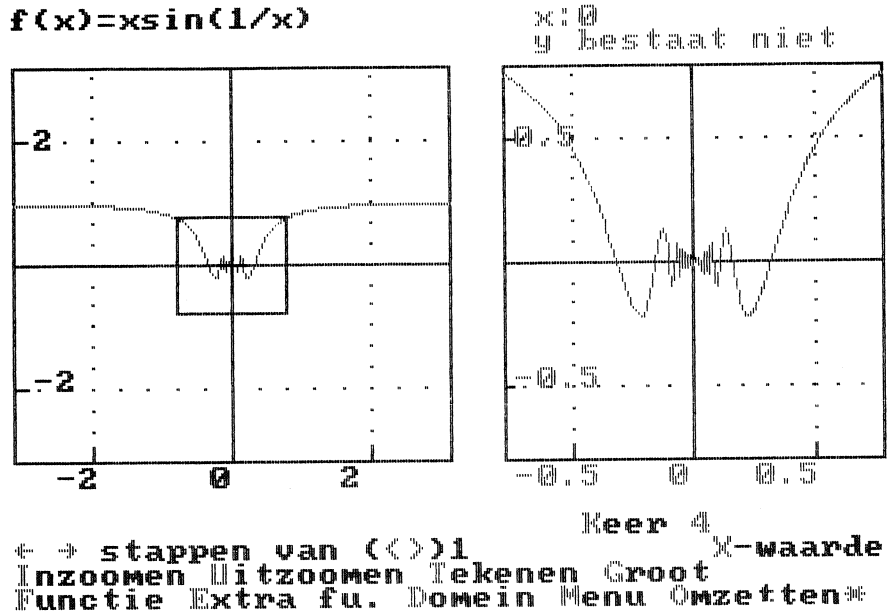
Veel leerlingen hebben moeite met funktie-voorschriften waarin parameters voorkomen. Ze kunnen zich niet precies voorstellen wat het effect van die parameters is. Een aanpak die leraren vaak aanbevelen luidt: teken een plaatje bij enkele waarden van die parameter. Helaas is dat schetsen een tijdrovend werkje dat bovendien de aandacht kan afleiden van het eigenlijke probleem.

Met het onderdeel 'Grafiek tekenen' van VU-grafiek komen we aan dit bezwaar tegemoet. Zo gauw er parameters in een funktievoorschrift voorkomen, wordt de vraag gesteld: welke waarde heeft de constante. Ook kan men met een zekere stapgrootte de constante laten variëren en vervolgens beslissen hoeveel funkties men in beeld wil hebben. Dit leidt bij het funktie voorschrift $f(x) = a \sin(bx + c)$ tot het volgende plaatje.



FIGUUR 8.

Er zijn funkties waarvan de grafiek in de buurt van een bepaald punt een bizar gedrag vertoont waar men graag wat meer van zou willen weten. Met de optie 'Uitvergroten' van VU-grafiek kan men de situatie 'en detail' bekijken. Als voorbeeld kiezen we funktie $f(x) = x \sin(1 / x)$



FIGUUR 9.

Twee vragen dringen zich op:

- Hoe gedraagt de funktie zich in de buurt van $x = 0$?
  Deze vraag wordt door het rechter plaatje al aardig beantwoord.
  Desgewenst kan men nog verder inzoomen.
  En een tweede vraag:
- Hoe gedraagt de funktie zich als $|x|$ heel groot wordt?
  Deze vraag kunnen we beantwoorden door juist flink uit te zoomen. Bovendien komt nu de optie '*X*-waarde' erg van pas. Zo vinden we bij $x = 1000$ de *Y*- waarde 0.99999998.

Met deze voorbeelden is, hopen we, duidelijk geworden hoe de leerling met dit programma gestimuleerd kan worden zelf op onderzoek uit te gaan. Daarbij wordt hij niet steeds gehinderd door reken- en/of teken-technische details.

