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

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

MC SYLLABUS 29.1a

J.C.P. BUS (RED.)

COLLOQUIUM NUMERIEKE PROGRAMMATUUR

DEEL 1a

MATHEMATISCH CENTRUM AMSTERDAM 1976

AMS(MOS) subject classification scheme (1970): 65-01

ISBN 90 6196 128 9

Inhoud

Inhoud

VOORWOORD

Het Colloquium Numerieke Programmatuur, waarvan het eerste deel werd gehouden in de eerste helft van 1976, werd georganiseerd door de Afdeling Numerieke Wiskunde van het Mathematisch Centrum. Het doel van de lezingen was voornamelijk om geïnteresseerden vertrouwd te maken met het gebruik van de programmatheken ACCULIB (ALGOL 60 en FORTRAN), IMSL (FORTRAN), NAG (ALGOL 60 en FORTRAN) en NUMAL (ALGOL 60). Het colloquium was dan ook vooral bestemd voor, wat men bij de rekencentra noemt, de gebruiker. Hiermee worden de vele personen aangeduid, die voor de oplossing van hun fysische, chemische, statistisch of andere problemen gebruik kunnen maken van numerieke algoritmen die in de vorm van routines beschikbaar zijn in de genoemde programmatheken bij de verschillende rekencentra. Er is dan ook zoveel mogelijk geprobeerd om aan de hand van konkrete praktijkproblemen het gebruik van de beschikbare programmatuur toe te lichten. Het gebied van de numerieke wiskunde dat in het eerste deel van dit colloquium is bestreken is zeer ruim. Het omvat lineaire algebra, differentiaalvergelijkingen, speciale funkties, Fouriertransformaties en optimalisering.

Wij hopen dat de twee boekjes waarin de lezingen van dit colloquium zijn weergegeven voor de bovengenoemde gebruiker een nuttige gids kunnen zijn bij de selektie van de juiste programmatuur voor de oplossing van zijn problemen. Tenslotte wil de redakteur zijn dank betuigen aan degenen die aan de totstandkoming van deze syllabus hebben meegewerkt, met name ook aan mevr. I. Pins voor het typen van het manuscript en aan D. Zwarts en J. Schipper voor het drukken en binden van deze boekjes.

J.C.P. B.

1. LINEAIRE ALGEBRA

1.1. Oplossing van stelsels lineaire vergelijkingen en inversie

door J.C.P. Bus
(Mathematisch Centrum)

1.1.1. Inleiding

Bezien we het totale gebied van de numerieke wiskunde dan kunnen we wel konkluderen dat de numerieke (lineaire) algebra één van de meest uitgewerkte deelgebieden is. Met name voor de oplossing van stelsels lineaire vergelijkingen bestaan vele goed geanalyseerde algoritmen en iedere programmatheek bevat dan ook een uitgebreide verzameling programmatuur op dit gebied. Wij zullen ons in deze bijdrage beperken tot vier programmatheken welke op het SARA-systeem beschikbaar zijn of binnenkort beschikbaar komen. Dit zijn:

ACCULIB	(ALGOL 60 en FORTRAN);
IMSL	(FORTRAN);
NAG	(ALGOL 60 en FORTRAN);
NUMAL	(ALGOL 60).

Het doel van deze bijdrage is tweeledig. Enerzijds willen wij een overzicht geven van de programmatuur voor het oplossen van stelsels lineaire vergelijkingen en inversie, welke beschikbaar is in deze vier programmatheken. Anderzijds willen we aan de hand van een probleem uit de molecuulfysika, de knelpunten aangeven bij het gebruik van deze programmatuur voor het oplossen van bepaalde problemen.

1.1.2. Overzicht van de beschikbare programmatuur

We beschouwen het probleem een lineair stelsel

(1.1.2.1) $Ax = b$

op te lossen, waarbij A een $(n \times m)$-matrix, x een m-vektor en b een n-vektor is. Dikwijls zal men bij een gegeven matrix de oplossing van (1.1.2.1) willen vinden voor verschillende rechterleden b. We kunnen dit noteren als: los op

(1.1.2.1) $AX = B$,

waarbij B een $(n \times p)$-matrix is van p rechterleden (kolommen van B); de kolommen van de $(m \times p)$-matrix X zijn dan de overeenkomstige oplossingen.

We zullen een overzicht van de programmatuur in de vier beschouwde programmatheken geven door middel van een aantal tabellen. In tabel 1 wordt een ruwe indeling gegeven van de verschillende stelsels lineaire vergelijkingen, waarbij naar de tabellen 2 t/m 6 wordt verwezen voor de te gebruiken procedures of routines. In tabel 7 wordt de programmatuur voor het inverteren van een matrix gegeven en in tabel 8 die voor het berekenen van de pseudo-inverse. De pseudo-inverse A^+ van een matrix A wordt gedefinieerd door:

$$AA^+A = A, \qquad A^+AA^+ = A^+,$$
$$(A^+A)^T = A^+A, \qquad (AA^+)^T = AA^+.$$

A^+ is hierdoor éénduidig gedefinieerd voor elke A. Voor de berekening van A^+ zij verwezen naar sektie 1.2.

TABEL 1

Ruwe indeling van stelsels lineaire vergelijkingen.

Eigenschappen van de matrix	Verwijzing naar tabel
reëel,vol n = m, rang = n	2
reëel,vol n > m, rang = m kleinste-kwadraten-oplossing	3
reëel, rang ≤ min(n,m)	4
reëel, ijl n = m, rang = n	5
complex	6

In de verschillende tabellen worden routines, die uitsluitend geschikt zijn voor het oplossen van een lineair stelsel met één rechterlid, aangegeven door een asterisk (*) voor de naam. Bij de routines uit ACCULIB duidt (A) en (F) op de programmeertaal ALGOL 60 resp. FORTRAN. Evenzo duidt

bijvoorbeeld F04AAA/F in de NAG programmatheek op de routines F04AAA in ALGOL 60 en F04AAF in FORTRAN. IMSL bevat alleen FORTRAN subroutines, terwijl de NUMAL uitsluitend ALGOL 60 procedures bevat.

TABEL 2

Programmatuur voor het oplossen van lineaire stelsels
met een reële, volle, niet-singuliere vierkante matrix.

geen nadere specificatie

Alleen oplossen

ACCULIB : *DETSOL (A)
DET + SOL (A)
*CROSOL (F)
CRODEC + SOLDEC (F)

IMSL : LEQT1F

NAG : F04AAA/F
*F04ARA/F

NUMAL : *decsol
dec + sol
*gsssol
gsselm + solelm

Oplossen met iteratief verbeteren

IMSL : LEQT2F

NAG : F02AEA/F
*F04ATA/F

NUMAL : *gssitisol
gsselm + itisol

Oplossen met berekening bovengrens fout

NUMAL : *gsssolerb
gsserb + solelm
*gssitisolerb
gssnri + itisolerb

symmetrische positief definiete matrix

Alleen oplossen

ACCULIB : SYMDEC + SYMSOLE (A)
*DETSOLSYM2 (A)
DETSYM2 + SOLSYM2 (A)
*DSOLSY (F)
CHODEC + SOLSYM (F)

IMSL : LEQT1P

NAG : F01BFA/F + F04AQA/F

NUMAL : *chldecsol1
chldec1 + chlsol1
*chdecsol2
chldec2 + chlsol2

Oplossen met iteratief verbeteren

IMSL : LEQT2P

NAG : F04ABA/F
*F04ASA/F

symmetrische niet-positief definiete matrix

Alleen oplossen

IMSL : LEQ1S [1)]

NUMAL : *symdecsol1 [1)]
symdec1 + symsol1 [1)]
*symdecsol2 [1)]
symdec2 + symsol2 [1)]

Oplossen met ieteratief verbeteren

IMSL : LEQ2S

[1)] De methode, die gebruikt is in de IMSL programmatheek is stabiel voor indefiniete matrices. Dit geldt niet voor de methode die is gebruikt in de NUMAL.

TABEL 3

Programmatuur voor het oplossen van lineaire overbepaalde stelsels met reële matrix van volle rang; kleinste-kwadratenoplossing.

Alleen oplossen

ACCULIB : *LEASTSQ (A)
DECOMPOSE + SOLVLQ (A)
DCMPOS + SOLVLQ (F)

NAG : F01AXA/F + F04ANA/F

NUMAL : *lsqortdecsol
lsqortdec + lsqsol

Oplossen met iteratief verbeteren

NAG : F04AMA/F

TABEL 4

Programmatuur voor het oplossen van lineaire stelsels met reële matrix van orde n×m; niet noodzakelijk van volle rang.

Oplossen overbepaalde stelsels (n>m)

IMSL : LLSQAR

NUMAL : *solovr
qrisngvaldec + solsvdovr

Oplossen onderbepaalde stelsels (n<m)

NUMAL : *solund
qrisngvaldec + solsvdund

Oplossen homogene vergelijking
(rechterlid is nulvektor)

NUMAL : homsol

TABEL 5

Programmatuur voor het oplossen van stelsels lineaire vergelijkingen met vierkante reële, ijle, niet-singuliere matrix.

Direkte methoden		
geen nadere specificatie		NAG : F03AJA/F of F03AKA/F + F04APA/F
bandmatrix	Alleen oplossen	IMSL : LEQT1B NUMAL : *decsolbnd decbnd + solbnd
	Oplossen met iteratief verbeteren	IMSL : LEQT2B
symmetrische bandmatrix (pos. definiet)	Alleen oplossen	IMSL : LEQ1PB NAG : F04ACA/F NUMAL : *chldecsolbnd chldecbnd + chlsolbnd
	Oplossen met iteratief verbeteren	IMSL : LEQ2PB
tridiagonaal-matrix	Alleen oplossen	NUMAL : *decsoltri dectri + soltri *decsoltripiv dectripiv + soltripiv
symmetrische pos. definiete tridiagonaal-matrix	Alleen oplossen	NUMAL : *decsolsymtri decsymtri + solsymtri

Iteratieve methoden

ACCULIB	:	CG(A)
NUMAL	:	conj grad
		conj resi

TABEL 6

Programmatuur voor het oplossen van stelsels lineaire vergelijkingen met vierkante, niet-singuliere, complexe matrix.

Alleen oplossen

IMSL	:	LEQT1C
NAG	:	F04ADA/F

Oplossen met iteratief verbeteren

IMSL	:	LEQT2C

TABEL 7

Programmatuur voor inversie van een reële, niet-singuliere matrix.

geen nadere specifikatie

Alleen inversie

ACCULIB	:	DETINV (A)
		CRODEC + INVDEC (F)
IMSL	:	LINV1F
		LINV3F
NAG	:	F01AA/F
NUMAL	:	decinv
		gssinv

Inversie met iteratief verbeteren

IMSL	:	LINV2F

Inversie met bovengrens fout

NUMAL	:	gssinverb

symmetrische positief definiete matrix	Alleen inversie		
	ACCULIB	:	DETINVSYM2 (A)
			CHODEC + INVSYM (F)
	IMSL	:	LINV1P
	NAG	:	F01ADA/F
	NUMAL	:	chldecinv1
			chldecinv2
	Inversie met iteratief verbeteren		
	IMSL	:	LINV2P
	NAG	:	F01ABA/F
			F01ACA/F
symmetrische niet-noodzakelijk positief definiete matrix	Alleen inversie		
	NUMAL	:	symdecinv1 [1]
			symdecinv2 [1]
symmetrische pos. definiete bandmatrix	Alleen inversie		
	IMSL	:	LIN1PB
	Inversie met iteratief verbeteren		
	IMSL	:	LIN2PB

[1] De methode, die gebruikt is in de IMSL programmatheek is stabiel voor indefiniete matrices. Dit geldt niet voor de methode die is gebruikt in de NUMAL.

TABEL 8

Programmatuur voor de berekening van de pseudo-inverse van een matrix.

IMSL	:	LPSDOR
NUMAL	:	psdinv

N.B. Meestal zal berekening van de pseudo-inverse niet nodig zijn en is een singuliere waarden dekompositie voldoende, evenals inversie van een matrix meestal niet nodig is maar de ontbinding voldoende.

Opmerkingen

1. Als ergens in de tabellen twee routines tezamen worden gegeven, bijv. in tabel 2: dec + sol, dan wordt daarmee aangeduid dat voor de oplossing van het probleem eerst de eerste routine (dec) moet worden aangeroepen, gevolgd door één of meer aanroepen van de tweede routine (sol).
2. Bij verschillende routines in de IMSL programmatheek bestaat de mogelijkheid een test op de precisie uit te voeren door een zekere integer waarde aan de parameter IDGT mee te geven. Als aan de test is voldaan dan betekent dit dat de gevonden oplossing de exacte oplossing is van een lineair stelsel met een matrix die in IDGT decimalen overeenstemt met de gegeven matrix. Het gebruik van deze parameter kan tot verwarring leiden omdat het niets zegt omtrent het aantal korrekte decimalen in de berekende oplossing. Dit blijkt duidelijk uit het volgende voorbeeld.

We losten een lineair stelsel op met de routines LEQT1F en LEQT2F. De konditie van de matrix was 10^8. We deden dit voor IDGT = 0,2,4,6,8,10,12, 14. Het bleek dat LEQT1F een oplossing berekende met een relatieve fout van $2_{10} - 7$, terwijl alleen voor IDGT = 14 een waarschuwing werd gegeven over de precisie.

De situatie bij gebruik van LEQT2F waarvan wordt gezegd dat de oplossing in hoge precisie wordt berekend is nog verwarder. We geven de resultaten in tabel 9.

Voor IDGT = 14 wordt gemeld dat slechts 6 decimalen niet veranderden na iteratief verbeteren, wat de indruk geeft als zou de gevonden oplossing slechts in 6 decimalen korrekt zijn.

Het blijkt dus dat de gebruiker, indien hij niet weet dat de konditie van de matrix ongeveer 1 is, beter IDGT = 0 kan invoeren als hij LEQT2F gebruikt en een zo hoog mogelijke precisie in de oplossing wil verkrijgen. In dat geval wordt namelijk geen precisie test uitgevoerd en altijd iteratief verbeterd. Dezelfde opmerkingen gelden ook voor de andere routines die deze parameter hebben.

TABEL 9

Gebruik van LEQT2F voor oplossing van een slecht gekonditioneerd stelsel voor verschillende waarden van IDGT.

IDGT	fout in berekende oplossing	waarschuwing gegeven
0	4_{10}-15	nee
2	2_{10}-17	nee
4	2_{10}- 7	nee
6	2_{10}- 7	nee
8	2_{10}- 7	nee
10	2_{10}- 7	nee
12	2_{10}- 7	nee
14	4_{10}-15	ja

1.1.3. Gebruik van de programmatuur aan de hand van een probleem uit de molecuul-fysika.

Het fysisch probleem waar het hier om gaat wordt uitgebreid beschreven door HUBERS & LOS [1975]. Wij zullen kort op de achtergronden ingaan.

Bij reaktieve botsingen tussen zekere atomen en moleculen bij variabele temperatuur wordt de doorsnede voor de produktie van een specifiek ion bij gegeven temperatuur T gegeven door

$$(1.1.3.1)\quad Q(E_{rel},\beta) = \int_0^\infty f(E_i,\beta)\sigma(E_{rel},E_i)dE_i,$$

met E_{rel} : relatieve kinetische energie;
$\beta = 1/kT$, k is de Boltzmann-konstante;
E_i : totale trillingsenergie van het molecuul;
$f(E_i,\beta)$: de fraktie moleculen die trillingsenergie E_i hebben bij temperatuur T;
$\sigma(E_{rel},E_i)$: de specifieke doorsnede.

Het doel is $\sigma(E_i) = \sigma(E_{rel},E_i)$ te bepalen bij gegeven relatieve kinetische energie. Zij nu $\rho(E_i)$ de dichtheid van de moleculaire trillingstoestanden bij energie E_i dan geldt voor de z.g. verdelings-funktie

$$Z(\beta) = \int_0^\infty \rho(E_i)\ \exp(-E_i\beta)dE_i;$$

$$(1.1.3.2)\quad Z(\beta) = 1/\prod_{j=1}^{n} 2\ \sinh(\beta h\nu_j/2),$$

waarbij n het aantal trillingstoestanden is, ν_j (j=1,...,n) de verschillende frequenties en h een schalingskonstante. Met gebruik van de Maxwell-Boltzmann distributie en de experimenteel bepaalde relatie

$$(1.1.3.3)\quad Q(E_{rel},\beta) = C\exp(A/\beta),$$

voor zekere konstanten C en A, krijgen we na enig omwerken

$$(1.1.3.4)\quad Z(\beta)Q(\beta) = \mathfrak{L}(\rho(E_i')\sigma_1(E_i')),$$

waarbij E_i' en σ_1 door eenvoudige transformaties uit E_i en σ worden verkregen en $\mathfrak{L}(f(x))$ de Laplace getransformeerde van f(x) aanduidt. Dus uit (1.1.3.2), (1.1.3.3) en (1.1.3.4) krijgen we

$$(1.1.3.5)\quad \mathfrak{L}(\rho(E_i)\sigma_1(E_i)) = C\exp(A/\beta)\ \frac{1}{2^n}\prod_{j=1}^{n}[\sinh(\beta h\nu_j/2)]^{-1}.$$

Van het rechterlid is niet analytisch de Laplace-inverse te berekenen. Daarom willen we op het interval $[\lambda_0,\lambda_1]$ de funktie

$$\frac{1}{2^n\lambda}\prod_{j=1}^{n}\frac{1}{\sinh(h\nu_j/2\lambda)}$$

benaderen door een polynoom in λ van graad N

$$P_N(\lambda) = x_0 + x_1\lambda + \ldots + x_N\lambda^N.$$

Dan geldt namelijk onder zekere voorwaarden

$$\rho(E_i')\sigma_1(E_i') \approx C \sum_{\mu=0}^{n} \left(\frac{E_i'}{A}\right)^{\mu/2} x_\mu I_\mu(2\sqrt{AE_i'}),$$

met I_μ de μ-de orde gemodificeerde Besselfunktie en $\lambda = 1/\beta$; zodat bij gegeven ρ dan σ_1 en dus σ kan worden berekend.

Na transformatie, diskretisatie en gebruik van de euclidische norm komen we op het volgende probleem.

Bepaal een polynoom $P_N(t) = x_0 + x_1 t + \ldots + x_N t^N$ van graad N op het interval [0,1] , zodanig dat

$$(1.1.3.6) \quad \phi_n(x_0,\ldots,x_N) = \sum_{i=0}^{m} (f(t_i) - P_N(t_i))^2$$

minimaal is, waarbij m + 1 het aantal diskretisatiepunten is,

$$t_i = i/m, \qquad i = 0,\ldots,m,$$

$$f(t) = \frac{1}{2^n \lambda(t)} \prod_{j=1}^{n} 1/(\sinh(h\nu_j/2\lambda(t)));$$

$$\lambda(t) = 200k(4t+1); \qquad n = 15;$$
$$k = 8.617065_{10}-5;$$
$$h\nu_1 = 0.09590;$$
$$h\nu_i = 0.07956, \qquad i = 2,3;$$
$$h\nu_i = 0.1164, \qquad i = 4,5,6;$$
$$h\nu_i = 0.0761, \qquad i = 7,8,9;$$
$$h\nu_i = 0.0651, \qquad i = 10,11,12;$$
$$h\nu_i = 0.0430, \qquad i = 13,14,15.$$

Wij kiezen m = 32.

Wij bepalen de beste graad N als volgt. Onder de aanname van enige statistische veronderstellingen, geldt dat de variantie

$$\sigma_k^2 = \phi_k(x_0,\ldots,x_k)/(m-k-1)$$

onafhankelijk is van k, voor $k \geq N$.
In de praktijk betekent dit dat we σ_k^2 berekenen voor $k = 1,2,\ldots,$ $k < m - 1$. Zolang σ_{k+1}^2 voldoende kleiner is dan σ_k^2, is $P_{k+1}(t)$ een betere benadering van f(t) en kennelijk is σ^2 nog afhankelijk van k. Als σ_k^2 niet meer voldoende daalt voor $k > k_1$, dan kiezen we $N = k_1$. (Zie ook FORSYTHE [1957] en de daarin genoemde referenties.) Opgemerkt moet worden dat dit niet een zeer efficiente benadering is omdat m polynoom evaluaties nodig zijn voor de berekening van σ_k^2, nog naast de oplossing van het optimaliseringsprobleem. Dit is echter niet bezwaarlijk, omdat ons doel niet is het gestelde probleem zo efficient mogelijk op te lossen, doch veeleer om de problemen te schetsen die men ervaart wanneer het met de voor de hand liggende methoden wordt opgelost.

Noteren we voor zekere $k \geq 1$

$$F = (f(t_0), f(t_1),\ldots,f(t_m))^T$$

en

$$x = (x_0,\ldots,x_k)^T,$$

dan krijgen we voor (1.1.3.6)

(1.1.3.7) $\phi(x) = (Vx-F)^T (Vx-F)$,

waarbij V de zg. Vandermonde matrix is:

$$V = \begin{pmatrix} 1 & t_0 & \ldots & t_0^k \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 1 & t_m & & t_m^k \end{pmatrix}.$$

Het is eenvoudig te bewijzen dat het minimum van $\phi(x)$ éénduidig is bepaald door

$$\frac{d}{dx}\,\phi(x) = 0.$$

Dit leidt tot de oplossing van het lineaire stelsel met symmetrische positief definiete matrix:

(1.1.3.8) $V^TVx = V^TF$

Een dergelijke vergelijking staat ook bekend als de normaal-vergelijking behorend bij het overbepaalde lineaire stelsel $Vx = F$.

Direkte toepassing van de in paragraaf 1.1.2. gegeven tabellen voor de oplossing van de vergelijking (1.1.3.8) leidt tot routines die Cholesky's methode gebruiken (tabel 2). We hebben dit geprogrammeerd op een aantal verschillende manieren.

1. Met gebruik van de NUMAL:
   a. door eenvoudig oplossen van (1.1.3.8) met chldecsol2, waarbij V^TV en V^TF in enkele precisie worden berekend;
   b. door eenvoudig oplossen van (1.1.3.8) met chldecsol2, waarbij V^TV en V^TF in dubbele precisie worden berekend.
2. Met gebruik van de IMSL:
   door oplossen van (1.1.3.8) met LEQT2P.
3. Met gebruik van de NAG(FORTRAN):
   oplossen van (1.1.3.8) met F04ASF.

N.B. In de IMSL en NAG programmatheek worden de inwendige produkten altijd in dubbele precisie berekend, zodat een onderscheid zoals bij NUMAL niet mogelijk is. Deze programma's worden gegeven in de bijlage, evenals de resultaten.

De vraag die we ons kunnen stellen is: wat is de relatieve fout in de verkregen coëfficiënten? De relatieve fout in de oplossing van het vierkante lineaire stelsel van orde n:

$$Ax = b$$

is evenredig met:

(1.1.3.9) $\|\delta x\| \,/\, \|x\| \approx \kappa(A)g(n)\varepsilon,$

waarbij ε de machine-precisie is, $g(n)$ een funktie die afhangt van de orde n en $\kappa(A)$ is het konditiegetal van de matrix A:

$$\kappa(A) = \|A\| \, \|A^{-1}\|.$$

Beschouwen we de matrix van het stelsel (1.1.3.8) voor grote m dan geldt voor het (i,j)-de element:

$$(V^T V)_{ij} = \sum_{\mu=0}^{m} t_\mu^i t_\mu^j \approx m \int_0^1 t^i t^j dt = m/(i+j+1).$$

Dus $V^T V$ lijkt, afgezien van een konstante faktor m, op het (k+1)-ste segment van de Hilbertmatrix. Het konditiegetal hiervan loopt snel op bij toenemende orde. (Voor orde 10 is het konditiegetal ongeveer 10^{+12}).