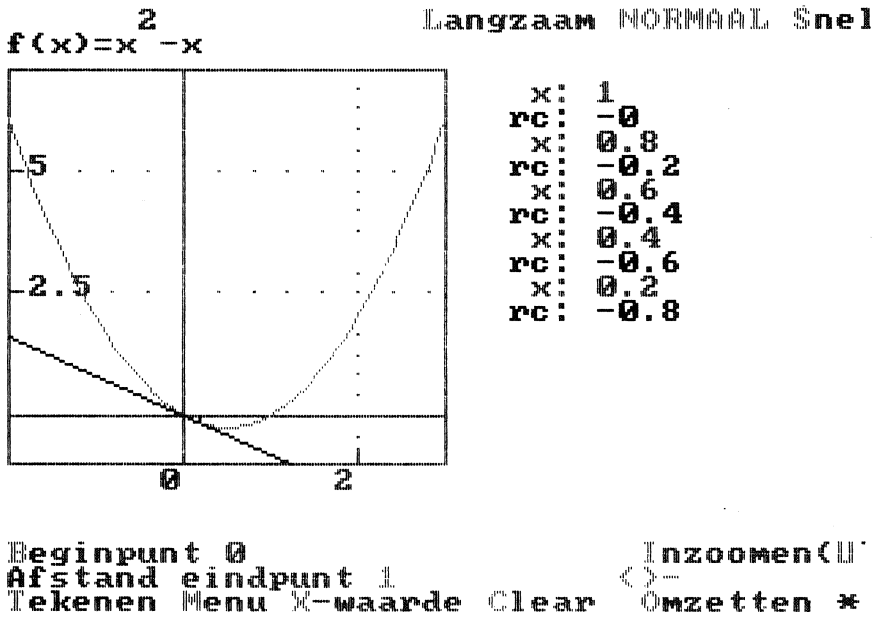
Een probleem dat juist door deze extra-mogelijkheden kan ontstaan is echter: hoeveel behoefte heeft de leerling nog aan een formeel bewijs. Het plaatje overtuigt toch volledig? Soortgelijke problemen deden zich ook voor bij de introductie van de rekenmachine. Zeker is in elk geval dat veel leerlingen niet vatbaar zijn voor formele bewijzen, en deze leerlingen kunnen nu wel 'overtuigd' worden.

Aan de andere kant: juist de computer is in staat de leerling te confronteren met grafieken die om nadere begripsbepaling vragen. We komen hierop terug

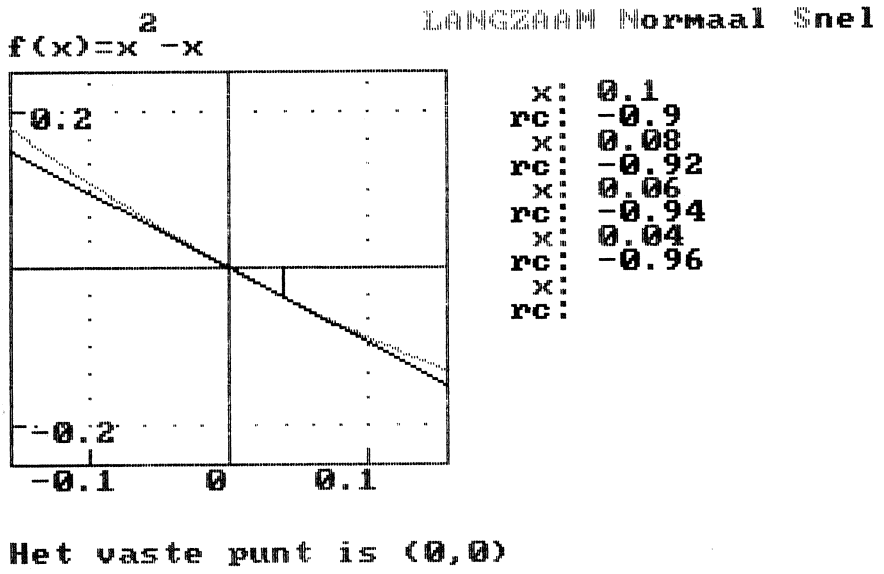
bij het onderdeel 'differentiëren'.

3.2. De computer die een andere didactiek mogelijk maakt
We kiezen als voorbeeld de manier waarop het begrip afgeleide geïntroduceerd kan worden, in de vierde klas VWO of HAVO.
Twee zaken zijn daarbij van belang:
- is de funktie differentiëerbaar voor een bepaalde waarde van x?
- is de funktie differentiëerbaar op zijn domein dan bestaat er een afgeleide funktie, waarvan we het voorschrift willen bepalen.

Differentiëerbaar in een punt komt er op neer dat de funktie in de buurt van dat punt benaderd kan worden door een lineaire funktie. Wat deze voorwaarde inhoudt kan duidelijk worden als we de inzoom-mogelijkheid van het programma benutten.
We nemen $f(x) = x^2$