Bij de gegeven machine-precisie van 10^{-14} kunnen we dus op grond van (1.1.3.9) wel konkluderen dat bij de orde 12 of 13 mogelijk geen enkel cijfer in de resultaten meer zinvol is. Hoe ernstig dit in de praktijk is blijkt ook uit de verschillen in resultaten van de programma's 1a en 1b. Hoewel de in dubbele precisie berekende matrix en rechterlid slechts ongeveer 10^{-12} verschillen, is de een singulier bij orde 12 en de ander bij orde 13, terwijl ook de varianties verschillen. In de beschikbare programmatheken bestaat niet de mogelijkheid een bovengrens voor de fout te berekenen bij Cholesky's methode. Dit kan wel als we gaus-eliminatie gebruiken en dus geen gebruik maken van de symmetrie, met behulp van de NUMAL. In de bijlage wordt een programma gegeven waarin de oplossing van (1.1.3.8) met gauss-eliminatie wordt berekend. Een a-priori bovengrens voor de fout wordt berekend (deze is meestal vrij pessimistisch); als deze grens kleiner is dan 10^{-3} (ongeveer de relatieve precisie van de data) dan wordt het stelsel gewoon opgelost, anders wordt de oplossing ook iteratief verbeterd en wordt een nieuwe, realistische bovengrens voor de fout berekend. Het blijkt dat, als de a-priori fout te groot wordt, ook iteratief verbeteren geen redelijke foutgrens meer geeft.

Een stabieler wijze om de oplossing van (1.1.3.8) te berekenen is met behulp van Householder orthogonalisatie. We ontbinden de matrix V:

$$V = QR,$$

waarbij Q een orthogonale (m+1) × (k+1) matrix is en R een (k+1) × (k+1) bovendriehoeksmatrix. Als V, en dus R, niet singulier is krijgen we

(1.1.3.10) $Rx = Q^T F$.

Deze oplossingsmethode is equivalent met de berekening van de *kleinste-kwadratenoplossing* van het *overbepaalde lineaire stelsel*

(1.1.3.11) $Vx = F$.

Met behulp van tabel 3 komen we tot de volgende programma's:

1. met NUMAL: oplossing met lsqortdecsol;
2. met NAG: oplossing met F04AMF.

Programma's en resultaten worden gegeven in de bijlage. Het blijkt dat de coëfficiënten die we nu krijgen volkomen verschillen van die welke we verkregen met Cholesky's methode (in dubbele lengte berekende matrix). Ook breekt de berekening nu pas af bij graad 21 i.p.v. bij graad 13. De relatieve fout bij deze berekening voldoet ook aan (1.1.3.9), maar nu voor het stelsel (1.1.3.11). Dus het konditiegetal is $\kappa(V) = \sqrt{\kappa(V^T V)}$ als we de spectraalnorm gebruiken.

Een methode die eveneens een kleinste-kwadratenoplossing berekent van (1.1.3.11) kan worden verkregen met behulp van de singuliere waarden ontbinding (zie sektie 1.2). Dit is mathematisch equivalent met de berekening van

$$x = V^+ F,$$

waarbij V^+ de pseudo-inverse is van V. Het voordeel van deze methode is dat hij ook stabiel is als de matrix in de gegeven rekenprecisie niet van volle rang is. Met gebruik van tabel 4 zien we dat de procedure solovr uit NUMAL en de routine LLSQAR uit IMSL hiervoor beschikbaar zijn. Programma's en resultaten zijn gegeven in de bijlage. Bij vergelijking van solovr met de met Householder orthogonalisatie uit de NUMAL verkregen resultaten blijkt dat slechts de grootste coëfficienten in één à twee cijfers overeenstemmen.

De oorzaak van de problemen die we hier signaleren ligt in wezen bij de wijze waarop wij het te bepalen polynoom representeren. Wij schreven

$$P_N(t) = \sum_{\mu=0}^{N} x_\mu t^\mu .$$

Echter, de polynomen t^μ, $\mu = 0,1,\ldots,N$, vormen niet een voldoende onafhankelijke basis voor de ruimte van polynomen van graad N, voor wat grotere waarden van N. Dit is de oorzaak van de (numerieke) afhankelijkheid van de kolommen in de matrix V en dus van de slechte konditie van V. We kunnen $P_N(t)$ representeren met behulp van orthogonale polynomen:

$$P_N(t) = \sum_{\mu=0}^{N} x_\mu q_\mu(t),$$

waarbij $q_\mu(t), \mu = 0,1,\ldots,N$, polynomen zijn van graad $\leq N$ welke voldoen aan

$$\sum_{i=0}^{m} q_\mu(t_i)\, q_\nu(t_i) = \begin{cases} 0, & \mu \neq \nu, \\ \neq 0, & \mu = \nu. \end{cases}$$

De matrix van het stelsel lineaire vergelijkingen die we op deze wijze krijgen is een diagonaalmatrix en dus is dit stelsel eenvoudig en op stabiele wijze oplosbaar. Voor een nadere beschouwing over approximatie van funkties met behulp van orthogonale polynomen zij verwezen naar FORSYTHE [1957].

Het probleem de coëfficiënten te bepalen van een polynoom $P_n(t)$ zodat ϕ_n (cf. (1.1.3.6)) optimaal is blijft echter een slecht gesteld probleem omdat bij hoge graad, het verschil tussen twee polynomen $P_N(t)$ en $P'_N(t)$ op een interval klein kan zijn, terwijl toch de coëfficiënten aanzienlijk verschillen. Oplossing van (1.1.3.5) op een andere wijze dan door middel van polynoom fitten moet dan ook zeker worden overwogen.

Literatuur

FORSYTHE, G.E. [1957], *Generation and use of orthogonal polynomials for data-fitting with a digital computer*, J. SIAM. 5, 74-88.

FORSYTHE, G.E. [1970], *Pitfalls in computation, or why a math. book isn't enough*, Stanford Univ. TR. CS 147.

HUBERS, M.M. & J. LOS [1975], *Ion pair formation in alkali - SF_6 collisions: dependence on collisional and vibrational energy*, Chemical Phys.

10, 235-259.

WILKINSON, J.H. [1963], *Rounding errors in algebraic processes*, Notes on applied science no. 32, Prentice Hall.

Bijlagen

```
"BEGIN" "COMMENT" PROGRAMMA VOOR POLYNOOM FITTING MET NORMAALVERG.
    EN CHOLESKY DECOMPOSITIE, VOORBEELDPROGRAMMA VOOR GEBRUIK IN HET
    KADER VAN HET COLLOQUIUM NUMERIEKE PROGRAMMATUUR, 760113, JBUS;

    "PROCEDURE" DUPVEC(L,U,S,A,B); "CODE" 31030;
    "PROCEDURE" DUPMAT(LR,UR,LC,UC,A,B); "CODE" 31035;
    "PROCEDURE" DUPCOLVEC(L,U,J,A,B); "CODE" 31034;
    "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B); "CODE" 34014;
    "PROCEDURE" LNGTAMMAT(L,U,I,J,A,B,C,CC,D,DD); "CODE" 34414;
    "PROCEDURE" FULTAMVEC(LR,UR,LC,UC,A,B,C); "CODE" 31501;
    "PROCEDURE" LNGFULTAMVEC(LR,UR,LC,UC,A,B,C); "CODE" 31506;
    "PROCEDURE" CHLDECSOL2(A,N,AUX,B); "CODE" 34392;

    "INTEGER" N, M, M1;
    N:= 15; M:= 32; M1:= M + 1;
    "BEGIN"
        "REAL" "PROCEDURE" F(T); "VALUE" T; "REAL" T;
        "BEGIN" "REAL" LABDA, PROD; "INTEGER" I;
            "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
            LABDA:= (T * 4 + 1) * K * 200;
            PROD:= 1; "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            PROD:= PROD / SINH(HNU[I] / (LABDA * 2));
            F:= PROD / (LABDA * 2 ** N)
        "END" F;

        "REAL" "PROCEDURE" POL(NN, X, T); "VALUE" NN, T;
        "INTEGER" NN; "REAL" T; "ARRAY" X;
        "BEGIN" "INTEGER" I; "REAL" PROD;
            PROD:= X[NN + 1]; "FOR" I:= NN "STEP" -1 "UNTIL" 1 "DO"
            PROD:= PROD * T + X[I]; POL:= PROD
        "END" POL;

        "PROCEDURE" OUT(BG, BV, COEF); "INTEGER" BG; "REAL" BV;
        "ARRAY" COEF;
        "BEGIN" "INTEGER" I;
            OUTPUT(71, "("//,"("DE BESTE GRAAD IS ")" 2ZDBB,
            "("HET VARIANTIE ")",B,4D"+3DB,//")", BG, BV);
            OUTPUT(71, "("("("DE COEFFICIENTEN ZIJN ")",/")");
            "FOR" I:= 0 "STEP" 1 "UNTIL" BG "DO"
            OUTPUT(71, "("2ZDB,B+,14D"+3DB,/")", I, COEF[I+1,BG])
        "END" OUT;

        "INTEGER" NN, I, J, BESTEGRAAD1, BESTEGRAAD2;
        "BOOLEAN" SNG, SNGD;
        "REAL" K, NRM, BESTEVAR1, BESTEVAR2;
        "ARRAY" HNU[1:N], SIGMA1, SIGMA2[1:M-1], FF[1:M1], AUX[2:3],
        COEF1, COEF2[1:M,1:M-1], A[1:M1,1:M], ATA, ATAD[1:M,1:M],
        ATF, ATFD[1:M];

        K:= 8,617065"-5; SNG:= SNGD:= "FALSE";
        OUTPUT(71, "("/,"("CHOLESKY DECOMPOSITIE")",/")");
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        INPUT(70, "("/,+,5D"+3D")", HNU[I]);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO" FF[I + 1]:= F(I/M);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 0 "STEP" 1 "UNTIL" M - 1 "DO"
        A[I + 1, J + 1]:= "IF" J = 0 "THEN" 1 "ELSE" "IF" I = 0 "THEN"
        0 "ELSE" (I/M) ** J;
```

```
        NRM:= 0; "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 1 "STEP" 1 "UNTIL" I "DO"
        "BEGIN" "REAL" D, DD, AAI, AADI;
            ATA[I, J]:= ATA[J, I]:= AAI:= TAMMAT(1, M1, I, J, A, A);
            LNGTAMMAT(1, M1, I, J, A, A, 0, 0, D, DD);
            ATAD[I, J]:= ATAD[J, I]:= AADI:= D + DD;
            NRM:= NRM + (AAI - AADI) ** 2 * 2
        "END" BEREKENING VAN NORMAALMATRIX;
        OUTPUT(71,"("//,"("DISKREPANTIE TUSSEN NORMAALMATR. BEREKEND")"
        /,"(" MET DUBBELE LENGTE EN ENKELE LENGTE IS: ")"B,2D"+2D,/")",
        SQRT(NRM));

        FULTAMVEC(1, M1, 1, M, A, FF, ATF);
        LNGFULTAMVEC(1, M1, 1, N, A, FF, ATFD);
        AUX[2]:= "-14; BESTEGRAAD1:= BESTEGRAAD2:= 1;
        BESTEVAR1:= BESTEVAR2:= "100;
        OUTPUT(71,"("//,"("GRAAD  VARIANTIE1  VARIANTIE2")",//")");
        "FOR" NN:= 1 "STEP" 1 "UNTIL" M - 1 "DO"
        "BEGIN" "INTEGER" N1;
            "BOOLEAN" "PROCEDURE" CHOLESKY(MAT, N, AUX, B, SIG, COEF,
            BG, BV);
            "VALUE" N; "INTEGER" N, BG; "REAL" BV;
            "ARRAY" MAT, AUX, B, SIG, COEF;
            "BEGIN" "INTEGER" I; "REAL" DEL, SG; "BOOLEAN" OK;
                "ARRAY" MAT1[1:N,1:N], BB[1:N];
                DUPMAT(1, N, 1, N, MAT1, MAT);
                DUPVEC(1, N, 0, BB, B);
                CHLDECSOL2(MAT1, N, AUX, BB);
                CHOLESKY:= OK:= AUX[3] = N; "IF" OK "THEN"
                "BEGIN" DUPCOLVEC(1, N, NN, COEF, BB); DEL:= 0;
                    "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
                    DEL:= DEL + (POL(NN, BB, I/M) - FF[I + 1]) ** 2;
                    SG:= SIG[NN]:= DEL / (M - NN);
                    OUTPUT(71, "("B,4D"+3DB")", SG);
                    "IF" SG < BV "AND" BV > "-8 "THEN"
                    "BEGIN" BG:= NN; BV:= SG "END"
                "END" "ELSE"
                OUTPUT(71, "("6B,"("*")",5B")")
            "END" CHOLESKY;
            N1:= NN + 1; OUTPUT(71, "("BBZDBB")", NN);
            "IF" ¬SNG "THEN"
            SNG:= ¬CHOLESKY(ATA, N1, AUX, ATF, SIGMA1, COEF1,
            BESTEGRAAD1, BESTEVAR1);
            "IF" ¬SNGD "THEN"
            SNGD:= ¬CHOLESKY(ATAD, N1, AUX, ATFD, SIGMA2, COEF2,
            BESTEGRAAD2, BESTEVAR2);
            OUTPUT(71,"("/")");
            "IF" SNG "AND".SNGD "THEN" "GOTO" EXIT
        "END" LOOP;
    EXIT:  OUTPUT(71, "("/,"("DE RESULTATEN IN ENKELE LENGTE ZIJN")",
        //")"); OUT(BESTEGRAAD1, BESTEVAR1, COEF1);
        OUTPUT(71, "("/,"("DE RESULTATEN IN DUBBELE LENGTE ZIJN")",
        //")"); OUT(BESTEGRAAD2, BESTEVAR2, COEF1)
    "END"
"END"
        "EOP"
```

CHOLESKY DECOMPOSITIE

DISKREPANTIE TUSSEN NORMAALMATR. BEREKEND
MET DUBBELE LENGTE EN ENKELE LENGTE IS: .15"-12

GRAAD	VARIANTIE1	VARIANTIE2
1	.1899"+003	.1899"+003
2	.8737"+002	.8737"+002
3	.2875"+002	.2875"+002
4	.6672"+001	.6672"+001
5	.1072"+001	.1072"+001
6	.1163"+000	.1163"+000
7	.8234"-002	.8234"-002
8	.3642"-003	.3642"-003
9	.9475"-005	.9475"-005
10	.1565"-006	.1348"-006
11	.1395"-007	.1989"-007
12	.8382"-009	.2650"-007
13	*	*

DE RESULTATEN IN ENKELE LENGTE ZIJN

DE BESTE GRAAD IS 12 MET VARIANTIE .8382"-009

DE COEFFICIENTEN ZIJN

0	-.20325621710991"-004
1	+.10110341786336"-001
2	-.37777232008877"+000
3	+.53233122108854"+001
4	-.37871244083677"+002
5	+.15177989467513"+003
6	-.34709818731717"+003
7	+.58493646623176"+003
8	+.79297719686768"+002
9	-.86429265074692"+003
10	+.12160326922579"+004
11	-.80719541478130"+003
12	+.29741176470588"+003

DE RESULTATEN IN DUBBELE LENGTE ZIJN

DE BESTE GRAAD IS 11 MET VARIANTIE .1989"-007

DE COEFFICIENTEN ZIJN

0	+.64687239912617"-004
1	-.298918356652 51"-001
2	+.10716378772455"+001
3	-.14329765518802"+002
4	+.95149868189160"+002
5	-.34676099837787"+003
6	+.68413237449675"+003
7	-.53114320944882"+003
8	-.55052137711083"+003
9	+.16897620680760"+004
10	-.15871423872212"+004
11	+.63776820512820"+003

```
"BEGIN" "COMMENT" PROGRAMMA VOOR POLYNOOM FITTING MET NORMAALVERG.
    EN BEREKENING VAN DE FOUT, VOORBEELDPROGRAMMA VOOR GEBRUIK IN HET
    KADER VAN HET COLLOQUIUM NUMERIEKE PROGRAMMATUUR, 760113, JBUS;

    "PROCEDURE" DUPVEC(L,U,S,A,B); "CODE" 31030;
    "PROCEDURE" DUPMAT(LR,UR,LC,UC,A,B); "CODE" 31035;
    "PROCEDURE" DUPCOLVEC(L,U,J,A,B); "CODE" 31034;
    "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B); "CODE" 34014;
    "PROCEDURE" FULTAMVEC(LR,UR,LC,UC,A,B,C); "CODE" 31501;
     "PROCEDURE" GSSERB(A,N,AUX,RI,CI); "CODE" 34242;
    "PROCEDURE" SOLELM(A, N, RI, CI, B); "CODE" 34061;
    "PROCEDURE" ITISOLERB(A,LU,N,AUX,RI,CI,B); "CODE" 34253;

    "INTEGER" N, M, M1;
    N:= 15; M:= 32; M1:= M + 1;
    "BEGIN"
        "REAL" "PROCEDURE" F(T); "VALUE" T; "REAL" T;
        "BEGIN" "REAL" LABDA, PROD; "INTEGER" I;
            "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
            LABDA:= (T * 4 + 1) * K * 200;
            PROD:= 1; "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            PROD:= PROD / SINH(HNU[I] / (LABDA * 2));
            F:= PROD / (LABDA * 2 ** N)
        "END" F;

        "REAL" "PROCEDURE" POL(NN, X, T); "VALUE" NN, T;
        "INTEGER" NN; "REAL" T; "ARRAY" X;
        "BEGIN" "INTEGER" I; "REAL" PROD;
            PROD:= X[NN + 1]; "FOR" I:= NN "STEP" -1 "UNTIL" 1 "DO"
            PROD:= PROD * T + X[I]; POL:= PROD
        "END" POL;

        "INTEGER" NN, I, J, BESTEGRAAD;
        "REAL" K, BESTEVAR;
        "ARRAY" HNU[1:N], SIGMA[1:M-1], FF[1:M1], AUX[0:13],
        COEF[1:M,1:M-1], A[1:M1,1:M], ATA[1:M,1:M], ATF[1:M];

        K:= 8.617065"-5;
        OUTPUT(71, "("/,"("GAUSSELIMINATIE MET FOUTBEREKENING")",/")");
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        INPUT(70, "("/,+.5D"+3D")", HNU[I]);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO" FF[I + 1]:= F(I/M);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 0 "STEP" 1 "UNTIL" M - 1 "DO"
        A[I + 1, J + 1]:= "IF" J = 0 "THEN" 1 "ELSE" "IF" I = 0 "THEN"
        0 "ELSE" (I/M) ** J;
        "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 1 "STEP" 1 "UNTIL" I "DO"
        ATA[I, J]:= ATA[J, I]:= TAMMAT(1, M1, I, J, A, A);
```

```
        FULTAMVEC(1, M1, 1, M, A, FF, ATF);
        AUX[0]:=AUX[2]:= "-14; AUX[4]:= 8; AUX[6]:= "-13; AUX[8]:= "-4;
        AUX[10]:= "-4; AUX[12]:= 10; BESTEGRAAD:= 1; BESTEVAR:= "100;
        OUTPUT(71,"("//,"("GRAAD  VARIANTIE  FOUT")",//")");
        "FOR" NN:= 1 "STEP" 1 "UNTIL" M - 1 "DO"
        "BEGIN" "INTEGER" I, N1;
            "ARRAY" MAT[1:NN+1,1:NN+1], BB[1:NN+1];
            "INTEGER" "ARRAY" RI, CI[1:NN+1];
            N1:= NN + 1; OUTPUT(71, "("2BZD2B")", NN);
            DUPMAT(1, N1, 1, N1, MAT, ATA);
            DUPVEC(1, N1, 0, BB, ATF);
            GSSERB(MAT, N1, AUX, RI, CI);
            "IF" AUX[3] = N1 "THEN"
            "BEGIN" "INTEGER" I; "REAL" DEL; "BOOLEAN" BOOL;
                BOOL:= "FALSE"; "IF" ABS(AUX[11]) < "-3 "THEN"
                SOLELM(MAT, N1, RI, CI, BB) "ELSE"
                "BEGIN" ITISOLERB(ATA, MAT, N1, AUX, RI, CI, BB);
                    BOOL:= "TRUE"
                "END";
                DUPCOLVEC(1, N1, NN, COEF, BB); DEL:= 0;
                "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
                DEL:= DEL + (POL(NN, BB, I/M) - FF[I + 1]) ** 2;
                SIGMA[NN]:= DEL / (M - NN);
                OUTPUT(71, "("2(B-,4D"+3DB)")", SIGMA[NN], AUX[11]);
                "IF" SIGMA[NN] < BESTEVAR "AND" BESTEVAR > "-8 "THEN"
                "BEGIN" BESTEGRAAD:= NN; BESTEVAR:= SIGMA[NN] "END";
                "IF" ABS(AUX[11])>"-3 "THEN" OUTPUT(71,"(""("FOUT")"")")
                "ELSE" "IF" BOOL "THEN" OUTPUT(71,"(""("VERBETERD")"")"
                ); OUTPUT(71,"("/")")
            "END" "ELSE"
            "BEGIN" OUTPUT(71, "(""("SINGULAR")",/")");
                "GOTO" EXIT
            "END"
        "END" LOOP;
    EXIT: OUTPUT(71, "("//,"("DE BESTE GRAAD IS ")",2ZDBB,
        "("MET VARIANTIE ")",B,4D"+3DB,//")", BESTEGRAAD, BESTEVAR);
        OUTPUT(71,"(""("DE COEFFICIENTEN ZIJN ")", /")");
        "FOR" I:= 0 "STEP" 1 "UNTIL" BESTEGRAAD "DO"
        OUTPUT(71, "("2ZD8,B+,14D"+3DB,/")", I, COEF[I+1,BESTEGRAAD])
    "END"
"END"
        "EOP"
```

```
GAUSSELIMINATIE MET FOUTBEREKENING

GRAAD  VARIANTIE  FOUT

   1     ,1899"+003    .8159"-011
   2     ,8737"+002    .4071"-009
   3     ,2875"+002    .2447"-007
   4     ,6672"+001    .1196"-005
   5     ,1072"+001    .5146"-004
   6     .1163"+000   -.1000"+001 FOUT
   7     ,8234"-002   -.1000"+001 FOUT
   8     ,3642"-003   -.1000"+001 FOUT
   9     ,9475"-005   -.1000"+001 FOUT
  10     .1334"-006   -.1000"+Q01 FOUT
  11  SINGULAR

DE BESTE GRAAD IS  10  MET VARIANTIE  .1334"-006

DE COEFFICIENTEN ZIJN
   0  +.18993822446328"-003
   1  -.10815949482808"+000
   2  +.42327139662884"+001
   3  -.62622320156673"+002
   4  +.47430079726619"+003
   5  -.20921499171709"+004
   6  +.57153288658972"+004
   7  -.98766178139398"+004
   8  +.10629019744653"+005
   9  -.66298688492283"+004
  10  +.19164412234273"+004
```

```
"BEGIN" "COMMENT" PROGRAMMA VOOR POLYNOOM FITTING MET
    HOUSEHOLDER ORTHOGONALISATIE VOORBEELDPROGRAMMA IN HET
    KADER VAN HET COLLOQUIUM NUMERIEKE PROGRAMMATUUR,
    760116, JBUS;

    "PROCEDURE" DUPVEC(L,U,S,A,B); "CODE" 31030;
    "PROCEDURE" DUPMAT(LR,UR,LC,UC,A,B); "CODE" 31035;
    "PROCEDURE" DUPCOLVEC(L,U,I,A,B); "CODE" 31034;
    "PROCEDURE" LSQORTDECSOL(A,N,M,AUX,DIAG,B); "CODE" 34135;

    "INTEGER" N, M, M1;
    N:= 15; M:= 32; M1:= M + 1;
    "BEGIN"
        "REAL" "PROCEDURE" F(T); "VALUE" T; "REAL" T;
        "BEGIN" "REAL" LABDA, PROD; "INTEGER" I;
            "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
            LABDA:= (T * 4 + 1) * K * 200;
            PROD:= 1; "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            PROD:= PROD / SINH(HNU[I] / (LABDA * 2));
            F:= PROD / (LABDA * 2 ** N)
        "END" F;

        "REAL" "PROCEDURE" POL(NN, X, T); "VALUE" NN, T;
        "INTEGER" NN; "REAL" T; "ARRAY" X;
        "BEGIN" "INTEGER" I; "REAL" PROD;
            PROD:= X[NN + 1]; "FOR" I:= NN "STEP" -1 "UNTIL" 1 "DO"
            PROD:= PROD * T + X[I]; POL:= PROD
        "END" POL;

        "INTEGER" NN, I, J, BESTEGRAAD; "REAL" K, DEL, BESTEVAR;
        "ARRAY" HNU[1:N], SIGMA[1:M-1], FF[1:M1], AUX[2:5],
        COEF[1:M-1,1:M-1], A[1:M1,1:M];

        K:= 8,617065"-5;
        OUTPUT(71, "("/,"("HOUSEHOLDER ORTHOGONALISATIE")",/")");
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        INPUT(70, "("/,+.5D"+3D")", HNU[I]);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO" FF[I + 1]:= F(I/M);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 0 "STEP" 1 "UNTIL" M - 1 "DO"
        A[I + 1, J + 1]:= "IF" J = 0 "THEN" 1 "ELSE" "IF" I = 0 "THEN"
        0 "ELSE" (I/M) ** J;
        AUX[2]:= "-14; BESTEGRAAD:= 1; BESTEVAR:= "100;
        OUTPUT(71,"("//,"("GRAAD  VARIANTIE ")",//")");
```

```
        "FOR" NN:= 1 "STEP" 1 "UNTIL" M - 1 "DO"
        "BEGIN" "INTEGER" N1; "REAL" SIG;
            "ARRAY" MAT[1:M1,1:NN + 1], B[1:M1], DIAG[1:NN+1];
            N1:= NN + 1; OUTPUT(71, "("B8ZDBB")", NN);
            DUPMAT(1, M1, 1, N1, MAT, A);
            DUPVEC(1, M1, 0, B, FF);
            LSQORTDECSOL(MAT, M1, N1, AUX, DIAG, B);
            "IF" AUX[3] = N1 "THEN"
            "BEGIN" DUPCOLVEC(1, N1, NN, COEF, B); DEL:= 0;
                "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
                DEL:= DEL + (POL(NN, B, I/M) - FF[I + 1]) ** 2;
                SIG:= SIGMA[NN]:= DEL / (M - NN);
                OUTPUT(71, "("B,4D"+3DB,/")", SIG);
                "IF" SIG < BESTEVAR "AND" ~(BESTEVAR < "-8)" THEN"
                "BEGIN" BESTEGRAAD:= NN; BESTEVAR:= SIG "END"
            "END" "ELSE"
            "BEGIN" OUTPUT(71, "(""(" SINGULAR ")",/")");
                "GOTO" EXIT
            "END"
        "END" LOOP;
    EXIT: OUTPUT(71, "("//,"("DE BESTE GRAAD IS ")",2ZDBB,
        "("MET VARIANTIE ")", B,4D"+3DB,//")", BESTEGRAAD, BESTEVAR);
        OUTPUT(71, "(""("DE COEFFICIENTEN ZIJN ")",/")");
        "FOR" I:= 0 "STEP" 1 "UNTIL" BESTEGRAAD "DO"
        OUTPUT(71, "("2ZDB,B+,14D"+3DB,/")", I, COEF[I+1,BESTEGRAAD])
    "END"
"END"
        "EOP"
```

HOUSEHOLDER ORTHOGONALISATIE

GRAAD	VARIANTIE
1	.1899"+003
2	.8737"+002
3	.2875"+002
4	.6672"+001
5	.1072"+001
6	.1163"+000
7	.8234"-002
8	.3642"-003
9	.9475"-005
10	.1331"-006
11	.8848"-009
12	.2195"-011
13	.1147"-014
14	.3760"-022
15	.2106"-022
16	.5524"-023
17	.1150"-023
18	.2109"-024
19	.2143"-024
20	.1540"-024
21	SINGULAR

DE BESTE GRAAD IS 11 MET VARIANTIE .8848"-009

DE COEFFICIENTEN ZIJN

0	-.10155109781812"-004
1	+.10936783418512"-001
2	-.51852659376703"+000
3	+.92733459962192"+001
4	-.85687861503618"+002
5	+.46830052193930"+003
6	-.16222954168652"+004
7	+.36837434798592"+004
8	-.55200421343648"+004
9	+.53403948773815"+004
10	-.31067080678311"+004
11	+.91148550670628"+003

```
"BEGIN" "COMMENT" PROGRAMMA VUOR POLYNOOM FITTING MET
    SINGULIERE WAARDEN DEKOMPOSITIE, VOORBEELDPROGRAMMA IN HET
    KADER VAN HET COLLOQUIUM NUMERIEKE PROGRAMMATUUR,
    760116, JBUS;

    "PROCEDURE" DUPVEC(L,U,S,A,B); "CODE" 31030;
    "PROCEDURE" DUPMAT(LR,UR,LC,UC,A,B); "CODE" 31035;
    "PROCEDURE" DUPCOLVEC(L,U,I,A,B); "CODE" 31034;
    "INTEGER" "PROCEDURE" SOLOVR(A,M,N,X,EM); "CODE" 34281;

    "INTEGER" N, M, M1;
    N:= 15; M:= 32; M1:= M + 1;
    "BEGIN"
        "REAL" "PROCEDURE" F(T); "VALUE" T; "REAL" T;
        "BEGIN" "REAL" LABDA, PROD; "INTEGER" I;
            "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
            LABDA:= (T * 4 + 1) * K * 200;
            PROD:= 1; "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            PROD:= PROD / SINH(HNU[I] / (LABDA * 2));
            F:= PROD / (LABDA * 2 ** N)
        "END" F;

        "REAL" "PROCEDURE" POL(NN, X, T); "VALUE" NN, T;
        "INTEGER" NN; "REAL" T; "ARRAY" X;
        "BEGIN" "INTEGER" I; "REAL" PROD;
            PROD:= X[NN + 1]; "FOR" I:= NN "STEP" -1 "UNTIL" 1 "DO"
            PROD:= PROD * T + X[I]; POL:= PROD
        "END" POL;

        "INTEGER" NN, I, J, BESTEGRAAD; "REAL" K, DEL, BESTEVAR;
        "ARRAY" HNU[1:N], SIGMA[1:M-1], FF[1:M1], AUX[0:7],
        COEF[1:M,1:M-1], A[1:M1,1:M];

        K:= 8,617065"-5;
        OUTPUT(71, "("/,"("SINGULIERE WAARDEN ONTBINDING")",/")");
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        INPUT(70, "("/,+,5D"+3D")", HNU[I]);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO" FF[I + 1]:= F(I/M);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 0 "STEP" 1 "UNTIL" M - 1 "DO"
        A[I + 1, J + 1]:= "IF" J = 0 "THEN" 1 "ELSE" "IF" I = 0 "THEN"
        0 "ELSE" (I/M) ** J;
        BESTEGRAAD:= 1; BESTEVAR:= "100;
        AUX[0]:= "-14; AUX[2]:= "-12; AUX[4]:= 10 * M; AUX[6]:= "-10;
        OUTPUT(71,"("//,"("GRAAD  VARIANTIE ")",//")");
```

```
        "FOR" NN:= 1 "STEP" 1 "UNTIL" M - 1 "DO"
        "BEGIN" "INTEGER" N1; "REAL" SIG;
            "ARRAY" MAT[1:M1,1:NN + 1], B[1:M1];
            N1:= NN + 1; OUTPUT(71, "("BBZDBB")", NN);
            DUPMAT(1, M1, 1, N1, MAT, A);
            DUPVEC(1, M1, 0, B, FF);
            "IF" SOLOVR(MAT, M1, N1, B, AUX) = 0 "THEN"
            "BEGIN" DUPCOLVEC(1, N1, NN, COEF, B); DEL:= 0;
                "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
                DEL:= DEL + (POL(NN, B, I/M) - FF[I + 1]) ** 2;
                SIG:= SIGMA[NN]:= DEL / (M - NN);
                OUTPUT(71, "("B,4D"+3DB,/")", SIG);
                "IF" SIG < BESTEVAR "AND" BESTEVAR > "-8 "THEN"
                "BEGIN" BESTEGRAAD:= NN; BESTEVAR:= SIG "END"
            "END" "ELSE"
            "BEGIN" OUTPUT(71, "(""(" FAILED ")",/")");
                "GOTO" EXIT
            "END"
        "END" LOOP;
    EXIT: OUTPUT(71, "("//,"("DE BESTE GRAAD IS ")",2ZDBB,
        "("MET VARIANTIE ")", B,4D"+3DB,//")", BESTEGRAAD, BESTEVAR);
        OUTPUT(71, "(""("DE COEFFICIENTEN ZIJN ")",/")");
        "FOR" I:= 0 "STEP" 1 "UNTIL" BESTEGRAAD "DO"
        OUTPUT(71, "("2ZDB,B+,14D"+3DB,/")", I, COEF[I+1,BESTEGRAAD])
    "END"
"END"
        "EOP"
```

SINGULIERE WAARDEN ONTBI'IDING

GRAAD	VARIANTIE
1	.1899"+003
2	.8737"+002
3	.2875"+002
4	.6672"+001
5	.1072"+001
6	.1163"+000
7	.8234"-002
8	.3642"-003
9	.9475"-005
10	.1331"-006
11	.1810"-006
12	.1414"-008
13	.1876"-010
14	.1290"-013
15	.2818"-013
16	.5267"-013
17	.5294"-011
18	.3054"-009
19	.6746"-012
20	.2923"-010
21	.1356"-010
22	.3835"-010
23	.1596"-009
24	.9124"-010
25	.4809"-010
26	.1403"-010
27	.2579"-010
28	.1389"-009
29	.1525"-009
30	.1382"-009
31	.9923"-009

DE BESTE GRAAD IS 12 MET VARIANTIE .1414"-008

DE COEFFICIENTEN ZIJN

0	+.27812738886800"-004
1	-.38333024389270"-002
2	+.31983173288792"-001
3	+.90863504776289"+000
4	-.16176007065588"+002
5	+.11009134964974"+003
6	-.40516073844303"+003
7	+.88019192945630"+003
8	-.11206886139563"+004
9	+.71403872026785"+003
10	+.84944449137310"+001
11	-.30102335993254"+003
12	+.20725213516816"+003

```
      SUBROUTINE DUPVEC(L, U, A, B, N)
C     COPIEERT DE VECTOR IN ARRAY B(L .. U) NAAR DE VECTOR IN
C     ARRAY A(L .. U)
      INTEGER L, U
      REAL A(N), B(N)

      DO 10 I = L, U
10        A(I) = B(I)
      RETURN
      END


      SUBROUTINE DUPCV(L, U, J, A, B, R, C)
C     COPIEERT DE VECTOR B(L .. U) NAAR DE KOLOM VECTOR IN A(L .. U, J)
      INTEGER L, U, J, R, C
      REAL A(R, C), B(R)

      DO 10 K = L, U
10        A(K, J) = B(K)
      RETURN
      END


      SUBROUTINE DUPMAT(LR, UR, LC, UC, A, B, R, C)
C     COPIEERT DE MATRIX IN ARRAY B(LR .. UR, LC .. UC) NAAR DE MATRIX
C     IN ARRAY A(LR .. UR, LC .. UC).
      INTEGER LR, UR, LC, UC, R, C
      REAL A(R, C), B(R, C)

      INTEGER ROW, COL

      DO 10 ROW = LR, UR
      DO 10 COL = LC, UC
10        A(ROW, COL) = B(ROW, COL)
      RETURN
      END


      REAL FUNCTION F(T)
C     DE TE BENADEREN FUNCTIE.

      COMMON M, N, K, HNU
      REAL K, HNU
      DIMENSION HNU(15)

      REAL LABDA, PROD

      REAL SINH
      SINH(X) = (EXP(X) + EXP(-X)) * TANH(X) / 2

      LABDA = 200 * K * (4 * T + 1)
      PROD = 1.0
      DO 10 I = 1, N
10        PROD = PROD / SINH( HNU(I) / (2 * LABDA) )
      F = PROD / ( 2 ** N * LABDA)
      RETURN
      END
```

```
      SUBROUTINE INIA(A, M, M1)
C     DE VANDERMONDE MATRIX WORDT IN ARRAY A OPGESLAGEN.
      REAL A(M1, M)

      MMIN1 = M - 1
      DO 10 I = 1, M1
          A(I, 1) = 1.0
          T = (I - 1) / FLOAT(M)
          DO 10 J = 1, MMIN1
              A(I, J + 1) = T ** J
10            CONTINUE
      RETURN
      END


      SUBROUTINE INIFF(FF, M1)
C     DE VECTOR VAN FUNCTIEWAARDEN IN DE DISCRETISERINGSPUNTEN WORDT IN
C     ARRAY FF OPGEBORGEN.
      REAL FF(M1)

      COMMON M, N, K, HNU(15)
      REAL K, HNU

      DO 10 I = 1, M1
10        FF(I) = F( (I - 1) / FLOAT(M) )
      RETURN
      END


      REAL FUNCTION POL(N, X, T, M)
C     BEREKENT VOOR ARGUMENT T HET POLYNOOM VAN DE GRAAD N MET
C     COEFFICIENTEN IN ARRAY X.
      REAL X(M)

      P = X(N + 1)
      DO 10 I = 1, N
10        P = P * T + X(N + 1 - I)
      POL = P
      RETURN
      END
```

```
      PROGRAM POLFIT1(INPUT,OUTPUT)
C     FIT EEN PLOYNOOM MET EEN KLEINSTE KWADRATEN OPLOSSING VAN
C     1.3.11 MET BEHULP VAN SINGULIERE WAARDEN ONTBINDING DOOR AANROEP
C     VAN LLSQAR UIT IMSL.
C     POLFIT1 ROEPT DIRECT OF INDIRECT DE VOLGENDE SUBROUTINES OF
C     FUNCTIONS AAN :
C     DUPVEC, DUPCV, DUPMAT, F, INIA, INIFF, POL (ZIE VOORAFGAANDE),
C     GRAAD1, LSTSQ1 (ZIE VERVOLG),
C     LLSQAR (UIT IMSL).
C     HET PROGRAMMA VERWACHT DE GETALLEN HNU ALS INVOER.

      COMMON M, N, K, HNU(15)

      REAL A(33, 32), COEF(33, 33), SIGMA(32), HNU, K, FF(33)
      INTEGER DEGREE, DEGREE1

      EXTERNAL GRAAD1
      INTEGER GRAAD1

      M = 32  $ M1 = M + 1    $ N = 15
      K = 8.617065E-5
      DO 10 I = 1, N
10        READ *, HNU(I)
      CALL INIA(A, M, M1)
      CALL INIFF(FF, M1)
      DEGREE = GRAAD1(A, FF, SIGMA, COEF, M, M1)
      DEGREE1 = DEGREE + 1
      PRINT 100
      PRINT 120, DEGREE
      PRINT 130, SIGMA(DEGREE)
      PRINT 140
      DO 20 I = 1, DEGREE1
20        PRINT 150, COEF(I, DEGREE)
100   FORMAT("1POLYNOOM-FITTING, KLEINSTE KWADRATENOPLOSSING,",
     C    "SINGULIERE WAARDEN ONTBINDING MET LLSQAR UIT IMSL,")
120   FORMAT("0GRAAD : ",I4, /)
130   FORMAT(" VARIANTIE :", E24.14,/)
140   FORMAT(" COEFFICIENTEN : "/)
150   FORMAT(E24.14)
      END
```

```
      SUBROUTINE LSTSQ1(MAT, N, B, SIG, COEF, M, M1)
C     BIJ GEGEVEN GRAAD WORDT EEN KLEINSTE KWADRATENOPLOSSING MET
C     LLSQAR BEREKEND, EN DE BIJBEHORENDE VARIANTIE.
      REAL MAT(M1, M), B(M1), SIG(M), COEF(M1, M1)

      REAL A(33,32), FF(33), WKAREA(1300)

      N1 = N + 1
      CALL DUPMAT(1, M1, 1, N1, A, MAT, M1, M)
      CALL DUPVEC(1, M1, FF, B, M1)
      CALL LLSQAR(A, FF, M1, N1, 1, M1, M1, 14, WKAREA, IER)
      CALL DUPCV(1, N1, N, COEF, FF, M1, M1)
      DEL = 0.0
      DO 10 I = 1, M1
          T = (I - 1) / FLOAT(M)
10        DEL = DEL + ( POL(N, FF, T, M) - B(I) ) ** 2
      SIG(N) = DEL / ( M - N )
      RETURN
      END


      INTEGER FUNCTION GRAAD1(MAT, B, SIG, COEF, M, M1)
C     BEPAALT DE BESTE GRAAD.
      REAL MAT(M1,M), B(M1), SIG(M), COEF(M1, M1)

      MMIN1 = M - 1
      DO 10 NN = 1, MMIN1
          CALL LSTSQ1(MAT, NN, B, SIG, COEF, M, M1)
          IF ( NN .EQ. 1 ) GOTO 10
          IF ( SIG(NN) .LT. SIG(NN - 1) ) GOTO 100
          GRAAD1 = NN - 1
          RETURN
100       IF ( SIG(NN) .GT. 1E-8) GOTO 10
          GRAAD1 = NN
          RETURN
10        CONTINUE
      GRAAD = MMIN1
      RETURN
      END
```

```
POLYNOOM-FITTING, KLEINSTE KWADRATENOPLOSSING,SINGULIERE WAARDEN
ONTBINDING MET LLSQAR UIT IMSL.

GRAAD :   11

VARIANTIE :      .88489571586134E-09

COEFFICIENTEN :

   -.10155166581505E-04
    .10936788593793E-01
   -.51852668796986E+00
    .92733465701235E+01
   -.85687862303937E+02
    .46830051658639E+03
   -.16222953894616E+04
    .36837434240242E+04
   -.55200420734602E+04
    .53403948433599E+04
   -.31067080594829E+04
    .91148550654053E+03


      PROGRAM POLFIT2(INPUT,OUTPUT)
C     FIT EEN POLYNOOM MET EEN KLEINSTE KWADRATENOPLOSSING VAN
C     1,3,11 MET BEHULP VAN F04AMF UIT NAG.
C     POLFIT2 ROEPT DIRECT OF INDIRECT DE VOLGENDE SUBROUTINES OF
C     FUNCTIONS AAN:
C     DUPCV, F, INIA, INIFF, POL (ZIE VOORAFGAANDE),
C     LSTSQ2, GRAAD2 (ZIE VERVOLG),
C     F04AMF (UIT NAG).
C     HET PROGRAMMA VERWACHT DE GETALLEN HNU ALS INVOER.
      COMMON M, N, K, HNU(15)

      REAL A(33, 32), COEF(33, 33), SIGMA(32), HNU, K, FF(33)
      INTEGER M, N, I, DEGREE, DEGREE1

      EXTERNAL GRAAD2
      INTEGER GRAAD2

      M = 32  $ M1 = M + 1    $ N = 15
      K = 8.617065E-5
      DO 10 I = 1, N
10        READ *, HNU(I)
      CALL INIA(A, M, M1)
      CALL INIFF(FF, M1)
      DEGREE = GRAAD2(A, FF, SIGMA, COEF, M, M1)
      DEGREE1 = DEGREE + 1
      PRINT 100
      PRINT 120, DEGREE
      PRINT 130, SIGMA(DEGREE)
      PRINT 140
      DO 20 I = 1, DEGREE1
20        PRINT 150, COEF(I, DEGREE)
100   FORMAT("1POLYNOOM-FITTING, KLEINSTE KWADRATENOPLOSSING,",
     C    " F04AMF UIT NAG.")
120   FORMAT("0GRAAD2 : ",I4, /)
130   FORMAT(" VARIANTIE :", E24.14,/)
140   FORMAT(" COEFFICIENTEN : "/)
150   FORMAT(E24.14)
      END
```

```
      LOGICAL FUNCTION LSTSQ2(MAT, N, B, SIG, COEF, M, M1)
      REAL MAT(M1, M), B(M), SIG(M), COEF(M1, M1)

      REAL A(33, 32), ALPHA(32), E(32), Y(32), Z(32), R(32), FF(33)
      INTEGER IPIV(32)

      ETA = 2.0 ** (-47)
      IP = IFAIL = 1     $     N1 = N + 1
      CALL F04AMF(MAT, M1, FF, M1, B, M1, M1, N1, IP, ETA, A, M1,
     C     ALPHA, E, Y, Z, R, IPIV, IFAIL)
      IF ( IFAIL .EQ. 1 ) GOTO 100
      IF ( IFAIL .EQ. 2 ) GOTO 200

C     IFAIL .EQ. 0 :
      CALL DUPCV(1, N1, N, COEF, FF, M1, M1)
      DEL = 0.0
      DO 10 I = 1, M1
          T = (I - 1) / FLOAT(M)
10        DEL = DEL + ( POL(N, FF, T, M) - B(I) ) ** 2
      SIG(N) = DEL / (M - N)
      LSTSQ2 = .TRUE.
      RETURN

C     IFAIL .EQ. 1 :
100   PRINT *, "SINGULIER VOOR GRAAD2 : ", N
      LSTSQ2 = .FALSE.
      RETURN

C     IFAIL .EQ. 2 :
200   PRINT *, "GEEN POGING OPLOSSING TE VINDEN, GRAAD :", N
      LSTSQ2 = .FALSE.
      RETURN
      END


      INTEGER FUNCTION GRAAD2(MAT, B, SIG, COEF, M, M1)
      REAL MAT(M1,M), B(M1), SIG(M), COEF(M1, M1)

      EXTERNAL LSTSQ2
      LOGICAL VAROK, LSTSQ2

      MMIN1 = M - 1
      DO 10 NN = 1, MMIN1
          IF ( LSTSQ2(MAT, NN, B, SIG, COEF, M, M1) ) GOTO 100
200       GRAAD2 = NN - 1
          RETURN
250       GRAAD2 = NN
          RETURN
100       IF ( NN .EQ. 1 ) GOTO 10
          IF ( SIG(NN) .GE. SIG(NN-1) ) GOTO 200
          IF ( SIG(NN) .LE. 1E-8 ) GOTO 250
10        CONTINUE
      GRAAD2 = MMIN1
      RETURN
      END
```

POLYNOOM-FITTING, KLEINSTE KWADRATENOPLOSSING, F04AMF UIT NAG,

GRAAD2 : 11

VARIANTIE : ,88478836046671E-09

COEFFICIENTEN :

-,10155114131025E-04
,10936779934852E-01
-,51852648136585E+00
,92733445522349E+01
-,85687851639230E+02
,46830048145521E+03
-,16222953109401E+04
,36837432988473E+04
-,55200419331245E+04
,53403947371247E+04
-,31067080121878E+04
,91148549712100E+03

```
      PROGRAM POLFIT3(INPUT,OUTPUT)
C     FIT EEN POLYNOOM DOOR DE NORMAALVERGELIJKING 1,3,8 MET
C     CHOLESKY'S METHODE OP TE LOSSEN MET BEHULP VAN LEQT2P UIT IMSL.
C     POLFIT3 ROEPT DIRECT OF INDIRECT DE VOLGENDE SUBROUTINES OF
C     FUNCTIONS AAN:
C     DUPCV, DUPVEC, INIA, INIFF, F, POL (ZIE VOORAFGAANDE),
C     GRAAD3, CHLSKY3, INIATF (ZIE VERVOLG),
C     LEQT2P, VTPROF (UIT IMSL).
C     HET PROGRAMMA VERWACHT DE GETALLEN HNU ALS INVOER.
      COMMON M, N, K, HNU(15)

      REAL A(33, 32), ATA(528), ATF(32), COEF(32, 32), SIGMA(32),
     C     HNU, K, FF(33)
      INTEGER DEGREE, DEGREE1, UB

      EXTERNAL GRAAD3
      INTEGER GRAAD3

      M = 32  $ M1 = M + 1    $ N = 15
      UB = M * (M + 1) / 2
      K = 8,617065E-5
      DO 10 I = 1, N
10        READ *, HNU(I)
      CALL INIA(A, M, M1)
      CALL INIFF(FF, M1)
      CALL VTPROF(A, M1, M , M1, ATA)
      CALL INIATF(A, FF, ATF, M, M1)
      DEGREE = GRAAD3(ATA, ATF, SIGMA, COEF, M, M1, FF, UB)
      DEGREE1 = DEGREE + 1
      PRINT 100
      PRINT 120, DEGREE
      PRINT 130, SIGMA(DEGREE)
      PRINT 140
      DO 20 I = 1, DEGREE1
20        PRINT 150, COEF(I, DEGREE)
100   FORMAT("1POLYNOOM-FITTING, NORMAALVERGELIJKING EN CHOLESKY",
     C    " ONTBINDING, LEQT2P UIT IMSL.", /)
120   FORMAT("0GRAAD3 : ",I4, /)
130   FORMAT(" VARIANTIE :", E24.14,/)
140   FORMAT(" COEFFICIENTEN : "/)
150   FORMAT(E24.14)
      END


      SUBROUTINE INIATF(A, FF, ATF, M, M1)
      REAL A(M1, M), FF(M1), ATF(M)

      DOUBLE PRECISION SUM

      DO 10 I = 1, M
          SUM = 0.0D1
          DO 20 J = 1, M1
20            SUM = SUM + DBLE( A(J,I) ) * DBLE( FF(J) )
10        ATF(I) = SUM
      RETURN
      END
```

```
      LOGICAL FUNCTION CHLSKY3(MAT, N, ATF, SIG, COEF, M, M1, FF, UB)
      INTEGER UB
      REAL MAT(UB), ATF(M), SIG(M), COEF(M, M), FF(M1)

      REAL BB(32), WKAREA(592)
      INTEGER IDGT, IER, I, N1
      LOGICAL OK

      N1 = N + 1
      CALL DUPVEC(1, N1, BB, ATF, M)
      CALL LEQT2P(MAT, 1, N1, M, BB, IDGT, D1, D2, WKAREA, IER)
      CHLSKY3 = OK = ( IER .NE. 129 ) .AND. ( IER .NE. 130 )
      IF ( .NOT. OK ) RETURN
      CALL DUPCV(1, N1, N, COEF, BB, M, M)
      DEL = 0.0
      DO 10 I = 1, M1
          T = (I - 1) / FLOAT(M)
10        DEL = DEL + ( POL( N, BB, T, M) - FF(I) ) ** 2
      SIG(N) = DEL / (M - N)
      RETURN
      END


      INTEGER FUNCTION GRAAD3(MAT, B, SIG, COEF, M, M1, FF, UB)
      INTEGER M, M1, UB
      REAL MAT(UB), B(M), SIG(M), COEF(M,M), FF(M1)

      EXTERNAL CHLSKY3
      LOGICAL CHLSKY3

      INTEGER NN, MMIN1
      LOGICAL VAROK

      MMIN1 = M - 1
      DO 10 NN = 1, MMIN1
          IF ( CHLSKY3(MAT, NN, B, SIG, COEF, M, M1, FF, UB) ) GOTO 100
200       GRAAD3 = NN - 1
          RETURN
250       GRAAD3 = NN
          RETURN
100       IF ( NN .EQ. 1 ) GOTO 10
          IF ( SIG(NN) .GE. SIG(NN-1) ) GOTO 200
          IF ( SIG(NN) .LE. 1E-8 ) GOTO 250
10        CONTINUE
      GRAAD3 = MMIN1
      RETURN
      END
```

POLYNOOM-FITTING, NORMAALVERGELLIJKING EN CHOLESKY ONTBINDING, LEQT2P
UIT IMSL.