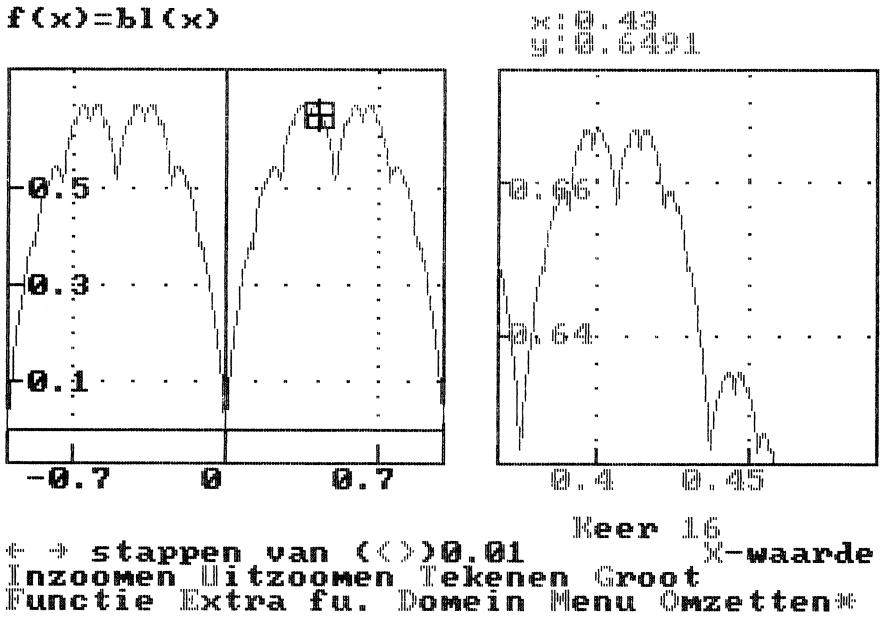
FIGUUR 10.



FIGUUR 11.

Uit het laatste plaatje blijkt dat bij sterke uitvergroting de grafiek van $f(x)$ er inderdaad rechtlijnig uitziet.

In de wiskunde, maar ook in het alledaagse leven zijn leerlingen vertrouwd geraakt met de gedachte dat rechtlijnigheid iets heel gewoons is. Maar zeker als men voorwerpen sterk vergroot blijft er van die rechtlijnigheid weinig over: bekijk maar eens een lineaal onder een microscoop, stelt David Tall [16], [17] voor. Het fenomeen 'vergroten om er rechtlijnig uit te zien' is dan ook een speciale wiskundige eigenschap. Hoe speciaal wordt duidelijk als we de Blancmange-funktie bekijken:



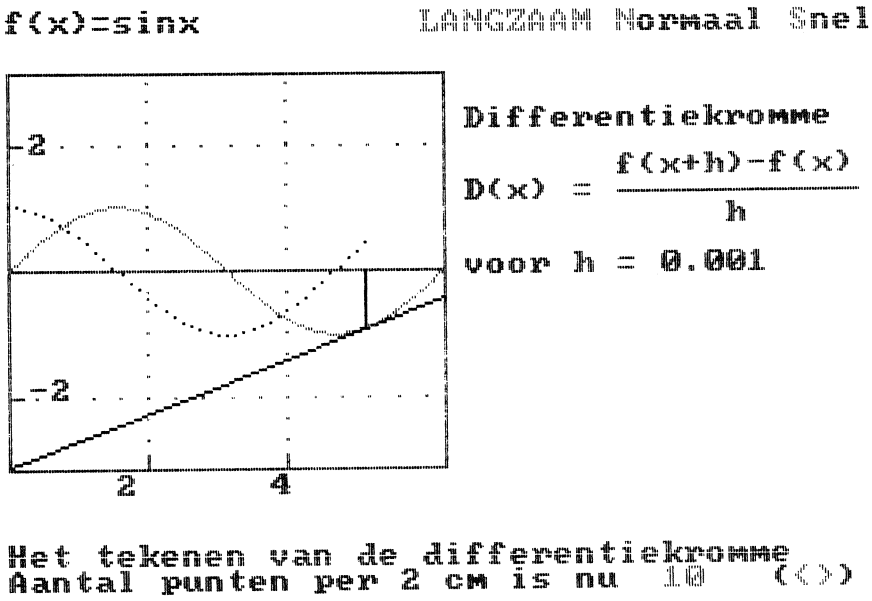
FIGUUR 12.

Deze, door de Japanner Tagaki beschreven funktie is overal continu maar nergens differentiëerbaar. Of, meer in de termen van deze voordracht geformuleerd: hoe sterk men de grafiek ook vergroot, nooit gaat hij er rechtlijnig uitzien. Volgens David Tall hebben we met deze zeer gespecialiceerde eigenschap het fundament van de analyse blootgelegd. We demonstreren dat met de funktie $f(x) = \sin x + bl(100x) / 100$. De grafiek ziet er in eerste instantie 'keurig differentieerbaar' uit, maar uitvergroten maakt duidelijk dat dit maar schijn is.

De laatste jaren is onder andere door het werk van Martin Kindt [18] de term hellingfunktie als alternatief voor 'afgeleide funktie' in het analyse-onderwijs ingeburgerd geraakt. Het is een gelukkige samenloop van omstandigheden dat deze meetkundige benadering precies past in de 'Graphic Calculus' van David Tall.