GRAAD3 : 11

VARIANTIE : ,553655219999326E-08

COEFFICIENTEN :

,43820332073484E-04
-,15652772869211E-01
,48992303401633E+00
-,53987976113787E+01
,24898356789310E+02
-,23290338480592E+02
-,24760564350372E+03
,11973184247179E+04
-,26148523440469E+04
,32232572878527E+04
-,22317281120634E+04
,75488348328084E+03

```
      PROGRAM POLFIT4(INPUT,OUTPUT)
C     FIT EEN POLYNOOM DOOR DE NORMAALVERGELIJKING 1.3.8 MET
C     CHOLESKY'S METHODE OP TE LOSSEN MET BEHULP VAN F04ASF UIT NAG.
C     POLFIT4 ROEPT DIRECT OF INDIRECT DE VOLGENDE SUBROUTINES OF
C     FUNCTIONS AAN:
C     DUPCV, INIA, INIFF, POL (ZIE VOORAFGAANDE),
C     CHLSKY4, GRAAD4, INIATAF (ZIE VERVOLG),
C     F01CJF, F01CKF, F04ASF (UIT NAG).
C     HET PROGRAMMA VERWACHT DE GETALLEN HNU ALS INVOER.
      COMMON M, N, K, HNU(15)

      REAL A(33, 32), ATA(32, 32), ATF(32), COEF(32, 32), SIGMA(32),
     C     HNU, K, FF(33)
      INTEGER DEGREE, DEGREE1

      EXTERNAL GRAAD4
      INTEGER GRAAD4

      M = 32  $ M1 = M + 1   $ N = 15
      K = 8.617065E-5
      DO 10 I = 1, N
10        READ *, HNU(I)
      CALL INIA(A, M, M1)
      CALL INIFF(FF, M1)
      CALL INIATAF(A, ATA, ATF, FF, M, M1)
      DEGREE = GRAAD4(ATA, ATF, SIGMA, COEF, M, M1, FF)
      DEGREE1 = DEGREE + 1
      PRINT 100
      PRINT 120, DEGREE
      PRINT 130, SIGMA(DEGREE)
      PRINT 140
      DO 20 I = 1, DEGREE1
20        PRINT 150, COEF(I, DEGREE)
100   FORMAT("1POLYNOOM-FITTING, NORMAALVERGELIJKING EN CHOLESKY",
     C    " ONTBINDING, F04ASF UIT NAG.")
120   FORMAT("0GRAAD= ",I4, /)
130   FORMAT(" VARIANTIE =", E24.14,/)
140   FORMAT(" COEFFICIENTEN : "/)
150   FORMAT(E24.14)
      END


      SUBROUTINE INIATAF(A, ATA, ATF, FF, M, M1)
      REAL A(M1, M), ATA(M, M), ATF(M), FF(M1)

      REAL AT(32, 33), Z(1)

      IFAIL = 0
      CALL F01CJF(AT, A, M, M1, 0, IFAIL)
      CALL F01CKF(ATA, AT, A, M, M, M1, Z, 1, 1, IFAIL)
      CALL F01CKF(ATF, AT, FF, M, 1, M1, Z, 1, 1, IFAIL)
      RETURN
      END
```

```
      LOGICAL FUNCTION CHLSKY4(MAT, N, B, SIG, COEF, M, M1, FF)
      REAL MAT(M, M), B(M), SIG(M), COEF(M, M), FF(M1)

      REAL SOL(32), WK1(32), WK2(32), DEL, T

      N1 = N + 1
      IFAIL = 1
      CALL F04ASF(MAT, M, B, N1, SOL, WK1, WK2, IFAIL)
      IF ( IFAIL - 1 ) 300, 200, 100

C     IFAIL .EQ. 2 :
100   PRINT *, 'MATRIX TE SLECHT GECONDITIONEERD. GRAAD :', N
      CHLSKY4 = .FALSE.
      RETURN

C     IFAIL .EQ. 1 :
200   PRINT *, 'MATRIX NIET POSITIEF DEFINIET. GRAAD : ', N
      CHLSKY4 = .FALSE.
      RETURN

C     IFAIL .EQ. 0: AANROEP WAS SUCCESVOL.
300   CHLSKY4 = .TRUE.
      CALL DUPCV(1, N1, N, COEF, SOL, M, M)
      DEL = 0.0
      DO 10 I = 1, M1
          T = (I - 1) / FLOAT(M)
10        DEL = DEL + ( POL( N, SOL, T, M) - FF(I) ) ** 2
      SIG(N) = DEL / (M - N)
      RETURN
      END


      INTEGER FUNCTION GRAAD4(MAT, B, SIG, COEF, M, M1, FF)
      INTEGER M, M1
      REAL MAT(M,M), B(M), SIG(M), COEF(M,M), FF(M1)

      EXTERNAL CHLSKY4
      LOGICAL CHLSKY4

      INTEGER NN, MMIN1

      MMIN1 = M - 1
      DO 10 NN = 1, MMIN1
          IF ( CHLSKY4(MAT, NN, B, SIG, COEF, M, M1, FF) ) GOTO 100
200       GRAAD4 = NN - 1
          RETURN
250       GRAAD4 = NN
          RETURN
100       IF ( NN .EQ. 1 ) GOTO 10
          IF ( SIG(NN) .GE. SIG(NN-1) ) GOTO 200
          IF ( SIG(NN) .LE. 1E-8 ) GOTO 250
10        CONTINUE
      GRAAD4 = MMIN1
      RETURN
      END
```

POLYNOOM-FITTING. NORMAALVERGELIJKING EN CHOLESKY ONTBINDING. F04ASF UIT NAG.

GRAAD4 : 11

VARIANTIE : .55365522163548E-08

COEFFICIENTEN :

 .43820332073484E-04
-.15652772869211E-01
 .48992303401633E+00
-.53987976113787E+01
 .24898356789310E+02
-.23290338480592E+02
-.24760564350372E+03
 .11973184247179E+04
-.26148523440469E+04
 .32232572878527E+04
-.22317281120634E+04
 .7548834832808 4E+03

1. LINEAIRE ALGEBRA

1.2. Het eigenwaardenprobleem en de singuliere waarden ontbinding

door D.T. Winter
(Mathematisch Centrum)

1.2.1. Inleiding

Evenals voor het onderwerp lineaire stelsels bestaan er ook voor het eigenwaardenprobleem en de singuliere waarden ontbinding goed geanalyseerde algoritmen. We zullen bekijken wat de programmatheken ACCULIB (ALGOL 60 en FORTRAN), IMSL (FORTRAN), NAG (ALGOL 60 en FORTRAN) en NUMAL (ALGOL 60) ons op dit gebied te bieden hebben. Tevens zullen we, om meer inzicht in de singuliere waarden ontbinding te verkrijgen, het probleem aangeroerd bij het onderwerp lineaire stelsels met deze ontbinding trachten op te lossen.

Ter verduidelijking is bij alle programmatuur uit ACCULIB met een A (ALGOL 60) of een F (FORTRAN) aangegeven voor welke taal de betreffende routine geschikt is. In de NAG programmatheek komt iedere routine twee maal voor, voor beide programmeertalen eenmaal, dit is aangegeven door een A of een F aan het eind van de routinenaam. Wij hebben deze routines overal samengenomen door de laatste letter te vervangen door A/F. Zo staat F02AXA/F voor de twee routines F02AXA (ALGOL 60) en F02AXF (FORTRAN).

1.2.2. Overzicht beschikbare programmatuur voor het eigenwaardenprobleem

We beschouwen het (gegeneraliseerde) eigenwaardenprobleem

$$Ax = \lambda Bx, \tag{1.2.2.1}$$

waarbij A en B gegeven n bij n matrices zijn, x een n-vector is (een eigenvector) en λ een scalar is (een eigenwaarde). Het is bekend dat vergelijking (1.2.2.1) n eigenwaarden heeft, die echter niet noodzakelijk verschillend zijn. Heeft deze vergelijking verder n lineair onafhankelijke eigenvectoren, dan noemen we de vergelijking niet-defect. In het defecte geval heeft de vergelijking dus minder dan n lineair onafhankelijke eigenvectoren en is in feite incompleet.

Het eigenwaardenprobleem is het probleem één of meerdere eigenwaarden en/of eigenvectoren van (1.2.2.1) te berekenen.

Aangezien er geen programmatuur beschikbaar is voor defecte vergelijkingen beperken we ons tot de niet-defecte vergelijkingen. In de volgende gevallen is van te voren bekend dat de vergelijking niet-defect is, in andere gevallen is dat niet of moeilijk te bepalen:

a) A reëel symmetrisch, B reëel symmetrisch en niet singulier;
b) A complex hermitisch, B complex hermitisch en niet singulier;

(B mag dus in beide gevallen de eenheidsmatrix zijn).

We gaan nu het probleem uitsplitsen in het gewone eigenwaardenprobleem, waarbij B de eenheidsmatrix is.

(1.2.2.2) $Ax = \lambda x$

en het gegeneraliseerde eigenwaardenprobleem, met B ongelijk aan de eenheidsmatrix.

1.2.2.1. Het gewone eigenwaardenprobleem

We hebben de beschikbare programmatuur ondergebracht in een aantal tabellen. Tabel 1 bevat een ruwe indeling van het type matrix (A) en verwijst naar tabellen 2 tot 5. We hebben speciale vormen van A (zoals tridiagonaalvorm etc.) weggelaten omdat de meeste routines er vanuit gaan dat die speciale vorm bereikt is met andere routines uit dezelfde programmatheek.

TABEL 1	
A reëel, symmetrisch	zie Tabel 2
A reëel, niet-symmetrisch	zie Tabel 3
A complex, hermitisch	zie Tabel 4
A complex, niet-hermitisch	zie Tabel 5

TABEL 2

Alleen eigenwaarden	ACCULIB	disktri + tql2(F)	1)
	IMSL	EIGRS	2)
	NAG	F02AAA/F	
	NUMAL	grivalsym1	
		grivalsym2	
Eigenwaarden + eigenvectoren	ACCULIB	jacob (A/F)	
		disktri + tql2 + treigv (F)	1)
		tred2 + tql2 (A/F)	
	IMSL	EIGRS	2)
	NAG	F02ABA/F	
	NUMAL	qrisym	
Beperkt aantal eigenwaarden	NUMAL	eigvalsym1	
		eigvalsym2	
Beperkt aantal eigenwaarden	NAG	F02AGA/F	
+ eigenvectoren	NUMAL	eigsym1	
		eigsym 2	

1) Speciaal geschikt voor zeer grote matrices.
2) Berekent naar wens de eigenvectoren.

TABEL 3

Alleen eigenwaarden	IMSL	EIGRF	1)
	NAG	F02AFA/F	
	NUMAL	reaeigval	2)
		comeigval	
Eigenwaarden + eigenvectoren	ACCULIB	eigen (A/F)	3)
	IMSL	EIGRF	1)
	NAG	F02AGA/F	
	NUMAL	reaeig1	2)
		reaeig3	2)
		comeig1	
Beperkt aantal eigenwaarden	NAG	F02AGA/F	
+ eigenvectoren			

1) Berekent naar wens de eigenvectoren.
2) Alleen te gebruiken als van te voren bekend is dat alle eigenwaarden reëel zijn.
3) Berekent ook de eigenrijen (d.i. de oplossing van $x^T A = \lambda x^T$).

TABEL 4

Alleen eigenwaarden	IMSL	EIGCH	1)
	NAG	F02AWA/F	
	NUMAL	qrivalhrm	
Eigenwaarden + eigenvectoren	IMSL	EIGCH	1)
	NAG	F02AXA/F	
	NUMAL	qrihrm	
Beperkt aantal eigenwaarden	NUMAL	eigvalhrm	
Beperkt aantal eigenwaarden + eigenvectoren	NUMAL	eighrm	

1) Berekent naar wens de eigenvectoren.

TABEL 5

Alleen eigenwaarden	IMSL	EIGCC	1)
	NAG	F02AJA/F	
	NUMAL	eigvalcom	
Eigenwaarden + eigenvectoren	ACCULIB	equil + nordig + renorm	
	IMSL	EIGCC	1)
	NAG	F02AKA/F	
	NUMAL	eigcom	
Beperkt aantal eigenwaarden + eigenvectoren	NAG	F02ALA/F	

1) Berekent naar wens de eigenvectoren.

1.2.2.2. Het gegeneraliseerde eigenwaardenprobleem

Er is geen programmatuur beschikbaar voor complexe matrices. Het reële geval kunnen we onderverdelen in:

1) A is symmetrisch, B positief definiet zie tabel 6;

2) Aan 1) is niet voldaan zie tabel 7.

In geval 2) moeten we er rekening mee houden dat eigenwaarden oneindig kunnen zijn, en zelfs ongedefinieerd (als Ax = Bx = 0).

TABEL 6

Alleen eigenwaarden	NAG	F02ADA/F
Eigenwaarden + eigenvectoren	NAG	F02AEA/F

TABEL 7

Alleen eigenwaarden	IMSL	EIGZF	1)
	NUMAL	qzival	
Eigenwaarden + eigenvectoren	IMSL	EIGZF	1)
	NUMAL	qzi	

1) Berekent naar wens de eigenvectoren.

1.2.2.3. Gebruik van eigenwaarden bij het berekenen van nulpunten van polynomen

Met behulp van de eigenwaardenroutines voor symmetrische matrices kan men de nulpunten van orthogonale polynomen berekenen. Hiervoor is nog slechts één routine voorgesteld (in NUMAL). Wel is er in NUMAL een routine (polzeros) die met behulp van de routines voor berekening van eigenwaarden de nulpunten van een reëel polynoom berekenen. De zo berekende nulpunten zijn niet erg nauwkeurig, ze kunnen echter goed gebruikt worden als beginschattingen voor andere routines.

1.2.3. Programmatuur voor de singuliere waarden ontbinding

De singuliere waarden ontbinding van een n bij m matrix A wordt gedefinieerd door

(1.2.3.1) $A = U\Sigma V^H$

met U en V unitaire matrices en Σ een diagonaal matrix. De elementen op de diagonaal van Σ dienen positief of 0 te zijn, en heten singuliere waarden.

We kunnen de singuliere waarden ontbinding, onder andere, in de volgende gevallen gebruiken:

a. Bepaling van de rang van A. De rang is gelijk aan het aantal singuliere waarden ongelijk aan 0. Als we numeriek te werk gaan zullen we natuurlijk singuliere waarden kleiner dan een bepaald (klein) getal als nul beschouwen.
b. Bepaling van de pseudo-inverse van A (voor definitie zie hoofdstuk 1.1.). De pseudo-inverse van A wordt bepaald door

(1.2.3.2) $A^{+} = U\Sigma^{+}V^{H}$

waarbij Σ^{+} ook een diagonaal matrix is, met nullen op de diagonaal waar Σ ook nullen heeft, en verder de inversen van de singuliere waarden (zie ook de opmerking bij a.).
c. Oplossing van een kleinste-kwadratenprobleem. Laat gegeven zijn het stelsel

(1.2.3.3) $Ax = b$

met de vraag x zodanig te bepalen dat de lengte van de vector $r = Ax - b$ minimaal is. In dit geval is

(1.2.3.4) $x = A^{+}b.$

een oplossing, en wel, indien er meerdere oplossingen zijn, de oplossing van minimale lengte.
d. Bepaling eigenwaarden en eigenvectoren van $A^{H}A$. De eigenwaarden zijn gelijk aan de kwadraten van de singuliere waarden van A, en de eigenvectoren staan in matrix V (want $A^{H}A = V\Sigma^{H}\Sigma V^{H}$). De singuliere waarden ontbinding is nauwkeuriger dan wanneer we eerst $A^{H}A$ berekenen en daarna hiervan het eigenwaardenprobleem oplossen.
e. Bepaling van de inverse van $A^{H}A$. De inverse is eenvoudig $V\Sigma^{-1}(\Sigma^{H})^{-1}V^{H}$. Ook hier is weer de singuliere waarden ontbinding stabieler.
f. In het algemeen de oplossing van stelsels $Ax = b$. Vaak is oplossing met behulp van de singuliere waarden ontbinding stabieler dan met de andere methoden (vgl. hoofdstuk 1.1.). Dit geldt vooral als A een slechte conditie heeft.

In de volgende tabel kunnen we aflezen welke routines voor de singuliere waarden ontbinding beschikbaar zijn (er zijn geen routines voor complexe matrices).

TABEL 8

Alleen singuliere waarden	NUMAL	qrisngval	
Volledige ontbinding	IMSL	LSVALR	
	NAG	F01BGA/F	1)
		F01BHA/F	
	NUMAL	qrisngvaldec	

1) Berekent niet de ontbinding $U\Sigma V^T$, maar Σ, V en bij een gegeven matrix B: U^TB, dit is een deel van de berekening van A^+B.

1.2.4. Gebruik van de singuliere waarden ontbinding

In het probleem, vermeld in hoofdstuk 1.1.3. komen we uiteindelijk tot de volgende wiskundige formulering:

Bepaal een polynoom $P_N(t) = x_0 + x_1t + \ldots + x_Nt^N$ van graad N zodanig dat

$$(1.1.3.6) \quad \phi_N(x_0,\ldots,x_N) = \sum_{i=0}^{m} (f(t_i) - P_N(t_i))^2$$

minimaal is, waarbij m, f en t_i gedefinieerd zijn als in hoofdstuk 1.1.3. Noteren we voor zekere $N \geq 1$.

$$F = (f(t_0), f(t_1),\ldots,f(t_m))^T$$

en

$$x = (x_0,\ldots,x_N)^T;$$

en verder nog de zg. Vandermonde matrix:

$$V = \begin{pmatrix} 1 & t_0 & \ldots & t_0^N \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ 1 & t_m & & t_m^N \end{pmatrix}$$

dan wordt dus gevraagd de lengte van de vector Vx-F te minimaliseren, en dat is precies het kleinste-kwadratenprobleem uit voorbeeld 1.2.3.c.

Verder willen we natuurlijk dat het polynoom $P_N(t)$ van zo laag mogelijke graad is. Om dit te verwezenlijken laten we N lopen vanaf 1 en bepalen de rang (numeriek) van matrix V. Aangezien het aantal rijen van V groter is dan het aantal kolommen (N+1) is de rang dus maximaal N + 1. Geldt nu vanaf zekere N_0 dat de rang niet maximaal is, dan had toevoeging van de (N_0+1)ste kolom geen zin, en kunnen we stoppen. Voor het uitgewerkte voorbeeld zie men het bijgevoegde programma.

Ter bepaling van de numerieke rang werd de routine qrisngvaldec uit de NUMAL programmatheek gebruikt; en om uiteindelijk de vector x te berekenen werd de routine solsvdovr uit NUMAL gebruikt.

1.2.5. Aanvullende notities

1.2.5.1. De "performance indicator" in de IMSL programmatheek

De "performance indicator" wordt gebruikt in verscheidene routines. Hij geeft aan hoe goed het proces verlopen is. Een waarde groter dan 100 betekent slecht verloop, waardeloze resultaten, een waarde kleiner dan 1 betekent een gunstig verloop. Dit laatste wil zeggen dat, gegeven een matrix A, een vector x en een scalar λ, er kleine verstoringen ΔA, Δx en $\Delta\lambda$ zijn opdat $(A+\Delta A)(x+\Delta x) = (\lambda+\Delta\lambda)(x+\Delta x)$. Dit betekent dat λ dicht bij een echte eigenwaarde van A ligt, echter geenszins dat x dichtbij een eigenvector van A ligt.

1.2.5.2. De routines reaeigval, reaeig1 en reaeig3 uit NUMAL

Deze routines werken alleen maar correct als alle eigenwaarden reëel zijn. Zijn niet alle eigenwaarden reëel dan leveren de routines de reële delen van de eigenwaarden af en eigenvectoren die nergens op slaan. Uiteraard kan men altijd comeigval en comeig1 gebruiken, en deze routines zijn vaak sneller, óók als alle eigenwaarden reëel zijn!

1.2.5.3. Structuur van de routines

Alle routines transformeren de invoermatrix (matrices) eerst naar een speciale vorm, waarbij eigenwaarden c.q. singuliere waarden behouden blijven. Deze speciale vorm kunnen we aflezen uit de volgende tabel:

TABEL 9

probleem	speciale vorm
$Ax = \lambda x$, A reëel symmetrisch of complex hermitisch	tridiagonaal reëel symmetrisch
$Ax = \lambda x$, andere gevallen	hessenberg
$Ax = \lambda Bx$	A→hessenberg, B→driehoeksvorm
$A = U\Sigma V^T$	bidiagonaal

Hierna wordt het iteratieve proces losgelaten op de speciale vorm.

Is de matrix al in speciale vorm, dan kunnen we de transformatiefase overslaan; dit is echter niet nodig omdat direct ontdekt wordt dat deze fase leeg is. Bovendien als we deze fase overslaan en we willen eigenvectoren of de volledige singuliere waarden ontbinding berekenen, dan dienen we speciale kunstgrepen toe te passen.

1.2.5.4. Berekenen van eigenrijen

Alleen de routine eigen uit ACCULIB berekent eigenrijen. Willen we deze berekenen met één van de andere routines, dan dienen we het getransponeerde probleem in te voeren, immers de eigen vectoren van A^T zijn de eigenrijen van A etc.