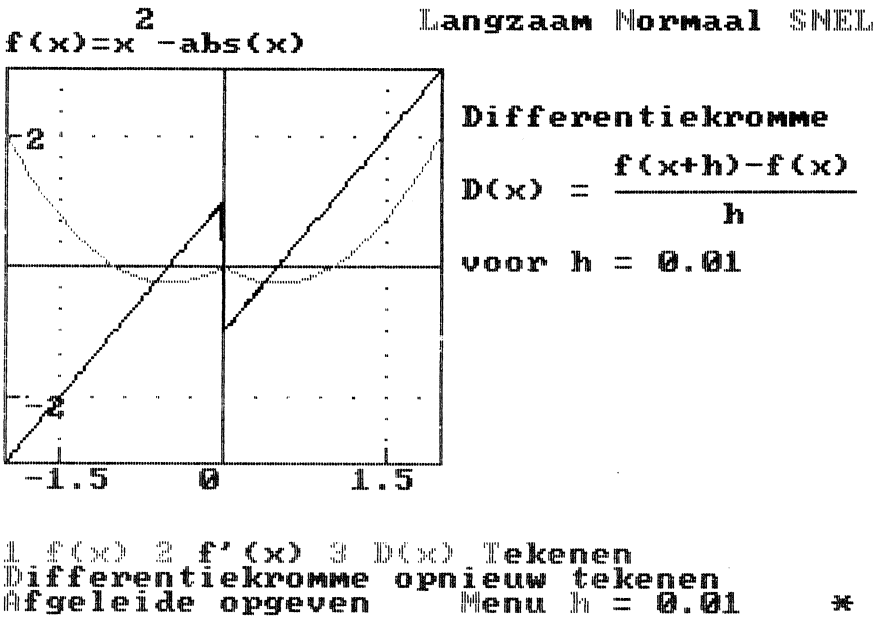
Door in een aantal punten van de grafiek van vijvoorbeeld $f(x) = \sin x$ de helling op te meten kan men de grafiek van de hellingfunktie schetsen en een vermoeden formuleren omtrent het funktievoorschrift van die hellingfunktie.

Bij dit soort handelingen komt een pakket als VU-grafiek zeer van pas.



FIGUUR 13.

Vervolgens kan men de afgeleide opgeven. Uiteraard moeten differentie-grafiek en die van de afgeliede (practisch) samenvallen. Dit type onderzoek is in enkele seconden te doen.
Dit geldt ook bij de funktie $f(x) = x^2 - abs(x)$. Deze funktie is niet differentieerbaar in $x = 0$. Wat er mis gaat in de buurt van dat punt wordt in onderstaand plaatje zichtbaar.



FIGUUR 14.

Samenvattend kunnen we vaststellen dat de micro-computer, mits voorzien van geschikte programmatuur, mogelijkheden biedt om het onderwijs in de analyse anders in te richten.

* Meer nadruk op de begripsvorming.
* Meer nadruk op het zelf experimenteren, zelf uitzoeken door de leerlingen.

Juist nu we kunnen voorzien dat binnen niet al te lange tijd de computer de algoritmiek voor zijn rekening zal nemen, is het belangrijk ook op deze mogelijkheden van de computer te wijzen. De microcomputer kan ook benut worden om leerlingen te laten oefenen in bepaalde vaardigheden. Dit kan gedemonstreerd worden met het onderdeel 'Zoek het funktie-voorschrift'. We volstaan hier met het noemen van dit onderdeel van VU-grafiek.
Wel willen we nog een ander aspect van computergebruik onder de aandacht brengen.

3.3. De computer die ons stimuleert tot het stellen van vragen over de inhoud van ons leerplan

In het voortgezet onderwijs worden vier a vijf typen vergelijkingen onderscheiden:

- lineaire vergelijkingen
- kwadratische vergelijkingen

- goniometrische vergelijkingen
- exponentiële en logarithmische vergelijkingen.