1.2.5.5. Singuliere waarden ontbinding van matrices met meer kolommen dan rijen

Alle routines voor de singuliere waarden verwachten dat het aantal rijen groter ,is dan of gelijk is aan het aantal kolommen. Is dit niet het geval, dan voeren we de getransponeerde matrix in.

1.2.5.6. Vierkante matrix U bij de singuliere waarden ontbinding

Soms bestaat er behoefte aan een vierkante matrix U terwijl de invoermatrix niet vierkant is. We lossen dit op door de invoermatrix met een aantal aanvullende nulkolommen vierkant te maken.

Literatuur

BOYLE, J.M., B.S. GARBOW, Y. IKEBE, V.C. KLEMA, C.B. MOLER & B.T. SMITH [1974], *Matrix Eigensystem Routines - EISPACK Guide*, Lecture Notes in Computer Science, Springer-Verlag, Berlin.

WILKINSON, J.H. [1965], *The algebraic eigenvalue problem*, Clarendon Press, Oxford.

WILKINSON, J.H. & C. REINSCH [1971], *Linear Algebra*, Handbook for automatic computation II, Springer-Verlag, Berlin.

Bijlagen

```
"BEGIN" "COMMENT" PROGRAMMA VOOR POLYNOOM FITTING MET SINGULIERE
    WAARDEN ONTBINDING, VOORBEELDPROGRAMMA VOOR GEBRUIK IN HET
    KADER VAN HET COLLOQUIUM NUMERIEKE PROGRAMMATUUR, DWINTER;

    "PROCEDURE" DUPVEC(L,U,SHIFT,A,B); "CODE" 31030;
    "PROCEDURE" DUPMAT(LR,UR,LC,UC,A,B); "CODE" 31035;
    "INTEGER" "PROCEDURE" ORISNGVALDEC(A,M,N,VAL,V,EM); "CODE" 34273;
    "PROCEDURE" SOLSVDOVR(U,VAL,V,M,N,X,EM); "CODE" 34280;

    "INTEGER" N, M, M1;
    N:= 15; M:= 32; M1:= 33;
    "BEGIN"
        "REAL" "PROCEDURE" F(T); "VALUE" T; "REAL" T;
        "BEGIN" "REAL" LABDA, PROD; "INTEGER" I;
            "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
            LABDA:= (T * 4 + 1) * K * 200;
            PROD:= 1; "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            PROD:= PROD / SINH(HNU[I] / (LABDA * 2));
            F:= PROD / (LABDA * 2 ** N)
        "END" F;

        "REAL" "PROCEDURE" POL(NN, X, T); "VALUE" NN, T;
        "INTEGER" NN; "REAL" T; "ARRAY" X;
        "BEGIN" "INTEGER" I; "REAL" PROD;
            PROD:= X[NN + 1]; "FOR" I:= NN "STEP" -1 "UNTIL" 1 "DO"
            PROD:= PROD * T + X[I]; POL:= PROD
        "END" POL;

        "PROCEDURE" OUT(BG, BV, COEF); "INTEGER" BG; "REAL" BV;
        "ARRAY" COEF;
        "BEGIN" "INTEGER" I;
            OUTPUT(61, "("//, "("DE BESTE GRAAD IS ")", 2ZD8B,
            "("MET VARIANTIE ")", B,4D"+3D, //")", BG, BV);
            OUTPUT(61, "(""("DE COEFFICIENTEN ZIJN")", /")");
            "FOR" I:= 0 "STEP" 1 "UNTIL" BG "DO"
            OUTPUT(61, "("2ZD2B, +.14D"+3D, /")", I, COEF[I + 1])
        "END" OUT;

        "INTEGER" NN, I, J, BESTEGRAAD;
        "REAL" NRM, K, MIN;
        "ARRAY" HNU[1:N], COEF, FF[1:M1], EM[0:7],
        VAL, VVAL[1:M], A, AA, UU[1:M1,1:M], V,VV[1:M,1:M];

        K:= 8,617065"-5;
        OUTPUT(61, "("/, "("INVOER VOOR HNU")", /")");
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" INPUT(60, "("/, +.5D"+3D")", HNU[I]);
            OUTPUT(61, "("/, +.5D"+3D")", HNU[I])
        "END";
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO" FF[I + 1]:= F(I / M);
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
        "FOR" J:= 0 "STEP" 1 "UNTIL" M - 1 "DO"
        A[I + 1,J + 1]:= "IF" J = 0 "THEN" 1 "ELSE" "IF" I = 0 "THEN"
        0 "ELSE" (I / M) ** J;
        EM[0]:= "-14; EM[2]:= "-12; EM[4]:= 100; EM[6]:= "-9;
        OUTPUT(61, "("//, "("GRAAD  RANG   MIN.SING.WAARDE")", //")");
```

```
        "FOR" NN:= 1 "STEP" 1 "UNTIL" M - 1 "DO"
        "BEGIN" DUPMAT(1, M1, 1, NN + 1, AA, A);
            I:= QRISNGVALDEC(AA, M1, NN + 1, VAL, V, EM);
            "IF" I =/= 0 "THEN"
            "BEGIN" OUTPUT(61, "("//,"("QRISNGVALDEC NIET GEREED")"")");
                "GOTO" EXIT1
            "END";
            MIN:= VAL[NN + 1];
            "FOR" I:= 1 "STEP" 1 "UNTIL" NN "DO"
            "IF" VAL[I] < MIN "THEN" MIN:= VAL[I];
            OUTPUT(61, "("2(3ZD3B), N, /")", NN, EM[7], MIN);
            "IF" EM[7] <= NN "THEN" "GOTO" EXIT;
            DUPMAT(1, M1, 1, NN + 1, UU, AA);
            DUPMAT(1, NN + 1, 1, NN + 1, VV, V);
            DUPVEC(1, NN + 1, 0, VVAL, VAL);
            BESTEGRAAD:= NN
        "END" LOOP;
    EXIT: DUPVEC(1, M1, 0, COEF, FF);
        SOLSVDOVR(UU, VVAL, VV, M1, BESTEGRAAD + 1, COEF, EM);
        NRM:= 0;
        "FOR" I:= 0 "STEP" 1 "UNTIL" M "DO"
        NRM:= NRM + (POL(BESTEGRAAD, COEF, I / M) - FF[I + 1]) ** 2;
        OUTPUT(61, "("/, "("DE RESULTATEN ZIJN")", //")");
        OUT(BESTEGRAAD, NRM / (M - BESTEGRAAD), COEF);
    EXIT1:
    "END"
"END"
```

```
INVOER VOOR HNU

+.95900"-001
+.79560"-001
+.79560"-001
+.11640"+000
+.11640"+000
+.11640"+000
+.76100"-001
+.76100"-001
+.76100"-001
+.65100"-001
+.65100"-001
+.65100"-001
+.43000"-001
+.43000"-001
+.43000"-001

GRAAD  RANG   MIN.SING.WAARDE

   1      2   +1.5175459824350"+000
   2      3   +3.1593009565087"-001
   3      4   +6.1617328496584"-002
   4      5   +1.1646779571785"-002
   5      6   +2.1575621543764"-003
   6      7   +3.9354133330154"-004
   7      8   +7.0821297825273"-005
   8      9   +1.2583790371458"-005
   9     10   +2.2077835158088"-006
  10     11   +3.8232548502205"-007
  11     12   +7.8775102798944"-009
  12     13   +5.6549357271770"-009
  13     13   +7.6550889206484"-011

DE RESULTATEN ZIJN


DE BESTE GRAAD IS  12  MET VARIANTIE  .1414"-008

DE COEFFICIENTEN ZIJN
  0  +.27812738886800"-004
  1  -.38333024389270"-002
  2  +.31983173288792"-001
  3  +.90863504778289"+000
  4  -.16176007065588"+002
  5  +.11009134964974"+003
  6  -.40516073844303"+003
  7  +.88019192945630"+003
  8  -.11206886139563"+004
  9  +.71403872026785"+003
 10  +.84944449137310"+001
 11  -.30102335993254"+003
 12  +.20725213816816"+003
```

2. BEGINWAARDE- en BEGIN-RANDWAARDEPROBLEMEN

2.1. Beginwaardeproblemen voor stelsels gewone differentiaalvergelijkingen

door P.A. Beentjes
(Mathematisch Centrum)

2.1.1. Inleiding; Inventarisatie programmatuur

Veel verschijnselen en problemen zijn wiskundig te beschrijven door middel van een stelsel gewone differentiaalvergelijkingen, waarvan de oplossing in een bepaald punt gegeven is

$$y' = f(x,y),$$

$$y(x_0) = y_0, \qquad f \text{ en } y \text{ mogelijk vectoren.}$$

Dikwijls zal men van deze beginwaardeproblemen de oplossing in één of meerdere punten willen weten. Als deze oplossing moeilijk of in het geheel niet analytisch te bepalen is (wat meestal het geval is), dan zal men zijn toevlucht moeten nemen tot numerieke methoden. In totaal zijn hiervoor in de te bespreken programmatheken (ACCULIB, IMSL, NAG en NUMAL) zo'n 30 procedures (subroutines) beschikbaar.

Een overzicht van deze programmatuur wordt gegeven in tabel 1. Voor eventuele verwijzingen zijn de procedures (met welke naam voor het gemak ook subroutines bedoeld kunnen zijn) van een nummer, met toevoeging A (ALGOL procedure) en/of F (FORTRAN subroutine), voorzien.

Na deze inleiding zullen allereerst de belangrijkste (stuur)parameters, die men bij een procedure voor het oplossen van beginwaardeproblemen kan tegenkomen, worden besproken. Vervolgens zullen, afhankelijk van probleemeigenschappen en specifieke wensen van de gebruiker, adviezen worden gegeven ten aanzien van de te gebruiken procedure(s).

Tenslotte zal, door middel van een tweetal voorbeelden, de praktijk van het oplossen, in de vorm van programma's, worden bekeken.

TABEL 1

Inventarisatie van de beschikbare programmatuur.

Programmatheek	Procedurenaam	Nummer	Algoritme
ACCULIB	DIFFSYS	1A	Bulirsch-Stoer
IMSL	DASCRU	2F	Merson
	DCSLDE	3F	
	DREBS	4F	Bulirsch-Stoer
	DVOGER	5F	Gear
NAG	D02A$^{AA(F)}_{BA(F)}$	6AF	Merson
	D02AEA(F)	7AF	Gear
	D02AHA(F)	8AF	Krogh
NUMAL	MULTISTEP	9A	Gear
	IMPEX	10A	
	LINIGER1VS	11A	Liniger-Willoughby
	RKE	12A	Runge-Kutta
	RK4NA	13A	Runge-Kutta
	RK5NA	14A	Runge-Kutta
	RK2N	15A	Runge-Kutta
	RK3N	16A	Runge-Kutta
	EXP.F.TAYLOR	17A	Taylor
	EFERK	18A	Runge-Kutta
	EFSIRK	19A	Runge-Kutta
	GMS	20A	Multistep
	DIFFSYS	21A	Bulirsch-Stoer
	MODIFIED TAYLOR	22A	Taylor
	EFRK	23A	Runge-Kutta
	LINIGER 2	24A	Liniger-Willoughby
	ARK	25A	Runge-Kutta

2.1.2. Belangrijke (stuur)parameters

Zoals bij de meeste procedures het geval is, onderscheiden we ook bij procedures voor beginwaardeproblemen input (i-), output (o-) en input/output (io-) parameters. Voor de naamgeving van de hieronder te bespreken parameters is getracht een zo goed mogelijke "doorsnee" benaming te nemen.

n - i-parameter.
Het aantal componenten van y. Sommige procedures hebben een analogon voor scalarvergelijkingen (1A, 12A - 16A).

x - io-parameter.
De onafhankelijk variabele waarin meestal de beginwaarde x_0 moet worden meegegeven.

b - i-parameter.
Eindpunt van het integratieproces. De beschrijvingen van 2F, 12A, 15A en 16A maken duidelijk dat óók $b < x_0$ mag zijn.
Bij sommige procedures (13A en 14A) mag b een oplossingsafhankelijke expressie zijn.
Als een gebruiker geïnteresseerd is in een rij waarden $\{y(x_i)\}_{i=1}^{N}$ dan moet hij in de regel (uitzonderingen: 5F, 9A, 10A) de door hem gebruikte procedure N-maal aanroepen.

y - io-parameter.
De afhankelijk variabele, meestal een array ter lengte n, waarin bij aanroep de beginwaarden gegeven moeten zijn.

der - Een procedure, door de gebruiker te schrijven, waarin de afgeleide wordt berekend.
Soms wordt deze procedure componentsgewijs aangeroepen (3F, 13A - 16A).

jac - Een procedure, wéér door de gebruiker te schrijven, waarin de jacobiaan - dat is de matrix met elementen $\partial f_i/\partial y_j$ - wordt berekend. In vele gevallen zal men dit liever numeriek willen doen. Als de hoofdprocedure hiervoor geen optie heeft, kan men in iedere programmatheek wel een geschikte hulpprocedure voor de jacobiaanberekening vinden.

rp,ap - i-parameters.
Relatieve en absolute precisie waarmee de *nauwkeurigheid van het integratieproces* wordt gestuurd. De verkregen *nauwkeurigheid in de oplossing* zal in de regel toenemen bij kleinere waarden van deze stuurparameters.
Dikwijls zijn rp en ap arrays.
De procedures 4F - 8AF willen van de gebruiker een indicatie omtrent de te voeren stuurprecisie (relatief, absoluut of beide).

e - o-parameter.
Geschatte fout (mogelijk componentsgewijs) in de oplossing. Over het algemeen zal e slechts de grootte-orde van de werkelijke fout benaderen. Men mag dan ook geen al te groot belang toekennen aan deze foutschatting.

h,hmin,hmax - io-parameters.
Respectievelijke begin-, minimale- en maximale stap voor het integratieproces. De waarde hmin zal in de regel slechts gebruikt worden als er zich moeilijkheden voordoen tijdens integratie; om deze mogelijke knelpunten het hoofd te bieden verdient het aanbeveling om hmin een veilige (= behoorlijk kleine) waarde te geven.
Voor integratie met een door de gebruiker voorgeschreven stap, kan men terecht bij de procedures 1A - 6AF, 23A en 24A.
De éénstapsprocedures 1A, 4F en 5F geven na iedere stap een suggestie voor de grootte van de volgende stap.
Ook bij veel andere dan de genoemde procedures kan men de stap, door middel van hmin = hmax, zelf regelen.

sigma - i-parameter.
Een schatting van de spectraalradius of dominante eigenwaarde(n) van de jacobiaan. Deze parameter komt alleen voor in de specialistische NUMAL procedures 11A, 17A - 20A en 22A - 25A.

out - o-procedure.
Bij sommige procedures zijn de resultaten hierdoor na iedere integratiestap beschikbaar.

foutmeldingen - De meeste procedures zijn beperkt in hun mogelijkheden tot het signaleren van fouten. Het afbreken van het integratieproces door het gedwongen gebruik van een te kleine integratiestaplengte (mogelijke oorzaak: hmin te groot) geeft de meest voorkomende foutmelding.

2.1.3. Aanbevolen procedures

Tabel 2 geeft via een boomstructuur de aanbevolen procedures voor het oplossen van beginwaardeproblemen voor gewone differentiaalvergelijkingen. Deze aanbevelingen kunnen worden gedaan op grond van

a. de resultaten beschreven in de testrapporten van HULL et al. [1972], CRANE & FOX [1969] en KROGH [1973];
b. specifieke kenmerken van het op te lossen probleem, zoals stijfheid en grootte;
c. de door de gebruiker gewenste nauwkeurigheid van de oplossing.

De punten b. en c. zullen we nader bespreken aan de hand van de ingangen van tabel 2.

n klein/n groot

De tak - *n groot* - slaat op problemen waarvan

(i) het aantal componenten, n, moeilijkheden geeft met betrekking tot de geheugencapaciteit, ôfwel
(ii) het aantal componenten, n, het gebruik van bepaalde procedures praktisch onmogelijk maakt op grond van daarin optredende $0(n^3)$ en/of $0(n^2)$ processen (bijvoorbeeld LU-decomposities en matrixvermenigvuldigingen).

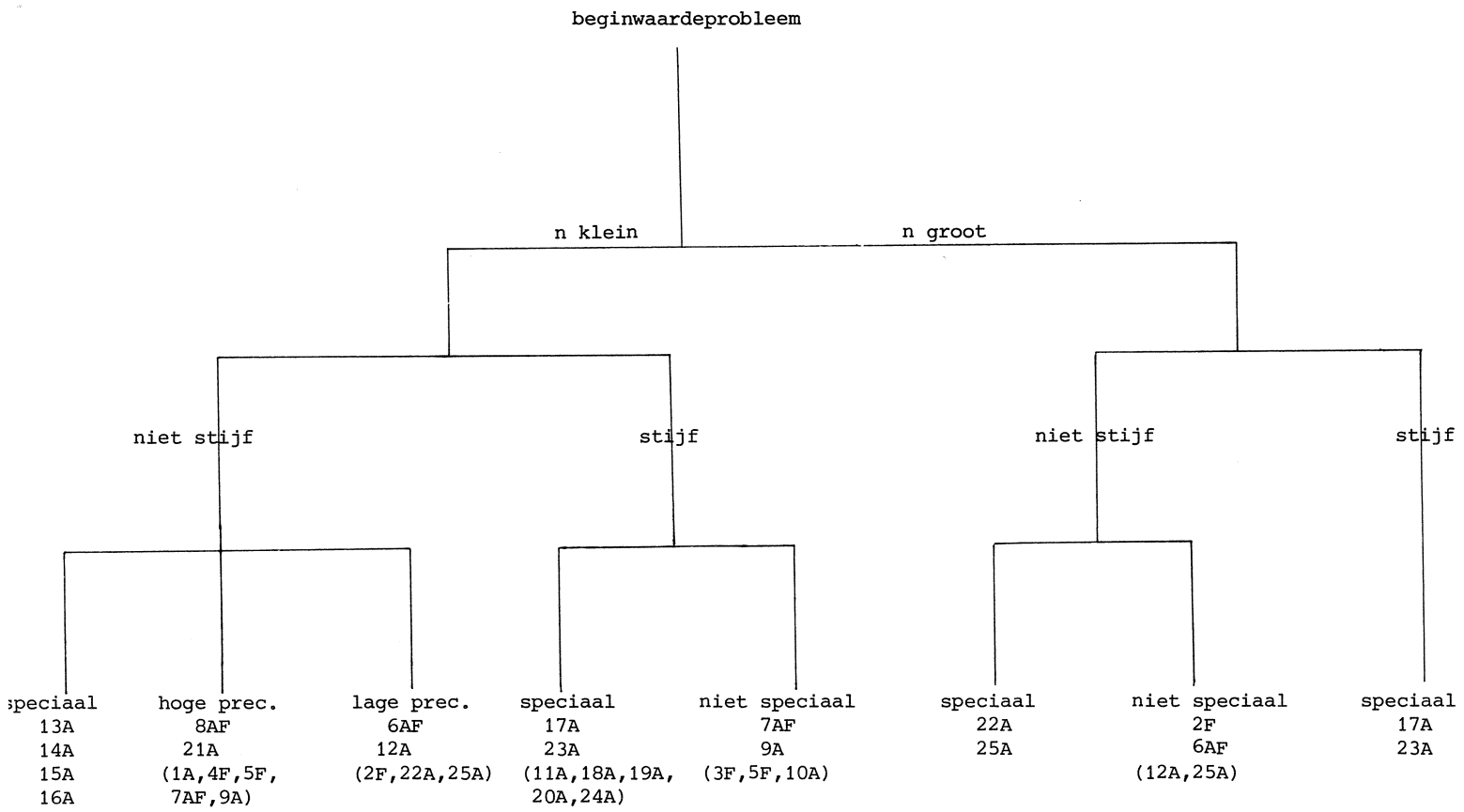
De overige problemen worden gedefinieerd door *n klein*. De problemen met *n groot* komen meestal uit de hoek van de (gediscretiseerde) partiële differentiaalvergelijkingen.

niet stijf/stijf

Stijve problemen worden gekenmerkt door zeer snel variërende componenten in de oplossing (samenhangend met een speciaal eigensysteem van de jacobiaan). De indeling *niet-stijf/stijf* is zinvol omdat stijve problemen praktisch niet op te lossen zijn met "niet-stijve" procedures.

TABEL 2

Aanbevolen programmatuur



niet speciaal/speciaal

De kwalificatie *speciaal* slaat enerzijds op de procedures die speciale informatie over het spectrum van de jacobiaan vragen en anderzijds op procedures die speciale eigenschappen bezitten, zoals het kunnen integreren tot het nulpunt van een expressie (13A en 14A) of het direct kunnen oplossen van tweede orde differentiaalvergelijkingen (15A en 16A).

hoge precisie/lage precisie

Omdat de verhouding (afgeleverde precisie) : (hoeveelheid rekenwerk) niet constant is voor een willekeurige procedure en een willekeurig probleem, is het belangrijk om het onderscheid *hoge precisie/lage precisie* te kunnen maken.

Opmerkingen en aanwijzigingen

1. Gebruik een kleine hmin, en een hmax, die niet groter is dan de minimale afstand tussen twee punten waarin u de oplossing wilt weten.
2. Gebruik voor een stijf probleem géén procedure waarvan de beschrijving niet expliciet vermeldt dat die procedure voor stijve problemen geschikt is. Omgekeerd kunt u wel "stijve" procedures op niet-stijve problemen loslaten (b.v. 5F, 7AF en 9A).
3. Als u geen idee heeft van de grootte-orde van de oplossing, dan doet u er verstandig aan om het integratieproces - zo mogelijk - met een relatieve en absolute precisie te sturen.
4. Kies de stuurprecisies niet te groot, maar ook niet onredelijk klein. Een goede stelregel is: niet groter dan 10^{-3} en niet kleiner dan 10^{-10}. De meeste procedures lopen "lekker" bij stuurprecisies in de buurt van 10^{-4} - 10^{-5}.
5. De nauwkeurigheid van de oplossing kunt u controleren door ôfwel het probleem enige malen met dezelfde procedure, met verschillende stuurprecisies, op te lossen, ôfwel door het probleem met verschillende procedures op te lossen.

2.1.4. Voorbeelden

Probleem 1.

In FRAZHO [1974] worden, door middel van een aantal differentiaalvergelijkingen, simulatiemodellen gegeven voor de groei van twee algensoorten ("groene" en "blauwgroene") onder invloed van anorganische stikstof en

fosfor, de intensiteit van het zonlicht en de watertemperatuur van het meer waarin het proces zich heet af te spelen. Het model dat hier als voorbeeld wordt gebruikt, ziet er uit als volgt:

$$\frac{dA_g}{dt} = [\mu_g(T)f_1(P)f_2(N) + c_1]A_g,$$

$$\frac{dA_{bg}}{dt} = [\mu_{bg}(T)f_1(P)f_3(N) + c_2]A_{bg},$$

(2.1.4.1)

$$\frac{dP}{dt} = P_I - c_3P - c_4f_1(P)[\mu_g(T)f_2(N)A_g + \mu_{bg}(T)f_3(N)A_{bg}],$$

$$\frac{dN}{dt} = N_I - c_3N - c_4f_1(P)f_2(N)[\mu_g(T)A_g + \mu_{bg}(T)A_{bg}].$$

Hierin zijn:

A_g, A_{bg}, N en P de concentraties van respectievelijk groene algen, blauwgroene algen, stikstof en fosfor;

P_I en N_I : fosfor en stikstof toevoer;
P_I en N_I zijn als (experimentele?) functies van de tijd, t, gegeven in de vorm van grafieken;

μ_g en μ_{bg} : groeisnelheden van de algen.
Deze functies van de temperatuur T zijn ook in grafiekvorm gegeven evenals de afhankelijkheid van T met betrekking tot t (seizoenen!).

In het artikel worden verder de overige functies en constanten van (2.1.4.1), alsook de beginvoorwaarden gegeven.

De vraag is hoe de algenconcentraties er over een tijdvak van 8 maanden (1 maart - 1 november) uit zullen zien.

De oplossing is een kromme met een groeiexplosie (voor de groene algen) omstreeks begin juni. In de bijlage staat een programma dat deze kromme als output geeft. Het probleem is niet stijf, maar wel moeilijk te integreren voor een aantal procedures.

Probleem 2.

Om ook een "stijve" methode te illustreren, beschouwen we een chemische reaktie met protoneringsevenwichten (de meer chemische kanten

en achtergronden zijn te vinden in BORKENT [1976].

$$P + B \xrightarrow{k_1} Am + H,$$

$$Am + B \xrightarrow{k_2} A + H ,$$

$$Ph \rightleftarrows P + H \text{ , evenwichtsconstante } K_1,$$

$$Amh \rightleftarrows Am + H, \text{ evenwichtsconstante } K_2,$$

$$Inh \rightleftarrows In + H, \text{ evenwichtsconstante } K_{in},$$

met

P	een diamine,
B	een bezoylchloride,
Am	een tussenprodukt,
A	het nuttig produkt,
In	een indikator,

Ph, Amh, Inh, geprotoneerde vormen van respectievelijk, P, Am en In.

De bovenstaande reakties zijn te beschrijven door een beginwaarde-probleem van 9 afhankelijke differentiaalvergelijkingen. Het stelsel is dan ook te herleiden tot het kleinere stelsel (met een bepaalde stof wordt nu de concentratie van die stof bedoeld)

$$\frac{dP}{dt} = -r_1 + r_3 ,$$

$$\frac{dB}{dt} = -r_1 + r_2 ,$$

(2.1.4.2)
$$\frac{dAm}{dt} = r_1 - r_2 + r_4 ,$$

$$\frac{dH}{dt} = r_1 + r_2 + r_3 + r_4 + r_5 ,$$

$$\frac{dA}{dt} = r_2 ,$$

met

$r_1 = k_1 PB,$

$r_2 = k_2 AmB,$

$r_3 = k_3(Ph - PH/K_1),$

$r_4 = k_4(Amh - AmH/K_2),$

$r_5 = k_5(Inh - InH/K_{in}),\ k_3, k_4, k_5 >> k_1, k_2,$

$In + Inh = In_0,$

$P + Am + A + Ph + Amh = P_0,$

$B + Am + 2A + Amh = B_0,$

$Ph + H + Inh = Am + 2A,$

en, op $t = 0$, P_0, B_0 en In_0 gegeven.

Het vrijwel direkt op evenwicht liggen van de laatste 3 reakties van (2.1.4.2) is kenmerkend voor de stijfheid van dit probleem.

De probleemstellers zijn geïnteresseerd in de concentraties van de verschillende stoffen op $t = i/20$, $i = 1(1)20$. In de bijlage geven we een programma wat de concentraties aan protonen op deze tijdstippen berekent.

Literatuur

HULL, T.E. et al. [1972], *Comparing numerical methods for ordinary differential equations*, SIAM J. Numer. Anal. 9, 603-637.

CRANE, P.C. & P.A. FOX [1969], *A comparative study of computer programs for integrating differential equations*, Numerical Mathematics Program Library Project, Bell Telephone Lab..

KROGH, F.T. [1973], *On testing a subroutine for the numerical integration of ordinary differential equations*, JACM 20, 545-562.

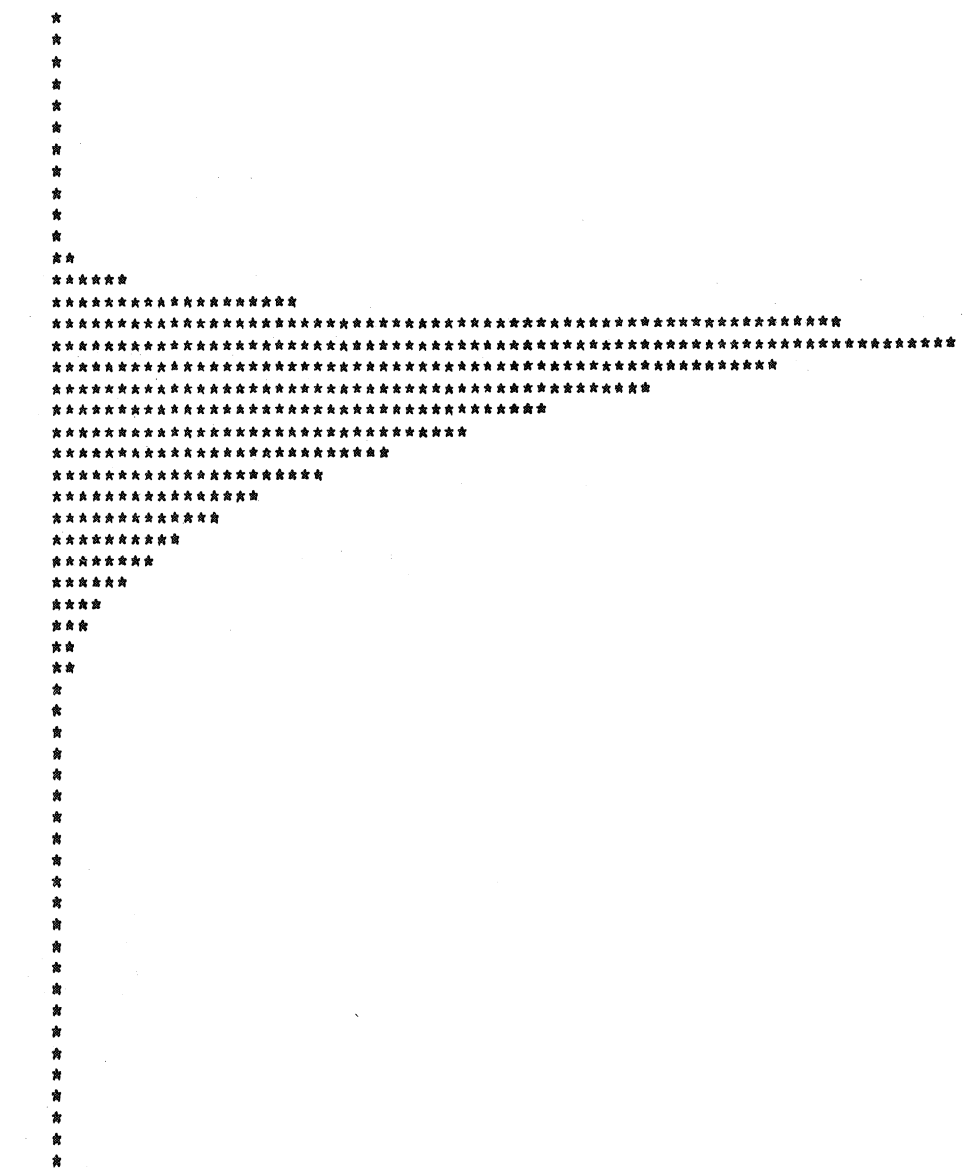
FRAZHO, D.B. et al. [1974], *Numerical integration aspects of a nutrient utilization ecological problem*, Lecture notes in mathematics 362, 167-182.

BORKENT, G. et al. [1976], *Kinetics of reactions of aromatic amines and acid chlorides in hexamethylphosphoric acid triamide*, to be published in Rec. Trav. Chim..

Bijlagen

```
      PROGRAM ALG(OUTPUT, INALG, TAPE4=INALG)
C        OPLOSSING MILIEUPROBLEEM M,B,V, DASCRU(IMSL)
      INTEGER ISYM(100)
      REAL X(4),WK(16)
      COMMON /DATA/ RBG, RKN, SMALLN, TAU, W, RG, RKP, BIGY
     +       /MUBG/ XMUBG(18), FMUBG(18)  /MUG/ XMUG(23), FMUG(23)
     +       /TEMP/ XTEMP(15), FTEMP(15)  /NI/   XNI(18),  FNI(18)
     +       /PI/     XPI(15),   FPI(15)
      EXTERNAL F
      DATA ISYM, RBG, RKN, SMALLN, TAU,   W , RG, RKP, BIGY
     +/ 100*1H*, .5 , .25,  .25  , .5 ,  12., 2., .03, .01/
      READ(4, *) (XMUBG(I), I= 1, 18), (FMUBG(I), I= 1, 18),
     +           (XMUG(I),  I= 1, 23), (FMUG(I),  I= 1, 23),
     +           (XTEMP(I), I= 1, 15), (FTEMP(I), I= 1, 15),
     +           (XNI(I),   I= 1, 18), (FNI(I),   I= 1, 18),
     +           (XPI(I),   I= 1, 15), (FPI(I),   I= 1, 15)
      A= .0 $ B= .125 $ H= 1E-5 $ N= 4
      X(1)= X(2)= .5  $ X(3)= .05 $ X(4)= 1.
      DO 2 J= 1, 64
      CALL DASCRU(F, A, B, H, N, X, WK, IER)
      IPOS= IFIX(X(1) * 5 + .5)
      PRINT 1, (ISYM(I), I= 1, IPOS)
1     FORMAT(1X, 100A1)
      A= B
2     B= B + .125
      END

      SUBROUTINE F(X, T, N, XP)
      REAL XP(4), X(4)
      COMMON /DATA/ RBG, RKN, SMALLN, TAU, W, RG, RKP, BIGY
      V1= X(3) / (RKP + X(3)) $ V2= X(4) / (RKN + X(4)) $ V4= FXMUG(T)
      V5= FXMUBG(T)           $ V6= V4 * X(1) * V2      $ V7= V5 * X(2)
      V3= (X(4) + SMALLN) / (RKN + X(4) + SMALLN)
      XP(1)= (V4 * V1 * V2 - TAU - RG) * X(1)
      XP(2)= (V5 * V1 * V3 - TAU - RBG) * X(2)
      XP(3)= PI(T) - X(3) * TAU - BIGY * V1 * (V6 + V7 * V3)
      XP(4)= FXNI(T) - X(4) * TAU - V1 * (V6 + V7 * V2) / W
      RETURN
      END

      REAL FUNCTION FXMUBG(X)
      COMMON /MUBG/ XMUBG(18), FMUBG(18)
      X1= TEMP(X)
      IF (X1 .LT. 5.)  X1= 5.
      IF (X1 .GT. 25.) X1= 25.
      DO 1 I= 2, 18
      IF (X1 .LE. XMUBG(I)) GO TO 2
1     CONTINUE
2     FXMUBG= ((X1 - XMUBG(I-1)) * FMUBG(I) - (X1 - XMUBG(I))
     +        * FMUBG(I-1)) / (XMUBG(I) - XMUBG(I-1))
      RETURN
      END
```

```
      REAL FUNCTION FXMUG(X)
      COMMON /MUG/ XMUG(23), FMUG(23)
      X1= TEMP(X)
      IF (X1 .LT. 5.)  X1= 5.
      IF (X1 .GT. 25.)  X1= 25.
      DO 1 I= 2, 23
      IF (X1 .LE. XMUG(I)) GO TO 2
    1 CONTINUE
    2 FXMUG= ((X1 - XMUG(I-1)) * FMUG(I) - (X1 - XMUG(I))
     +       * FMUG(I-1)) / (XMUG(I) - XMUG(I-1))
      RETURN
      END

      REAL FUNCTION TEMP(X)
      COMMON /TEMP/ XTEMP(15), FTEMP(15)
      DO 1 I= 2, 15
      IF (X .LE. XTEMP(I)) GO TO 2
    1 CONTINUE
    2 TEMP= ((X - XTEMP(I-1)) * FTEMP(I) - (X - XTEMP(I))
     +       * FTEMP(I-1)) / (XTEMP(I) - XTEMP(I-1))
      RETURN
      END

      REAL FUNCTION FXNI(X)
      COMMON /NI/ XNI(18), FNI(18)
      DO 1 I= 2, 18
      IF (X .LE. XNI(I)) GO TO 2
    1 CONTINUE
    2 FXNI= ((X - XNI(I-1)) * FNI(I) - (X - XNI(I)) * FNI(I-1)) /
     +       (XNI(I) - XNI(I-1))
      RETURN
      END

      REAL FUNCTION PI(X)
      COMMON /PI/ XPI(15), FPI(15)
      DO 1 I= 2, 15
      IF (X .LE. XPI(I))  GO TO 2
    1 CONTINUE
    2 PI= ((X - XPI(I-1)) * FPI(I) - (X - XPI(I)) * FPI(I-1)) /
     +       (XPI(I) - XPI(I-1))
      RETURN
      END
```



```
"BEGIN" "COMMENT" OPLOSSING CHEMISCHE REACTIES M,B,V, MULTISTEP(NUMAL);
     "INTEGER" I, NFE;
     "REAL" K1, K2, K3, K4, K5, KK1, KK2, KKIN, X, XE,
            R1, R2, R3, R4, R5, P, B, AM, H, A, PH, AMH, INH, IN,
            P0, B0, IN0;
     "ARRAY" Y[1 : 30], SAVE[-38 : 30], JAC[1 : 5, 1 : 5], YMAX[1 : 5];
     "BOOLEAN" FI;

     "PROCEDURE" DER(DY); "ARRAY" DY;
     "BEGIN" P:= Y[1]; B:= Y[2]; AM:= Y[3]; H:= Y[4]; A:= Y[5];
        AMH:= - 2 * A + B0 - AM - B; PH:= P0 - P - AM - A - AMH;
        INH:= 2 * A + AM - PH - H; IN:= IN0 - INH;
        R1:=K1 * P * B; R2:= K2 * AM * B; R3:= (-P * H /KK1 + PH) * K3;
        R4:=(-AM * H / KK2 + AMH) * K4; R5:=(-IN * H / KKIN + INH)* K5;
        DY[1]:= R3 - R1; DY[2]:= - R1 - R2; DY[3]:= R1 - R2 + R4;
        DY[4]:= R1 + R2 + R3 + R4 + R5; DY[5]:= R2; NFE:= NFE + 1
     "END" DER;

     "PROCEDURE" OUT(H, K); "VALUE" H, K; "REAL" H; "INTEGER" K;
     "IF" SAVE[-1] "<> 0 "THEN"
     "BEGIN" OUTPUT(61, "("""("SERIOUS TROUBLES ENCOUNTERED AT X ="")",
        ZD,3D, /")", X); "GOTO" ESC
     "END";

     "PROCEDURE" MULTISTEP(PARS);"CODE" 33080;

     P0:= ,0175; B0:= ,0152; IN0:= "-5;
     I:= 1; "FOR" H:= P0, B0, 0, 0, 0 "DO"
     "BEGIN" Y[I]:= H; YMAX[I]:= 1; I:= I + 1 "END";
     K1:= 5000; K2:= 2060; KK1:= ,244"-4; KK2:= ,358"-4;KKIN:=,3437"-3;
     K3:= K4:= K5:= "6;
     X:= 0; NFE:= 0;
     OUTPUT(61, "("""("  TIME    CONCENTR, H+")", //")");
     "FOR" XE:= ,05, XE + ,05 "WHILE" XE <= 1,01 "DO"
     "BEGIN" FI:= XE = ,05;
         MULTISTEP(X, XE, Y, "-8, ,05, YMAX, "-6, FI, SAVE,
                   DER, "FALSE", JAC, "TRUE", 5, OUT);
         OUTPUT(61, "(" 2ZD,3D, 4B, -D,3D"+D, /")", X, Y[4])
     "END";
     OUTPUT(61, "("//, "(" NUMBER OF FUNCTION EVALUATIONS :")", 3ZD, /,
     "(" LOCAL ERROR BOUND EXCEEDED     :")", 3ZD, "(" TIMES")", / ")",
     NFE, SAVE[-2]);
  ESC:
"END"
```

TIME	CONCENTR. H+
0.050	8.653"-5
0.100	1.749"-4
0.150	2.642"-4
0.200	3.504"-4
0.250	4.309"-4
0.300	5.044"-4
0.350	5.710"-4
0.400	6.310"-4
0.450	6.852"-4
0.500	7.342"-4
0.550	7.786"-4
0.600	8.191"-4
0.650	8.560"-4
0.700	8.899"-4
0.750	9.210"-4
0.800	9.496"-4
0.850	9.761"-4
0.900	1.001"-3
0.950	1.023"-3
1.000	1.044"-3

NUMBER OF FUNCTION EVALUATIONS : 464
LOCAL ERROR BOUND EXCEEDED : 0 TIMES

2. BEGINWAARDE- EN BEGIN-RANDWAARDEPROBLEMEN

2.2. Begin-randwaardeproblemen voor partiële differentiaalvergelijkingen

door P.J. van der Houwen
(Mathematisch Centrum)

2.2.1 Inleiding

De numerieke oplossing van begin-randwaardeproblemen voor partiële differentiaalvergelijkingen is een zeer gevarieerd onderwerp dat nog volop in ontwikkeling is. Er zijn tientallen oplossingstechnieken ontwikkeld en dit aantal groeit nog steeds. Uiteraard is het onmogelijk om hier een overzicht van al deze algoritmen te geven, te meer daar deze algoritmen veelal voor precies één probleem ontwikkeld zijn. Numerieke programmatuur voor partiële differentiaalvergelijkingen is dan ook (nog) vrij zeldzaam; het is weinig zinvol voor één enkel probleem een algoritme in een programmatheek op te nemen en uitvoerig te documenteren, tenzij het probleem in kwestie van voldoende importantie is, dat wil zeggen dat anderen dan alleen de auteur van het algoritme, hetzelfde probleem, met bijvoorbeeld andere begin- en randvoorwaarden, ook willen oplossen. In het algemeen echter zou men een algoritme pas dan in een programmatheek willen opnemen wanneer een voldoend grote klasse van begin-randwaardeproblemen hiermee opgelost kan worden; maar hoe groter de toepasbaarheid, hoe geringer de efficiëntie en daarmee deste groter de neiging van de gebruiker om òf het zelf maar te programmeren òf naar een numericus te stappen om hem de zaak te laten opknappen. Hoe het ook zij, geen van de vier programmatheken waaraan dit colloquium gewijd is, hebben programmatuur waarmee *rechtstreeks* begin-randwaardeproblemen opgelost kunnen worden. (Voor de volledigheid willen we hier nog wel wijzen op enkele programmatuurpakketten die geen deel uitmaken van een grotere programmatheek en speciaal ontwikkeld zijn voor bepaalde klassen van partiële differentiaalvergelijkingen; het betreft de pakketten van MORRIS en SCHIESSER [1968], CANDENAS en KARPLUS [1970], ZELLNER [1970], SINCOVEC en MADSEN [1975] en van COHEN en GROSSMAN [1975].) Het ontbreken van partiële differentiaalvergelijkingen-programmatuur in een programmatheek wil niet zeggen dat dit soort problemen dan maar apart geprogrammeerd moet worden. In deze voordracht zullen we een methode bespreken waarmee een begin-randwaardeprobleem tot een beginwaardeprobleem voor gewone differentiaalvergelijkingen gereduceerd kan worden, zodat we dan alleen nog maar een programmatheek met goede "gewone differentiaalvergelijkingen-integratoren" nodig hebben.

2.2.2 Concrete practijkproblemen

Aan de hand van een drietal practijkproblemen willen we laten zien hoe men toch met behulp van de programmatheken ACCULIB, IMSL, NAG en NUMAL niet-triviale problemen kan oplossen wanneer we zelf ook een "klein beetje

werk willen verzetten". In deze paragraaf wordt een korte bespreking gegeven van de gekozen practijkproblemen. Het eerste probleem betreft een niet-lineaire warmte-diffusie in een staaf waarin aanvankelijk een temperatuursprong voorkomt. Dit probleem is afkomstig van het FOM-laboratorium en is op het MC uitvoerig bestudeerd. Het tweede probleem is ontleend aan HOUGHTON en KASAHARA [1968], genoemd in SINCOVEC en MADSEN [1975], en beschrijft de niet-lineaire stroming over een geisoleerde barriere in een rivierbedding. Het derde probleem, de voorspelling van de waterstanden op de Noordzee, is in het kader van de Delta-commissie ingesteld na de stormramp van 1953, diepgaand onderzocht op het MC, zowel wat de analytische als de numerieke aspecten betreft.

2.2.2.1 Niet-lineaire diffusie met discontinue beginvoorwaarden

Gegeven de ééndimensionale, parabolische differentiaalvergelijking

$$(2.2.2.1)\quad \frac{\partial T}{\partial t} = (\alpha T+\beta)\left[\frac{\partial^2}{\partial x^2} + \frac{1}{x}\frac{\partial}{\partial x}\right] T$$

voor alle niet-negatieve waarden van de plaatsvariabele x en de tijdsvariabele t, met de randvoorwaarden

$$(2.2.2.2)\quad \frac{\partial T}{\partial x}(t,0) = 0 \qquad ,0 \leq t \leq \infty$$
$$T(t,\infty) = 0$$

en de beginvoorwaarde

$$(2.2.2.3)\quad T(0,x) = \begin{cases} 1 & \text{voor } 0 \leq x \leq 1 \\ 0 & \text{voor } x > 1 \end{cases} .$$

Gevraagd de oplossing van (2.2.2.1) - (2.2.2.3) met een nauwkeurigheid van 1% voor een reeks waarden van $\alpha < 0$ en $\beta > 0$ voor $0 \leq x \leq 2$ en tot het tijdstip waarop $T(t,0) \cong 2\ T(t,2)$.

2.2.2.2 Niet-lineaire stroming over een parabolische drempel

Gegeven het ééndimensionale stelsel van hyperbolische differentiaalvergelijkingen

$$\frac{\partial u}{\partial t} = - u\frac{\partial u}{\partial x} - g\frac{\partial}{\partial x}(h+b)$$

(2.2.2.4) $$,0 \leq t \leq 1.83,\ |x| \leq 400$$

$$\frac{\partial h}{\partial t} = - \frac{\partial}{\partial x}(hu)$$

waarin u de horizontale snelheid, h de diepte, b de bedding, x de coordinaat langs de rivier, t de tijd en g de versnelling van de zwaartekracht (g = = 980 cm/sec^2) voorstelt (zie SINCOVEC en MADSEN [1975]); de functie b wordt gegeven door

(2.2.2.5) $$b(x) = \max\left(0,10-10\left(\frac{x}{40}\right)^2\right),$$

de beginvoorwaarden door

(2.2.2.6) $$u(0,x) = 40,\ h(0,x) = 20-b(x)$$

en de randvoorwaarden door

(2.2.2.7) $$u(t,-400) = u(t,400) = 40,\ h(t,-400) = h(t,400) = 20.$$

Gevraagd de oplossing van (2.2.2.4) - (2.2.2.7) in de rechthoek $|x| \leq 400$, $0 \leq t \leq 1{\cdot}83$ met een nauwkeurigheid van 1%.

2.2.2.3 Gelineariseerd Noordzeemodel

Gegeven het stelsel 2-dimensionale, hyperbolische vergelijkingen

$$\frac{\partial u}{\partial t} = - \lambda u + \omega v - g\frac{\partial}{\partial x} z + f_x,$$

(2.2.2.8) $$\frac{\partial v}{\partial t} = - \omega u - \lambda v - g\frac{\partial}{\partial y} z + f_y,$$

$$\frac{\partial z}{\partial t} = - h\frac{\partial}{\partial x} u - h\frac{\partial}{\partial y} v,$$

waarin z en h respectivelijk de verhoging en de diepte van de zee in rust, u en v de horizontale snelheidscomponenten, λ de bodemwrijvingscoefficient ($\lambda = 25\ 10^{-6}$/sec), ω de Coriolisparameter ($\omega = 125\ 10^{-6}$/sec), g de versnelling van de zwaartekracht ($g = 9.8$ m/sec^2), f_x en f_y de horizontale componenten van de door de wind uitgeoefende schuifkracht, x en y de plaatscoördinaten en t de tijdsvariabele voorstellen; de randvoorwaarden zijn

$$\begin{aligned} &z(t,x,8\ 10^5) = 0 \\ &v(t,x,0) \quad\ = 0 \end{aligned} \quad \text{voor } 0 \le x \le 4\ 10^5$$

$$u(t,0,y) = u(t,4\ 10^5,y) = 0 \quad \text{voor} \quad 0 \le y \le 8\ 10^5 \tag{2.2.2.9}$$

en de beginsvoorwaarden

$$u(0,x,y) = v(0,x,y) = z(0,x,y) = 0. \tag{2.2.2.10}$$

Gevraagd de oplossing van (2.2.2.8) - (2.2.2.10) in het gebied $0 \le x \le 4\ 10^5$, $0 \le y \le 8\ 10^5$, $0 \le t \le 36\ 10^3$ voor het windveld

$$f_x = 0, \quad f_y = \frac{10^{-4}}{h} \tag{2.2.2.11}$$

en de dieptefunctie

$$h = \left[40 - \frac{15x}{2\ 10^5}\right] \cdot \left[1 + \left(\frac{y}{8\ 10^5}\right)^2\right] . \tag{2.2.2.12}$$

Gevraagde nauwkeurigheid 10%.