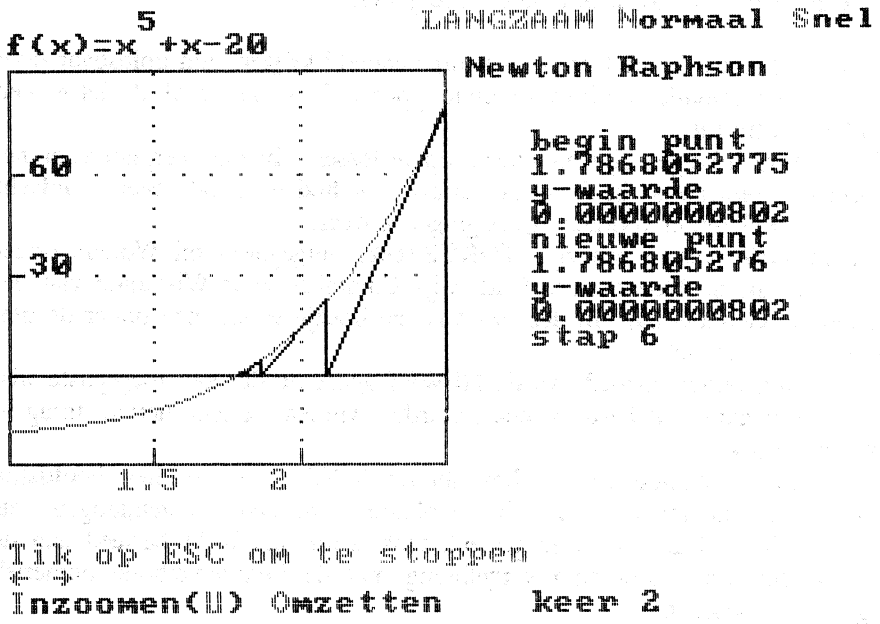
De vergelijkingen die de leerlingen voorgeschoteld krijgen, zijn oplosbaar door ze als een combinatie van bovenstaande typen te beschouwen al of niet na een handige substitutie.
Al deze vergelijkingen zijn analytisch op te lossen. In ons leerplan voor het voortgezet onderwijs hebben numerieke methoden nooit veel aandacht gekregen. Wel zijn er voorzichtige pogingen geweest
Ooit, in 1967, was er een zgn. TORUS-reeks, uitgegeven bij Wolters onder auspiciën van de Nederlandse Onderwijs Commisie voor Wiskunde van het Wiskundig Genootschap. In deze serie is een boekje verschenen onder de titel 'Inductie en Iteratie'.

Veel later, onder invloed van de HEWET-gedachte dat aan toegepaste wiskunde meer aandacht besteed moest worden, vinden we iets hiervan terug in een schoolboek.

Moderne Wiskunde in de bovenbouw VWO 4 bevat het hoofdstuk 'Numerieke methoden'. In dit hoofdstuk worden vergelijkingen als $x^5 + x - 20 = 0$ aan de orde gesteld. Ook wordt aandacht besteed aan de voorwaarden waaronder zo'n vergelijking met de iteratiemethode numeriek opgelost kan worden.

(Overigens is dit hoofdstuk in de zojuist verschenen herziene versie teruggebracht tot twee paragrafen. De auteurs zagen zich hiertoe gedwongen door de behoefte in het veld aan meer op de dagelijkse lespraktijk toegesneden leerboeken. Boeken dus waarin o.a. rekening gehouden werd met het teruggelopen aantal uren voor het vak wiskunde).

Met VU-grafiek kan bovengenoemde vergelijking numeriek opgelost worden. De volgende illustratie geeft daar een idee van.



FIGUUR 15.

Ook kunnen verschillende manieren op hun efficiëntie bekeken worden.
We kunnen ons voorstellen dat, als men in het onderwijs kan beschikken over dergelijke hulpmiddelen, er ook meer aandacht besteed gaat worden aan numerieke wiskunde.

Een principiële vraag komt dan op: hoe gebruikt men algoritmes.
Beschouwt men ze als onderdeel van een 'black box', onder het motto: vraag niet hoe het kan, maar profiteer er van, of stelt men als eis dat een leerling inzicht moet hebben in elk door hem benut algoritme? Een criterium om deze vraag te beantwoorden zou kunnen zijn: is het algoritme van belang in het leerproces van de leerling? Men kan dan verdedigen dat het iteratie-algoritme doorzien moet worden, terwijl dat bijvoorbeeld voor de simplexmethode minder duidelijk is. Overigens, weet U hoe uw rekenmachine $e^{\frac{1}{2}}$ uitrekent? En indien niet, hoe vaak belemmert U dat in uw nachtrust?

NOTEN

[1] D. KOK c.s, Raamplan NIVO-nascholing wiskunde (januari 1987).
[2] J. SPEELPENNING, Over micro computers, watermeters, badkuipen en couveuses, Nieuwe Wiskrant 2 nr. 4.
[3] P. VAN BLOKLAND, Statistiek en Computers, Euclides 61, nr. 7.
[4] ABELSON & DISESSA, Turtle Geometry, MIT Press. London, England.