2.2.3. Semi-discretisatie

Onder *semi-discretisatie* (ook wel *partiële discretisatie* of de *methode der lijnen* genoemd) verstaat men de vervanging van het door de plaatsvariabelen doorlopen gebied door een rooster met roosterpunten $\vec{x}_j$, de vervanging van de afhankelijke variabelen door op de roosterpunten $\vec{x}_j$ gedefinieerde roosterfuncties, en tenslotte de vervanging van alle differentiaties naar plaatsvariabelen door op het rooster gedefinieerde differentiaalquotienten. Bijvoorbeeld wanneer men in het (x,y)-vlak de roosterpunten

(2.2.3.1) $\vec{x}_j = (x_k, y_\ell) = (k\Delta x, \ell\Delta y)$, $j = (k,\ell)$,

waarin k en ℓ gehele getallen, Δx en Δy gegeven roosterpunt-afstanden zijn, definieert dan zou men $\partial/\partial x$ en $\partial/\partial y$ kunnen vervangen door differentie-operatoren $[\partial/\partial x]$ en $[\partial/\partial y]$ gedefinieerd door

$$(2.2.3.2) \quad \left[\frac{\partial}{\partial x}\right] f(x_k, y_\ell) = \frac{f(x_k+\Delta x, y_\ell) - f(x_k-\Delta x, y_\ell)}{2\Delta x},$$

en iets analoogs voor $[\partial/\partial y]$.

Op deze wijze kan een begin-randwaardeprobleem voor een *partiële* differentiaalvergelijking in een beginwaardeprobleem voor een stelsel *gewone* differentiaalvergelijkingen omgezet worden. We zullen dit uitvoeren voor de drie in de vorige paragraaf genoemde problemen.

2.2.3.1 Het diffusie-probleem

In de oorspronkelijke behandeling van dit probleem werd de partiële differentiaalvergelijking eerst getransformeerd met behulp van de transformatiefunctie

$$(2.2.3.3) \quad \begin{aligned} z &= 3x - \frac{2}{5\pi}\sin(5\pi(x-\tfrac{4}{5})) - \frac{8}{5} \text{ voor } |x-1| \le \frac{1}{5}, \\ z &= x \text{ voor } x \le \frac{4}{5}, \\ z &= x + \frac{4}{5} \text{ voor } x \ge \frac{6}{5}, \end{aligned}$$

tot de vergelijking

$$(2.2.3.4) \quad \frac{\partial T}{\partial t} = (\alpha T+\beta)\left[\left(\frac{dz}{dx}\right)^2 \frac{\partial^2}{\partial z^2} + \left(\frac{d^2z}{dx^2} + \frac{1}{x}\frac{dz}{dx}\right)\frac{\partial}{\partial z}\right] T.$$

Hiermee wordt bereikt dat een equidistant rooster op de z-as correspondeert met een zeer fijn rooster in de buurt het kritische punt x = 1 op de x-as. Verder zullen we in de buurt van het kritische punt z = 7/5 (i.e. x = 1) de beginvoorwaarde vervangen door

$$(2.2.3.5) \quad T(0,x(z)) = \tfrac{1}{2}\left[1+\cos\left(\frac{5z-6}{2}\pi\right)\right], \quad \frac{6}{5} \le z \le \frac{8}{5}.$$

We zijn nu zover om de semi-discretisatie uit te voeren: we vervangen de continue variabele z door de discrete variabele j Δ z, waarin Δ z de roosterpunt-afstand is. Verder schrijven we kortheidshalve

$$\left(\frac{dz}{dx}\right)^2 = a(z),\ a(j\Delta z) = a_j,$$

$$\frac{d^2z}{dx^2} + \frac{1}{x}\frac{dz}{dx} = b(z),\ b(j\Delta z) = b_j, \tag{2.2.3.6}$$

$$T(t,x(j\Delta z)) = T_j(t)$$

en benaderen we in j Δ z, $\partial T/\partial Z$ en $\partial^2 T/\partial Z^2$ door respectievelijk

$$\frac{T_{j+1}-T_{j-1}}{2\Delta z} \text{ en } \frac{T_{j+1}-2T_j+T_{j-1}}{(\Delta z)^2}. \tag{2.2.3.7}$$

We vinden dan het volgende stelsel gewone differentiaalvergelijkingen:

$$\frac{dT_0}{dt} = \frac{4}{(\Delta z)^2}(\alpha T_0+\beta)\, a_0\, (T_1-T_0),$$

$$\frac{dT_j}{dt} = \frac{1}{(\Delta z)^2}(\alpha T_j+\beta)\left[a_j-\tfrac{1}{2}\Delta b_j)T_{j-1}-2a_jT_j+(a_j+\tfrac{1}{2}\Delta z b_j)T_{j+1}\right], j = 1,2,\ldots .$$

Dit is een oneindig groot stelsel; in de practijk integreert men uiteraard alleen de relevante vergelijkingen, dat wil zeggen wanneer j zo groot is (jΔz zo groot) dat T_j beneden een bepaalde drempel gezakt, worden geen verdere differentiaalvergelijkingen meer beschouwd. In het begin van het integratieproces is het aantal vergelijkingen nog gering (nl. 8/5Δz) maar wordt groter naarmate de warmte zich naar rechts uitbreidt.

2.2.3.2 Het drempel-probleem

Kiezen we een uniform rooster met roosterpunten j Δ x langs de rivier, dan kan (2.2.2.4) gediscretiseerd worden tot het stelsel

$$\frac{du_{-j_0}}{dt} = -\frac{u_{-j_0}}{2\Delta x}\,(u_{-j_0+1}-40) - \frac{g}{2\Delta x}\,(h_{-j_0-1}+b_{-j_0+1}-20-b_{-j_0-1}),$$

$$\frac{dh_{-j_0}}{dt} = -\frac{1}{2\Delta x}\,(h_{-j_0+1}u_{-j_0+1}-800),$$

$$\frac{du_j}{dt} = -\frac{u_j}{2\Delta x}\,(u_{j+1}-u_{j-1}) - \frac{g}{2\Delta x}(h_{j+1}+b_{j+1}-h_{j-1}b_{j-1}),$$

(2.2.3.9)

$$\frac{dh_j}{dt} = -\frac{1}{2\Delta x}\,(h_{j+1}u_{j+1}-h_{j-1}u_{j-1}),$$

$$\frac{du_{j_0}}{dt} = -\frac{u_{j_0}}{2\Delta x}\,(40-u_{j_0-1}) - \frac{g}{2\Delta x}\,(20+b_{j_0+1}-h_{j_0-1}-b_{j_0-1}),$$

$$\frac{dh_{j_0}}{dt} = -\frac{1}{2\Delta x}\,(800-h_{j_0-1}u_{j_0-1}).$$

Hierin is $j_0 = \frac{400-\Delta x}{\Delta x}$ en loopt j van $-j_0 + 1$ tot en met $j_0 - 1$.

2.2.3.3 Het Noordzee-probleem

Kies in het (x,y)-vlak een rooster met vierkante maten van Δ x by Δ x meter; om de notatie wat compacter te maken definieren we de getallen

$$K = \frac{4\ 10^5}{\Delta x},\quad L = \frac{8\ 10^5}{\Delta x}$$

en de schuif-operatoren $X_\pm$ en $Y_\pm$ gedefinieerd door

$$X_\pm\, u_{k,\ell} = u_{k\pm1,\ell},$$

$$Y_\pm\, u_{k,\ell} = u_{k,\ell\pm1}$$

waarin u de numerieke benadering voor $u(k\Delta x,\ell\Delta x)$ voorstelt.

Het begin-randwaardeprobleem (2.2.2.8) - (2.2.2.10) kan nu benaderd worden door het stelsel differentiaalvergelijkingen:

(2.2.3.10a) *Interne punten* $\quad \frac{du}{dt} = -\lambda u + \omega v - \frac{g}{2\Delta x}\,(X_+-X_-)z + f_x,$

$(k=1,2,\ldots,K-1,\quad \frac{dv}{dt} = -\omega u - \lambda v - \frac{g}{2\Delta x}(Y_+ - Y_-)z + f_y,$

$\ell=1,2,\ldots,L-1) \quad \frac{dz}{dt} = -\frac{h}{2\Delta x}((X_+ - X_-)u + (Y_+ - Y_-)v);$

(2.2.3.10b) *Hoekpunten* $\frac{du}{dt} = \frac{dv_{0,0}}{dt} = \frac{dv_{K,0}}{dt} = 0$, $\frac{dv_{0,L}}{dt}$ en $\frac{dv_{K,L}}{dt}$ analoog aan de West- en Oostkust formules,

$(k=0,K, \quad \frac{dz_{0,0}}{dt} = -\frac{h_{0,0}}{\Delta x}(u_{1,0} + v_{0,1}),$

$\ell=0,L) \quad \frac{dz_{K,0}}{dt} = -\frac{h_{K,0}}{\Delta x}(-u_{K-1,0} + v_{K,1}),$

$\frac{dz_{0,L}}{dt} = \frac{dz_{K,L}}{dt} = 0;$

(2.2.3.10c) *Zuidkust* $\frac{du}{dt} = -\lambda u - \frac{g}{2\Delta x}(X_+ - X_-)z + f_x,$

$(k=1,\ldots,K-1, \quad \frac{dv}{dt} = 0,$

$\ell=0) \quad \frac{dz}{dt} = -\frac{h}{2\Delta x}((X_+ - X_-)u + 2Y_+ v);$

(2.2.3.10d) *Westkust* $\frac{du}{dt} = 0,$

$(k=0, \quad \frac{dv}{dt} = -\lambda v - \frac{g}{2\Delta x}(Y_+ - Y_-)z + f_y,$

$\ell=1,\ldots,L-1) \quad \frac{dz}{dt} = -\frac{h}{2\Delta x}(2X_+ u + (Y_+ - Y_-)v);$

(2.2.3.10e) *Oostkust* $\frac{du}{dt} = 0,$

$(k = K, \quad \frac{dv}{dt} = -\lambda v - \frac{g}{2\Delta x}(Y_+ - Y_-)z + f_y,$

$\ell=1,\ldots,L-1) \quad \frac{dz}{dt} = -\frac{h}{2\Delta x}(-2X_- u + (Y_+ - Y_-)v);$

(2.2.3.10f) *Oceaan-punten* $\frac{du}{dt} = -\lambda u + \omega v - \frac{g}{2\Delta x}(X_+ - X_-)z + f_x,$

$(k=1,\ldots,K-1, \quad \frac{dv}{dt} = -\omega u - \lambda v + \frac{g}{2\Delta x} Y_- z + f_y,$

$\ell=L) \quad \frac{dz}{dt} = 0.$

2.2.4. Kenmerken van door semi-discretisatie verkregen stelsels

Een gemeenschappelijk kenmerk van de stelsels (2.2.3.8), (2.2.3.9) en (2.2.3.10) is het relatief grote aantal vergelijkingen. In het *diffusie-probleem* zal bij een maaswijdte van $\Delta z = 1/10$ (dan wordt de beginvoorwaarde in het critische interval $6/5 \leq z \leq 8/5$ door 5 punten gepresenteerd), het aantal differentiaalvergelijkingen minimaal 16 bedragen en kan oplopen tot 100 vergelijkingen. In het *drempel-probleem* krijgen we voor $\Delta x = 1$ een stelsel van 797 differentiaalvergelijkingen en het Noordzee-probleem telt voor mazen van 10 bij 10 km, $3(K+1)(L+1) = 9963$ differentiaalvergelijkingen. Een tweede kenmerk is dat het spectrum van de Jacobiaan van door partiële discretisatie verkregen stelsels meestal òf in een strook langs de negatieve as (parabolische vergelijkingen) òf in een strook langs de imaginaire as (hyperbolische vergelijkingen) ligt. We zullen dit nagaan voor de stelsels (2.2.3.8) en (2.2.3.9); voor het stelsel (2.2.3.10) verwijzen we naar FISCHER [1959].

2.2.4.1. Het diffusie-probleem

De matrix van partiële afgeleiden van het rechterlid van 2.2.3.8 (de Jacobiaan van het stelsel) is van de vorm:

$$(2.2.4.1.)\quad \frac{1}{(\Delta z)^2}\begin{pmatrix} -4a_0(d_0+\alpha(T_0-T_1)) & 4a_0d_0 & 0 & 0 & 0 & \dots \\ d_1c_1^- & -2a_1d_1+e_1 & d_1c_1^+ & 0 & 0 & \dots \\ 0 & d_2c_2^- & -2a_2d_2+e_2 & d_2c_2^+ & 0 & \dots \\ 0 & 0 & \ddots & \ddots & \ddots & \end{pmatrix},$$

waarin

$$c_j^{\pm} = a_j \pm \tfrac{1}{2}\,\Delta z\, b_j,$$

$$d_j = \alpha T_j + \beta,$$

$$e_j = \alpha(c_j^- T_{j-1} - 2a_jT_j + c_j^+ T_{j+1}).$$

We hebben te maken met een tridiagonale matrix; hiervan is bekend dat de

eigenwaarden reeël zijn wanneer de neven-diagonaalelementen allen hetzelfde teken hebben (zie e.g. Wilkinson [1968, p. 333]). Aangezien de diffusiecoefficient d_j positief verondersteld mag worden gaan we na hoe $c_j^{\pm}$ zich gedraagt; volgens de definitie van $c_j^{\pm}$ geldt in elk geval dat alle $c_j^{\pm}$ *positief* zijn als $\Delta z \to 0$ ($a = (\partial z/\partial x)^2$), zodat het voor de hand ligt om de voorwaarde

$$(2.2.4.2) \quad \Delta z < 2 \frac{a_j}{|b_j|}, \quad j = 0,1,2,\ldots,$$

te stellen. Het is eenvoudig na te gaan dat in het geval (2.2.3.3) aan (2.2.4.2) zeker voldaan is als $\Delta z \leq 1/10$; zodat conditie (2.2.4.2) slechts een geringe beperking voor het rooster vormt.

Een verdere analyse van de Jacobiaan (2.2.4.1) leert dat de diagonaalelementen als negatieve getallen beschouwd kunnen worden omdat T_0-T_1 en de termen e_j verwaarloosd kunnen worden ten op zichte van $4\, a_0\, d_0$ en $2\, a_j\, d_j$. Toepassing van de stelling van Gerschgorin leert dan dat de eigenwaarden van (2.2.4.1) ongeveer in het interval

$$(2.2.4.3) \quad \left[-4 \frac{\max_j\{2d_0a_0, d_ja_j\}}{(\Delta z)^2}, 0\right]$$

liggen. Substitutie van a_j en d_j geeft het "veilige" eigenwaarde-interval

$$(2.2.4.3') \quad \left[-\frac{16\beta}{(\Delta z)^2}, 0\right].$$

2.2.4.2. <u>Het drempelprobleem</u>

Uit (2.2.3.9) volgt direct

$$(2.2.4.4) \quad J = -\frac{1}{2\Delta x}\begin{pmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{pmatrix},$$

waarin

$$J_{11} = \begin{pmatrix} u_{1-j_0}-40 & u_{-j_0} & 0 & \dots & 0 \\ -u_{1-j_0} & u_{2-j_0}-u_{-j_0} & u_{1-j_0} & \dots & 0 \\ & & \dots & & \\ 0 & \dots & -u_{j_0-1} & u_{j_0}-u_{j_0-2} & u_{j_0-1} \\ 0 & \dots & 0 & -u_{j_0} & 40-u_{j_0-1} \end{pmatrix}$$

$$J_{12} = \begin{pmatrix} 0 & g & 0 & \dots & 0 \\ -g & 0 & g & \dots & 0 \\ & & \dots & & \\ 0 & \dots & -g & 0 & g \\ 0 & \dots & 0 & -g & 0 \end{pmatrix},$$

$$J_{21} = \begin{pmatrix} 0 & h_{1-j_0} & 0 & \dots & 0 \\ -h_{-j_0} & 0 & h_{2-j_0} & \dots & 0 \\ & & \dots & & \\ 0 & \dots & -h_{j_0-2} & 0 & h_{j_0-2} \\ 0 & \dots & 0 & -h_{j_0-1} & 0 \end{pmatrix},$$

en waarin J_{22} uit J_{21} afgeleid kan worden door h door u te vervangen.

Omdat de roosterfuncties u_j en h_j als langzaam met j varierende functies beschouwd kunnen worden (uiteraard voor een voldoend fijn rooster) zijn de matrices J_{rk} "bijna" scheefsymmetrisch. Men kan aantonen dat voor *constante roosterfuncties* u_j en h_j de eigenwaarden van dit type matrices *zuiver imaginair* zijn (methode van constante coefficienten van VON NEUMANN (zie RICHTMYER-MORTON [1967])); de eigenwaarden van J blijken namelijk voor u_j en h_j constant dezelfde te zijn als die van de matrix

$$-\frac{i \sin(\gamma\Delta x)}{\Delta x}\begin{pmatrix} u & g \\ h & u \end{pmatrix},$$

waarin γ de waarden $\pm 1, \pm 2, \dots$ doorloopt. We vinden voor de eigenwaarden δ:

$$(2.2.4.5) \qquad \delta = -\frac{u \pm \sqrt{gh}}{\Delta x} i \sin(\gamma\Delta x),$$

waaruit het imaginaire eigenwaarde-interval

$$(2.2.4.6)\quad \left[-\frac{|u|+\sqrt{gh}}{\Delta x}\,i,\ \frac{|u|+\sqrt{gh}}{\Delta x}\,i\right]$$

volgt. Voor niet-constante roosterfuncties neemt men aan dat de eigenwaarden van J "ongeveer" in het grootste van de door de methode der constante coefficienten verkregen intervallen liggen.

2.2.4.3. Het Noordzee-probleem

Op dezelfde wijze als voor het drempel-probleem kan men afleiden dat de eigenwaarden bij het Noordzeeprobleem (2.2.3.10) "ongeveer" liggen in het grootste van de lokale intervallen (vergelijk FISCHER [1959])

$$(2.2.4.7)\quad \left[-\tfrac{1}{2}\lambda-\frac{\sqrt{2gh+(\omega^2-\frac{1}{4}\lambda^2)\Delta^2 x}}{\Delta x}\,i,\ \tfrac{1}{2}\lambda+\frac{\sqrt{2gh+(\omega^2-\frac{1}{4}\lambda^2)\Delta^2 x}}{\Delta x}\,i\right].$$

2.2.5. Numerieke integratie van de stelsels differentiaalvergelijkingen

Bij de keuze van een routine uit de bibliotheken ACCU, IMSL, NAG en NUMAL voor de integratie van de drie gestelde problemen zijn de voornaamste richtlijnen:

1. het betreft *grote* tot *zeer grote* stelsels
2. de gevraagde nauwkeurigheid is *niet meer dan 1%*.

Hiermee vallen in principe alle *impliciete* of *semi-impliciete* en alle *hoge orde* methoden af. Uit de documentatie van de genoemde programmatheken blijkt dan dat overblijven de in onderstaande tabel genoemde routines.

TABEL 1

Routines beschikbaar voor de integratie van de stelsels (2.2.3.8), (2.2.3.9) en (2.2.3.10)

Programmatheek	Routine	Geheugen	Geschikte problemen
IMSL	DASCRU = 2F	5n	Diffusie; Drempel
NAG	D02A $\begin{matrix}AA(F)\\BA(F)\end{matrix}$ = 6 AF	8n	Diffusie; Drempel
NUMAL	EXP.FIT.TAY. = 17A	3n	
	MOD.TAY. = 22A	2n	
	EFRK = 23A	2n	
	ARK(MAT) = 25A	3n	Diffusie; Drempel; Zee

Deze routines komen overeen met degenen die in de ochtendbijeenkomst genoemd werden (in de routine-kolom is de codering uit hoofdstuk 2.1 aan de routinenaam toegevoegd). We zullen deze routines afzonderlijk nader toelichten.

2.2.5.1. De routine DASCRU

DASCRU is gebaseerd op de *Merson-algorithme*, een 4^e orde Runge-Kuttamethode met ingebedde foutenformule. Hiervoor zijn 5 rechterlid-evaluaties per stap nodig; de algorithme is vrij optimaal geïmplementeerd en vraagt slechts ruimte voor 5n plaatsen (n is het aantal differentiaalvergelijkingen) plus nog wat werkruimte. Het diffusie- en het drempelprobleem met respectievelijk minder dan 100 en ongeveer 800 vergelijkingen kan zonder moeite geïntegreerd worden. Omdat DASCRU niet geschreven is voor stelsels afkomstig van partiele differentiaalvergelijkingen is de routine wel een wat dure integratiemethode voor problemen als het diffusie- en drempelprobleem; zo is de stapkeuze-strategie, gebaseerd op een referentie-oplossing, wat luxe voor ons bijzondere geval en zou een stapkeuze op grond van het stabiliteitsgebied van de algorithme veel tijd uitsparen. Ook de orde 4 is rijkelijk hoog voor een gevraagde nauwkeurigheid van 1%. In tabel 2.2.2 zijn wat numerieke resultaten opgenomen; deze zijn ontleend aan experimenten uitgevoerd door J. Petiet op de CYBER van SARA. Wat het Noordzeeprobleem met zijn ongeveer 10.000 vergelijkingen betreft, wij

hebben de voorkeur gegeven dit probleem met een meer op partiële differentiaalvergelijkingen gerichte routine te integreren (zie paragraaf 2.2.5.4).

2.2.5.2. De routine D02AAA(F)/BA(F)

Dit viertal routines (2 in FORTRAN en 2 in ALGOL 60) is ook gebaseerd op de *Merson-algorithme*. De FORTRAN-implementaties vragen een geheugenruimte van de orde 8n, de ALGOL 60-versies zouden zelfs 22n plaatsen vragen, althans volgens de betreffende documentatie. Voor het overige gelden dezelfde opmerkingen als gemaakt voor DASCRU.
Numerieke resultaten weer ontleend aan experimenten gedaan door J. Petiet vindt men in tabel 2.

2.2.5.3. De routines EXP.FIT.TAY, MOD.TAY en EFRK

Deze routines zijn weliswaar gebaseerd op algorithmen speciaal ontworpen voor grote stelsels, maar minder geschikt voor de beschouwde problemen. EXP.FIT.TAY. en MOD.TAY. omdat ze behalve de rechterlid-functie ook haar afgeleiden nodig heeft, en EFRK omdat deze procedures pas tot z'n recht komt bij eigenwaarden-spectra waarin de eigenwaarden in clusters gegroepeerd zijn.

2.2.5.4. De routine ARK

Deze routine is speciaal ontworpen voor de integratie van *niet-lineaire partiële differentiaalvergelijkingen*. Het geheugengebruik is verhoudingsgewijs gering, de orde van de methode is 1, 2 of 3 en kan door de gebruiker gekozen worden, en het stabiliteitsgedrag kan afgestemd worden op de te integreren vergelijking door het kiezen van het zogenaamde stabiliteitspolynoom en het opgeven van de spectrale radius van de Jacobiaan. Dit is dan tevens ook het nadeel van deze Adaptieve-Runge-Kutta-methode: *men moet het spectrum van de Jacobiaan onderzoeken en daarbij de coefficienten van een geschikt stabiliteitspolynoom (het array data [1:m]) opgeven.* Voor verdere details en literatuur-verwijzingen raadplege men de NUMAL-manual. Hier volstaan we met een tweetal voorbeelden van stabiliteitspolynomen:

(2.2.5.1) $1 + z + \frac{1}{2} z^2 + \frac{1}{16} z^3$ induceert een 2^e orde methode, geschikt voor *parabolische* vergelijkingen (*negatieve* eigenwaarden),

(2.2.5.2) $1 + z + \frac{1}{2} z^2 + \frac{1}{4} z^3$ induceert een 2^e orde methode, geschikt voor *hyperbolische* vergelijkingen (*imaginaire* eigenwaarden)

Het eerste polynoom is gebruikt voor de integratie van het diffusieprobleem, het tweede polynoom voor het drempelprobleem; resultaten van J. Petiet vindt men in tabel 2.