[5] M. KINDT & H.B. VERHAGE, Ruimtemeetkunde met de micro, Nieuwe Wiskrant 6, nr. 2 en 3.
[6] H.A. LAUWERIER, Meetkunde met de micro, Nieuwe Wiskrant 5, nr. 1.
[7] H.B. VERHAGE, Hewet & Micro, OW&OC, februari 1986.
[8] H.B. VERHAGE, Een scheve schaats omgeturnd, Nieuwe Wiskrant 6, nr. 4.
[9] D.R. STOUTEMYER, muMath. The Soft Warehouse, Honolulu, Hawaii, 96822.
[10] E. NEUWIRTH, The impact of computer algebra for the teaching of mathematics, Proceedings IFIP, Sofia, 1987.
[11] H.C. TIJMS, Educatieve Operations Research Software: Wis en Waarachtig, Euclides 62, nr. 8.
[12] J.W. FORRESTER, World dynamics, Wright-Allen Press, Cambridge, USA.
[13] NANCY ROBERTS e.a., Introduction to computer simulation, Addison-Weseley Publishing Company, Amsterdam.
[14] Vakgroep Bedrijfsinformatica & Accountancy, Simuleren volgens de system dynamics methode, Universiteit van Amsterdam.
[15] D.O. TALL, Graphic Calculus I, II, III, Glentop Press, London, 1985.
[16] D.O. TALL, Understanding the calculus, Mathematic Teacher 110, 49-53.
[17] D.O. TALL, The gradient of a graph, Mathematic Teacher 111, 48-52.
[18] M. KINDT & J. DE LANGE, Differentiëren I, Educaboek, Culemborg.

SOFTWARE

1. Badkuip, Globgraf en Flesvuller. SLO, Enschede
2. VU-grafiek. Vrije Universiteit, Amsterdam
3. Schatten. SCO, Amsterdam
4. Een wereld rond getallen. SLO, Enschede
5. VU-stat. Vrije Universiteit, Amsterdam
6. VU-dynamo. Vrije Universiteit, Amsterdam
7. muMath. The Soft Warehous, Honolulu, Hawaii
8. Linprog. Vrije Universiteit, Amsterdam
9. Graphic Calculus I, II, III. Glentop Press, London
10. Teaching with a Micro: Math. 1, 2, 3, 4, ITMA, Shell Centre for Mathematical Education, University of Nottingham, England
11. 132 Short Programs. Mathematical Association, Stanley Thornes Ltd, Cheltenham, England.

MC SYLLABI

1.1 F. Göbel, J. van de Lune. *Leergang besliskunde, deel 1: wiskundige basiskennis.* 1965.

1.2 J. Hemelrijk, J. Kriens. *Leergang besliskunde, deel 2: kansberekening.* 1965.

1.3 J. Hemelrijk, J. Kriens. *Leergang besliskunde, deel 3: statistiek.* 1966.

1.4 G. de Leve, W. Molenaar. *Leergang besliskunde, deel 4: Markovketens en wachttijden.* 1966.

1.5 J. Kriens, G. de Leve. *Leergang besliskunde, deel 5: inleiding tot de mathematische besliskunde.* 1966.

1.6a B. Dorhout, J. Kriens. *Leergang besliskunde, deel 6a: wiskundige programmering 1.* 1968.

1.6b B. Dorhout, J. Kriens, J.Th. van Lieshout. *Leergang besliskunde, deel 6b: wiskundige programmering 2.* 1977.

1.7a G. de Leve. *Leergang besliskunde, deel 7a: dynamische programmering 1.* 1968.

1.7b G. de Leve, H.C. Tijms. *Leergang besliskunde, deel 7b: dynamische programmering 2.* 1970.

1.7c G. de Leve, H.C. Tijms. *Leergang besliskunde, deel 7c: dynamische programmering 3.* 1971.

1.8 J. Kriens, F. Göbel, W. Molenaar. *Leergang besliskunde, deel 8: minimaxmethode, netwerkplanning, simulatie.* 1968.

2.1 G.J.R. Förch, P.J. van der Houwen, R.P. van de Riet. *Colloquium stabiliteit van differentieschema's, deel 1.* 1967.

2.2 L. Dekker, T.J. Dekker, P.J. van der Houwen, M.N. Spijker. *Colloquium stabiliteit van differentieschema's, deel 2.* 1968.

3.1 H.A. Lauwerier. *Randwaardeproblemen, deel 1.* 1967.

3.2 H.A. Lauwerier. *Randwaardeproblemen, deel 2.* 1968.

3.3 H.A. Lauwerier. *Randwaardeproblemen, deel 3.* 1968.

4 H.A. Lauwerier. *Representaties van groepen.* 1968.

5 J.H. van Lint, J.J. Seidel, P.C. Baayen. *Colloquium discrete wiskunde.* 1968.

6 K.K. Koksma. *Cursus ALGOL 60.* 1969.

7.1 *Colloquium moderne rekenmachines, deel 1.* 1969.

7.2 *Colloquium moderne rekenmachines, deel 2.* 1969.

8 H. Bavinck, J. Grasman. *Relaxatietrillingen.* 1969.

9.1 T.M.T. Coolen, G.J.R. Förch, E.M. de Jager, H.G.J. Pijls. *Colloquium elliptische differentiaalvergelijkingen, deel 1.* 1970.

9.2 W.P. van den Brink, T.M.T. Coolen, B. Dijkhuis, P.P.N. de Groen, P.J. van der Houwen, E.M. de Jager, N.M. Temme, R.J. de Vogelaere. *Colloquium elliptische differentiaalvergelijkingen, deel 2.* 1970.

10 J. Fabius, W.R. van Zwet. *Grondbegrippen van de waarschijnlijkheidsrekening.* 1970.

11 H. Bart, M.A. Kaashoek. H.G.J. Pijls, W.J. de Schipper, J. de Vries. *Colloquium halfalgebra's en positieve operatoren.* 1971.

12 T.J. Dekker. *Numerieke algebra.* 1971.

13 F.E.J. Kruseman Aretz. *Programmeren voor rekenautomaten; de MC ALGOL 60 vertaler voor de EL X8.* 1971.

14 H. Bavinck, W. Gautschi, G.M. Willems. *Colloquium approximatietheorie.* 1971.

15.1 T.J. Dekker, P.W. Hemker, P.J. van der Houwen. *Colloquium stijve differentiaalvergelijkingen, deel 1.* 1972.

15.2 P.A. Beentjes, K. Dekker, H.C. Hemker, S.P.N. van Kampen, G.M. Willems. *Colloquium stijve differentiaalvergelijkingen, deel 2.* 1973.

15.3 P.A. Beentjes, K. Dekker, P.W. Hemker, M. van Veldhuizen. *Colloquium stijve differentiaalvergelijkingen, deel 3.* 1975.

16.1 L. Geurts. *Cursus programmeren, deel 1: de elementen van het programmeren.* 1973.

16.2 L. Geurts. *Cursus programmeren, deel 2: de programmeertaal ALGOL 60.* 1973.

17.1 P.S. Stobbe. *Lineaire algebra, deel 1.* 1973.

17.2 P.S. Stobbe. *Lineaire algebra, deel 2.* 1973.

17.3 N.M. Temme. *Lineaire algebra, deel 3.* 1976.

18 F. van der Blij, H. Freudenthal, J.J. de Iongh, J.J. Seidel, A. van Wijngaarden. *Een kwart eeuw wiskunde 1946-1971, syllabus van de vakantiecursus 1971.* 1973.

19 A. Hordijk, R. Potharst, J.Th. Runnenburg. *Optimaal stoppen van Markovketens.* 1973.

20 T.M.T. Coolen, P.W. Hemker, P.J. van der Houwen, E. Slagt. *ALGOL 60 procedures voor begin- en randwaardeproblemen.* 1976.

21 J.W. de Bakker (red.). *Colloquium programmacorrectheid.* 1975.

22 R. Helmers, J. Oosterhoff, F.H. Ruymgaart, M.C.A. van Zuylen. *Asymptotische methoden in de toetsingstheorie; toepassingen van naburigheid.* 1976.

23.1 J.W. de Roever (red.). *Colloquium onderwerpen uit de biomathematica, deel 1.* 1976.

23.2 J.W. de Roever (red.). *Colloquium onderwerpen uit de biomathematica, deel 2.* 1977.

24.1 P.J. van der Houwen. *Numerieke integratie van differentiaalvergelijkingen, deel 1: eenstapsmethoden.* 1974.

25 *Colloquium structuur van programmeertalen.* 1976.

26.1 N.M. Temme (ed.). *Nonlinear analysis, volume 1.* 1976.

26.2 N.M. Temme (ed.). *Nonlinear analysis, volume 2.* 1976.

27 M. Bakker, P.W. Hemker, P.J. van der Houwen, S.J. Polak, M. van Veldhuizen. *Colloquium discretiseringsmethoden.* 1976.

28 O. Diekmann, N.M. Temme (eds.). *Nonlinear diffusion problems.* 1976.

29.1 J.C.P. Bus (red.). *Colloquium numerieke programmatuur, deel 1A, deel 1B.* 1976.

29.2 H.J.J. te Riele (red.). *Colloquium numerieke programmatuur, deel 2.* 1977.

30 J. Heering, P. Klint (red.). *Colloquium programmeeromgevingen.* 1983.

31 J.H. van Lint (red.). *Inleiding in de coderingstheorie.* 1976.

32 L. Geurts (red.). *Colloquium bedrijfsystemen.* 1976.

33 P.J. van der Houwen. *Berekening van waterstanden in zeeën en rivieren.* 1977.

34 J. Hemelrijk. *Oriënterende cursus mathematische statistiek.* 1977.

35 P.J.W. ten Hagen (red.). *Colloquium computer graphics.* 1978.

36 J.M. Aarts, J. de Vries. *Colloquium topologische dynamische systemen.* 1977.

37 J.C. van Vliet (red.). *Colloquium capita datastructuren.* 1978.

38.1 T.H. Koornwinder (ed.). *Representations of locally compact groups with applications, part I.* 1979.

38.2 T.H. Koornwinder (ed.). *Representations of locally compact groups with applications, part II.* 1979.

39 O.J. Vrieze, G.L. Wanrooy. *Colloquium stochastische spelen.* 1978.

40 J. van Tiel. *Convexe analyse.* 1979.

41 H.J.J. te Riele (ed.). *Colloquium numerical treatment of integral equations.* 1979.

42 J.C. van Vliet (red.). *Colloquium capita implementatie van programmeertalen.* 1980.

43 A.M. Cohen, H.A. Wilbrink. *Eindige groepen (een inleidende cursus).* 1980.

44 J.G. Verwer (ed.). *Colloquium numerical solution of partial differential equations.* 1980.

45 P. Klint (red.). *Colloquium hogere programmeertalen en computerarchitectuur.* 1980.

46.1 P.M.G. Apers (red.). *Colloquium databankorganisatie, deel 1.* 1981.

46.2 P.G.M. Apers (red.). *Colloquium databankorganisatie, deel 2.* 1981.

47.1 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60: general information and indices.* 1981.

47.2 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60, vol. 1: elementary procedures; vol. 2: algebraic evaluations.* 1981.

47.3 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60, vol. 3A: linear algebra, part I.* 1981.

47.4 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60, vol. 3B: linear algebra, part II.* 1981.

47.5 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60, vol. 4: analytical evaluations; vol. 5A: analytical problems, part I.* 1981.

47.6 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60, vol. 5B: analytical problems, part II.* 1981.

47.7 P.W. Hemker (ed.). *NUMAL, numerical procedures in ALGOL 60, vol. 6: special functions and constants; vol. 7: interpolation and approximation.* 1981.

48.1 P.M.B. Vitányi, J. van Leeuwen, P. van Emde Boas (red.). *Colloquium complexiteit en algoritmen, deel 1.* 1982.

48.2 P.M.B. Vitányi, J. van Leeuwen, P. van Emde Boas (red.). *Colloquium complexiteit en algoritmen, deel 2.* 1982.

49 T.H. Koornwinder (ed.). *The structure of real semisimple Lie groups.* 1982.

50 H. Nijmeijer. *Inleiding systeemtheorie.* 1982.

51 P.J. Hoogendoorn (red.). *Cursus cryptografie.* 1983.

CWI SYLLABI

1 Vacantiecursus 1984: *Hewet - plus wiskunde.* 1984.

2 E.M. de Jager, H.G.J. Pijls (eds.). *Proceedings Seminar 1981-1982. Mathematical structures in field theories.* 1984.

3 W.C.M. Kallenberg, et al. *Testing statistical hypotheses: worked solutions.* 1984.

4 J.G. Verwer (ed.). *Colloquium topics in applied numerical analysis, volume 1.* 1984.

5 J.G. Verwer (ed.). *Colloquium topics in applied numerical analysis, volume 2.* 1984.

6 P.J.M. Bongaarts, J.N. Buur, E.A. de Kerf, R. Martini, H.G.J. Pijls, J.W. de Roever. *Proceedings Seminar 1982-1983. Mathematical structures in field theories.* 1985.

7 Vacantiecurus 1985: *Variatierekening.* 1985.

8 G.M. Tuynman. *Proceedings Seminar 1983-1985. Mathematical structures in field theories, Vol.1 Geometric quantization.* 1985.

9 J. van Leeuwen, J.K. Lenstra (eds.). *Parallel computers and computations.* 1985.

10 Vacantiecursus 1986: *Matrices.* 1986.

11 P.W.H. Lemmens. *Discrete wiskunde: tellen, grafen, spelen en codes.* 1986.

12 J. van de Lune. *An introduction to Tauberian theory: from Tauber to Wiener.* 1986.

13 G.M. Tuynman, M.J. Bergvelt, A.P.E. ten Kroode. *Proceedings Seminar 1983-1985. Mathematical structures in field theories, Vol.2.* 1987.

14 Vacantiecursus 1987: *De personal computer en de wiskunde op school.* 1987.