2.2.5.5. De routine ARKMAT

ARKMAT is een 2-dimensionale versie van ARK, dat wil zeggen de differentiaalvergelijking kan aangeboden worden in de vorm

$$\frac{dY}{dt} = F(t,Y),$$

waarin Y een (rechthoekige) matrix is en F een matrix-functie van t en Y is. Voor de integratie van stelsels afkomstig van 2-dimensionale partiële differentiaalvergelijkingen is zo'n matrix-versie handig, anders zouden de 2-dimensionale roosterfuncties in een één-dimensionaal array opgeslagen moeten worden, hetgeen een bron van vergissingen kan worden.

Met ARKMAT en het polynoom (2.2.5.2) zijn de Noordzeevergelijkingen (2.2.3.10) geïntegreerd door H.G.J.Rozenhart; de resultaten vindt men in tabel 3. Hierbij moet wel opgemerkt worden dat de integratie van omvangrijke problemen als het Noordzeeprobleem, het verantwoord maken "ad hoc" methoden te ontwikkelen die dan veel efficienter kunnen zijn dan ARKMAT. Anderzijds werden onderstaande resultaten binnen 2 weken verkregen, terwijl b.v. een "special purpose" methode zoals die van LEENDERTSE [1967] vele manmaanden heeft gevergd.

Literatuur

CARDENAS, A.F. & W.Y. KARPLUS [1970], *PDEL - a language for partial differential equations*, Comm. ACM 13, 184-191.

COHEN, J. & P. GROSSMAN [1975], *Compilation of linear partial differential equations into finite-difference programs*, BIT 15, 373-380.

FISCHER, G. [1959], *Ein numerisches Verfahren zur Errechnung von Windstau und Gezeiten in Randmeeren*, Tellus 11, 60-76.

HOUGHTON, D.D. & A. KASAHARA [1968], *Nonlinear shallow fluid flow over an isolated ridge*, Comm. Pure Appl. Math. 21, 1-28.

MORRIS, S.M. & W.E. SCHIESSER [1968], *SALEM - a programming system for the simulation of systems described by partial differential equations*, In Proc. AFIPS 1968 Fall Joint Computer Conf., 33, 353-357.

RICHTMYER, R.D. & K.W. MORTON [1967], *Difference Methods for Initial value problems*, 2nd ed., Interscience.

SINCOVEC, R.F. & N.K. MADSEN [1975], *Software for nonlinear partial differential equations*, ACM Transactions on Mathematical Software, 1, 232-260.

ZELLNER, M.G. [1970], *DDS - distributed systems simulator*, Ph.D. Diss. Lehigh U., Bethlehem, Pa. .

TABEL 2

Numerieke resultaten verkregen voor het diffusie- en het drempel-probleem

DASCRU	tijd	x = 0	x = 0.8	x = 0.959	x = 1.041	x = 1.2	x = 2.0	f.ev
diffusie probl.	0.63	1.00	0.84	0.62	0.51	0.32	0.03	1400
$\Delta Z = 0.1$	4.0	0.69	0.52	0.45	0.41	0.34	0.16	9690
$\alpha = -0.263$, $\beta = 0.291$								
D02A								
diffusie probl.	0.63	1.00	0.84	0.62	0.51	0.32	0.03	932
$\Delta Z = 0.1$	4.0	0.69	0.52	0.45	0.41	0.34	0.16	1178
$\alpha,\beta \rightarrow$ DASCRU								
ARK								
diffusie probl.	0.63	1.00	0.84	0.62	0.51	0.32	0.03	946
$\Delta Z = 0.1$	4.0	0.69	0.52	0.45	0.41	0.34	0.16	4371
$\alpha,\beta \rightarrow$ DASCRU								
DASCRU	tijd	x = -40	x = -20	x = 0	x = 20	x = 40	x = 60	
drempel probl.	0.5	22.1	21.5	19.7	18.2	19.9	18.9	510
$\Delta x = 1$	1.0	21.7	20.5	17.8	10.3	19.8	19.8	1060
		bij t = 1.5 een TIME LIMIT						
D02A	tijd	x = -40	x = - 20	x = 0	x = 20	x = 40	x = 60	
drempel probl.	0.5	22.09	21.5	19.7	18.2	19.9	18.9	250
$\Delta x = 1$	1.0	21.7	20.5	17.8	10.3	19.8	19.8	525
		bij t = 1.5 een foutmelding (D02AF WITH ERROR 1)						
ARK	tijd							
drempel probl.	0.5	22.1	21.5	19.7	18.2	19.9	18.9	154
$\Delta x = 1$	1.0	21.7	20.5	17.9	10.4	19.8	19.7	305
		bij t = 1.5 arithmetic overflow.						

TABEL 3

Numerieke resultaten verkregen voor het Noordzeeprobleem met ARKMAT en ΔX resp. 40, 20 en 10 km

tijd in uren	waterstanden zuidkust in m							waterstanden oostkust in m							
01	-01.42	-01.44	-01.45		-01.46	-01.48		-00.60	+00.16		+00.18	+00.20	+00.22	+00.22	+00.23
02	-04.18	-04.33	-04.49		-04.69	-04.76		-01.76	+00.77		+01.08	+01.29	+01.35	+01.37	+01.38
03	-06.08	-06.52	-06.90		-07.68	-07.66		-03.10	+01.15		+02.49	+03.32	+03.55	+03.59	+03.61
04	-06.97	-07.60	-07.79		-09.42	-08.79		-04.79	+00.22		+03.45	+05.47	+06.32	+06.38	+06.40
05	-08.07	-08.74	-07.91		-10.22	-08.50		-06.76	-02.17		+02.99	+06.47	+08.82	+09.02	+09.11
06	-09.68	-10.55	-08.34		-10.74	-08.17		-08.51	-04.87		+00.94	+05.52	+10.33	+10.97	+11.30
07	-11.13	-12.45	-09.37		-11.36	-08.68		-09.49	-06.46		-01.83	+02.91	+10.35	+11.87	+12.76
08	-12.39	-13.95	-10.73		-12.30	-09.83		-09.84	-06.68		-04.06	-00.08	+08.79	+11.49	+13.25
01	-02.25	-02.34	-02.43	-02.47	-02.52	-02.62	-02.64	+00.15	+00.38	+00.38	+00.39	+00.40	+00.41	+00.41	+00.42
02	-04.17	-04.54	-04.99	-05.25	-05.56	-06.23	-06.21	-00.60	+01.84	+02.04	+02.18	+02.22	+02.24	+02.24	+02.24
03	-05.83	-06.13	-06.53	-06.62	-07.04	-07.48	-07.01	-03.62	+02.12	+03.47	+04.38	+04.80	+04.83	+04.81	+04.78
04	-07.41	-07.84	-08.37	-08.11	-08.64	-08.44	-08.01	-05.49	-00.58	+02.45	+04.76	+06.94	+07.47	+07.44	+07.38
05	-09.01	-08.80	-08.85	-08.44	-09.07	-09.66	-09.50	-05.53	-02.97	-00.32	+02.34	+07.36	+10.07	+10.13	+10.08
06	-10.75	-10.15	-09.60	-09.03	-09.59	-10.18	-07.47	-06.88	-03.00	-01.95	-00.31	+05.01	+11.93	+12.53	+12.58
07	-12.06	-11.18	-10.62	-09.86	-10.54	-10.63	-09.86	-07.91	-03.80	-02.23	-00.79	+02.01	+11.59	+13.79	+14.47
08	-13.58	-12.86	-12.32	-11.22	-11.72	-11.52	-10.91	-07.74	-05.31	-03.20	-01.35	+01.12	+08.66	+12.85	+15.09
01	-02.22	-02.42	-02.66	-02.81	-02.97	-03.34	-03.54	+00.48	+00.64	+00.64	+00.64	+00.65	+00.65	+00.65	+00.65
02	-04.25	-04.59	-04.95	-05.14	-05.34	-05.55	-05.41	-01.15	+02.31	+02.50	+02.53	+02.53	+02.52	+02.50	+02.48
03	-05.95	-06.37	-06.80	-06.99	-07.10	-07.22	-07.46	-03.26	+01.72	+03.95	+04.81	+05.06	+05.05	+05.02	+04.98
04	-07.49	-07.76	-08.02	-08.07	-08.16	-08.54	-08.64	-04.89	-00.89	+01.52	+04.62	+07.54	+07.81	+07.77	+07.70
05	-09.15	-08.74	-08.55	-08.72	-08.95	-09.28	-09.38	-05.51	-01.75	+00.24	+01.70	+07.37	+10.56	+10.52	+10.44
06	-10.48	-09.72	-09.36	-09.32	-09.42	-09.85	-10.19	-06.75	-02.97	-01.40	+01.01	+04.14	+12.61	+13.02	+12.96
07	-11.83	-11.07	-10.51	-10.38	-10.23	-10.29	-10.53	-07.11	-03.93	-01.91	-00.61	+03.33	+11.53	+14.54	+15.08
08	-13.43	-12.63	-12.03	-11.75	-11.51	-11.30	-11.25	-07.58	-04.35	-03.04	-01.38	+01.51	+07.98	+12.67	+15.74

3. RANDWAARDEPROBLEMEN

3.1. Oplossen van tweepunts randwaardeproblemen

door P.W. Hemker
(Mathematisch Centrum)
en
J.P. Roos
(Corporate Research, Akzo Arnhem)

3.1.1. Inleiding

Voor tweepunts randwaardeproblemen bestaat weinig programmatuur die eenvoudig toegankelijk is. In de programmatheken ACCULIB en IMSL vinden we geen programma's die direkt in staat zijn een tweepunts randwaardeprobleem op te lossen. In de NAG-programmatheek vinden we 2 routines en in de NUMAL vinden we 4 routines voor bepaalde klassen van tweepunts randwaardeproblemen. Bovendien vinden we in NUMAL een routine voor het schatten van onbekende parameters, die ook voor tweepunts randwaardeproblemen gebruikt kan worden.

De oorzaak van dit gebrek aan algemene routines voor randwaardeproblemen kan wellicht gevonden worden in het feit dat programma's voor deze problemen praktisch altijd efficiënter worden wanneer ze zich beperken tot een deelklasse van problemen. Dit houdt in dat men snel geneigd zal zijn voor ieder probleem afzonderlijk een programma te schrijven dat juist dat éne probleem efficiënt oplost. We vinden dan ook, naast de routines die speciaal bestemd zijn voor randwaardeproblemen, in de meeste programmatheken een arsenaal van routines voor het oplossen van ijle lineaire stelsels. Deze routines maken het mogelijk om randwaardeproblemen op te lossen nadat de gebruiker ze zelf heeft gediscretiseerd. Voor de tweepunts randwaardeproblemen zijn hier vooral nuttig de routines die lineaire stelsels oplossen waarvan de matrix een bandstructuur heeft.
Deze noodzaak tot het schrijven van ad hoc programma's zal echter verdwijnen als er toch voldoend efficiënte programma's voor algemene problemen gemaakt worden. Naast de programmatuur die in de programmatheken beschikbaar is, zijn er op verschillende plaatsen programma's in voorbereiding. We willen hier in het bijzonder vermelden een programma van Bulirsch en Stoer, gebaseerd op de multiple-shooting techniek (de definitieve publicatie is voralsnog onbekend) en programma's door Keller en Pereyra die gebaseerd zijn op een discretiserings-methode.

3.1.2. Oplossings methoden

Vele studies zijn verricht op het gebied van het numeriek oplossen van tweepunts randwaardeproblemen en nog steeds worden nieuwe en betere algoritmen ontwikkeld. Hoewel het onmogelijk is hier een overzicht te geven van de bestaande theorieën en methoden, is het toch van belang de

belangrijkste principes kort uiteen te zetten.

Een tweepunts randwaardeprobleem bestaat uit een (stelsel) differentiaalvergelijking(en) die gedefinieërd is (zijn) op een interval [a,b]; bovendien zijn condities van de functie gegeven in a en b. Voor het numeriek oplossen van deze problemen onderscheiden we twee klassen van methoden: de *schietmethoden* en de *discretiseringsmethoden*. Voor de beide basisprincipes zijn veel varianten mogelijk.

Schietmethoden

Schietmethoden reduceren het probleem tot een serie beginwaardeproblemen. De differentiaalvergelijking wordt geschreven als een stelsel van n eerste orde vergelijkingen en aan de rand (zeg $x = a$) wordt een aantal beginwaarden gekozen die in overeenstemming zijn met de randcondities voor $x = a$. De ontbrekende beginvoorwaarden bij $x = a$ worden geïntroduceerd als onbekende parameters.

Met verschillende waarden voor deze parameters wordt het beginwaardeprobleem geïntegreerd van a naar b. Aan de hand van de verkregen oplossing in b worden de parameters, in een iteratief proces, zodanig bepaald dat het beginwaardeprobleem aan de condities van het randwaardeprobleem voldoet. Verschillende varianten op deze schietmethode zijn mogelijk:

1. integratie van het beginwaardeprobleem van b naar a i.p.v. van a naar b.
2. integratie van a naar r, $a < r < b$, en van b naar r. In deze variant worden aan beide randen onbekende parameters geïntroduceerd. Deze parameters worden nu zó bepaald dat de oplossingen op de intervallen (a,r) en (r,b) op elkaar aansluiten in $x = r$ (continuïteit van de oplossing).
3. multiple-shooting.
   Hierbij wordt het interval [a,b] verdeeld in meer deelintervallen

$$[x_{i-1}, x_i],\ a = x_0 < x_1 < \ldots < x_N = b.$$

   Op ieder deelinterval wordt een beginwaardeprobleem opgelost. De parameters voor de onbekende beginvoorwaarden worden nu zodanig bepaald dat de uiteindelijke oplossing continu is in alle punten x_i, $1 \leq i \leq N - 1$, en zodat aan de randvoorwaarden wordt voldaan.

De belangrijkste moeilijkheid bij het oplossen van tweepunts randwaardeproblemen met een schietmethode is de mogelijk grote instabiliteit van een probleem. We illustreren dit met het volgende, eenvoudige probleem

$$y'' - (p+q)y' + pq\,y = s, \qquad a \le x \le b,$$

$$y(a) = \alpha,\ y(b) = \beta.$$

Dit probleem is stabiel wanneer $pq < 0$; d.w.z. als $pq < 0$ dan veroorzaakt een kleine verstoring in de gegevens s, α of β ook een kleine storing in de oplossing y. Het beginwaardeprobleem dat wordt opgelost wanneer een schietmethode wordt toegepast heeft een Jacobiaan met eigenwaarden p en q wanneer het in voorwaartse richting wordt geïntegreerd of met eigenwaarden -p en -q als het in achterwaartse richting wordt geïntegreerd. Voor een stabiel tweepunts randwaardeprobleem verschillen p en q van teken. Het beginwaardeprobleem is instabiel. Als voor een eigenwaarde (zeg p) bovendien $|p|$ groot is, dan is het beginwaardeprobleem stijf ($p<0$) of sterk instabiel ($p>0$). In beide gevallen is het berekenen van de onbekende parameters een slecht geconditioneerd probleem.

Discreteriseringsmethoden

Discretiseringsmethoden gaan uit van een partitie van het interval [a,b], d.i. een (groot) aantal punten x_i zodat $a = x_0 < x_1 < \ldots < x_n = b$. De oplossing wordt nu berekend voor deze van te voren gegeven x-waarden. Hiertoe worden de differentiaaloperator en het rechterlid van de vergelijking gediscretiseerd. Voor discretisatie behoeft de vergelijking niet gereduceerd te worden tot een stelsel eerste orde vergelijkingen. De discretisering kan op vele verschillende manieren plaatsvinden. Het uiteindelijke resultaat is echter dat de differentiaalvergelijking en de randcondities worden teruggebracht tot een (groot) stelsel algebraïsche vergelijkingen. De oplossing van dit stelsel levert de waarden van de gevraagde functie op de te voren gegeven partitie.

De belangrijkste varianten van de discretiseringsmethoden zijn de differentiemethoden en de globale methoden.

Differentiemethoden zijn het eenvoudigst. Hierbij worden de differentiaaloperatoren direkt vervangen door differentieoperatoren. De differentiemethoden worden voornamelijk toegepast voor een lage orde van nauwkeurigheid en voor uniforme partities.

Globale methoden baseren de discretisering op de keuze van een eindig aantal basisfuncties $\{\phi_i\}$ in de lineaire ruimte van alle mogelijke oplossingsfuncties. Een numerieke oplossing

$$y_h(x) = \sum_{j=0}^{M} a_j \phi_j(x)$$

wordt nu bepaald zodat $y_h(x)$, aan de hand van zekere criteria, zo goed mogelijk aan de differentiaalvergelijking en aan de randcondities voldoet. Tot deze groep van methoden behoren o.a. de *collocatiemethoden*, de *Galerkin-methoden* en de *kleinste-kwadratenmethode*.
Globale methoden kunnen eenvoudig op een niet-uniforme partitie worden toegepast en de constructie van deze methoden met een hoge orde van nauwkeurigheid is eenvoudiger dan voor differentiemethoden.

3.1.3. Overzicht van de beschikbare programmatuur

In de programmatheken NAG en NUMAL zijn de volgende routines beschikbaar voor het oplossen van tweepunts randwaardeproblemen.

1. D02ADA/F (NAG)

Deze routine lost een tweepunts randwaardeprobleem op voor een stelsel gewone differentiaalvergelijkingen over een interval [a,b]

$$\frac{dy_i}{dx} = f_i(x,y_1,\ldots,y_N), \qquad i = 1,2,\ldots,N. \tag{3.1.3.1}$$

De procedure gebruikt de bovengenoemde variant no. 2 van de schietmethode. De gebruiker specificeert een getal r, $a < r < b$, waarna beginwaardeproblemen worden opgelost van a naar r en van b naar r. Naast de N vergelijkingen (3.1.3.1) moet de gebruiker N randwaarden opgeven, sommige voor $x = a$, de andere voor $x = b$. Bovendien moet de gebruiker een beginschatting geven voor de ontbrekende randwaarden van y.

De procedure levert na afloop de berekende waarden van y voor $x = a$ en $x = b$ en, als daarom gevraagd wordt, de berekende waarden van y op een equidistante partitie van [a,b]. De beginwaardeproblemen worden in D02ADA/F opgelost met een Runge-Kutta-Merson methode, de onbekende parameters worden bepaald met een vorm van Newton-iteratie.

2. D02AGA/F (NAG)

Deze routine heeft veel overeenkomst met D02ADA/F. Het randwaardeprobleem wordt weer opgelost met een schietmethode, variant 2 (met de

Runge-Kutta-Merson integratiemethode en Newton-iteratie). De routine is echter algemener in die zin, dat ook parameters die geen randwaarden zijn (eigenwaarden, coeffiënten in de vergelijking etc.) bepaald kunnen worden. Het randwaardeprobleem dat kan worden opgelost luidt

$$(3.1.3.2)\quad \begin{cases} \dfrac{dy_i}{dx} = f_i(x,y_1,\ldots,y_N,p_1,\ldots,p_M), & a < x < b, \\ a = a(p_1,\ldots,p_M),\ b = (p_1,\ldots,p_M), \\ y(a) = ya(p_1,\ldots,p_M),\ y(b) = yb(p_1,\ldots,p_M), & M \le N. \end{cases}$$

De functies f_i, a, b, ya en yb moeten door de gebruiker worden gespecificeerd. Ook het punt r, $a < r < b$, waar de oplossingen van de beginwaardeproblemen aan elkaar gepast moeten worden, kan worden opgegeven afhankelijk van $p_1,\ldots,p_M$. De procedure heeft een beginschatting voor $\vec{p}$ nodig en levert een verbeterde waarde van $\vec{p}$ af.

3. PEIDE (NUMAL)

Deze procedure berekent parameters $p_1,\ldots,p_M$ zodanig dat

$$(3.1.3.3)\quad \sum_{j=1}^{J} |y_i(x_j) - s_j|^2, \qquad J \ge M,$$

geminimaliseerd wordt. Hierin is $\{(x_j,s_j)\}_{j=1}^{J}$ een gegeven verzameling punten ("observaties") en y_i is een component van de oplossing van het beginwaardeprobleem

$$(3.1.3.4)\quad \begin{cases} \dfrac{dy_i}{dx} = f_i(x,y_1,y_2,\ldots,y_N,p_1,\ldots,p_M), & i = 1,\ldots,N \\ y(a) = ya(p_1,\ldots,p_M). \end{cases}$$

Zo bepaalt PEIDE parameters in de vergelijking en in de randvoorwaarden, zodanig dat de oplossing van de vergelijking een gegeven verzameling "observaties" zo dicht mogelijk passeert.

De routine gebruikt de multiple-shooting techniek waarbij punten x_j, $a<x_j<b$, gebruikt kunnen worden als partitie-punten. De beginwaardeproblemen worden opgelost met Gear's methode (geschikt voor stijve differ-

entiaalvergelijkingen). De minimalisering van het residu (3.1.3.3) gebeurt met de methode van Marquardt. Hierbij maakt de procedure gebruik van, door de gebruiker te specificeren, partiële afgeleiden:

$$\frac{\partial ya_i}{\partial p_j} \quad , \quad \frac{\partial f_i}{\partial y_k} \quad , \quad \frac{\partial f_i}{\partial p_j} \qquad i,k = 1,\ldots,N;\ j = 1,\ldots,M.$$

De procedure PEIDE kan gebruikt worden voor het oplossen van het tweepunts-randwaardeprobleem (3.1.3.1) door te nemen

$$\begin{aligned} & y_i(a) = ya(p_1,p_2,\ldots,p_M) \\ & x_j = b,\ s_j = y_{i(j)}(b),\ j = 1,2,\ldots,M. \end{aligned} \tag{3.1.3.5}$$

4. FEMLAG, FEMLAGSYM, FEMLAGSKEW (NUMAL)

Deze procedures lossen een probleem op van de vorm

$$\begin{aligned} & -(p(x)y'(x))' + r(x)\ y(x) = f(x), \qquad a \le x \le b, \text{ of} \\ & -y''(x) + q(x)y'(x) + r(x)\ y(x) = f(x), \qquad a \le x \le b, \end{aligned} \tag{3.1.3.6}$$

met de randvoorwaarden

$$\begin{aligned} & c_1y(a) + c_2y'(a) = c_3, \\ & c_4y(b) + c_5y'(b) = c_6. \end{aligned}$$

De oplossing wordt verkregen op een, van te voren gegeven, partitie $a = x_0 < x_1 < \ldots < x_N = b$. Er zijn geen beginschattingen voor parameters nodig. Deze procedures gebruiken een Galerkin-methode en zijn zeer nauwkeurig. De orde van nauwkeurigheid kan worden opgegeven en is $O(h^2)$, $O(h^4)$ of $O(h^6)$, waarin h de maaswijdte van de partitie is. De coëfficiënt p(x) moet positief zijn.

5. FEMHERMSYM (NUMAL).

Deze procedure lost een 4-de orde differentiaalvergelijking op van de vorm

(3.1.3.7) $(p(x)\ y''(x))'' - (q(x)\ y'(x))' + r(x)\ y(x) = f(x), \qquad a \le x \le b,$

met de randvoorwaarden

$$y(a) = c_1,\ y'(a) = c_2,$$
$$y(b) = c_3,\ y'(b) = c_4.$$

De waarden van y(x) en y'(x) op een van te voren gegeven partitie worden berekend met een Galerkin-methode. De orde van nauwkeurigheid is $O(h^4)$, $O(h^6)$ of $O(h^8)$.

3.1.4. Het gebruik van de programmatuur

In deze sectie lichten we het gebruik van de programmatuur toe aan de hand van een drietal problemen. Deze problemen zijn op verschillende manieren opgelost. De betreffende programmatuur en de bijbehorende numerieke resultaten kunnen worden gevonden in de bijlagen.

3.1.4.1. Een zuiver tweepunts randwaardeprobleem

We willen de hoek berekenen waaronder een projectiel afgeschoten moet worden om een bepaalde horizontale afstand af te leggen. De bewegingsvergelijkingen zijn voor x(t) en y(t)

$$(3.1.4.1)\quad \begin{cases} x'' = -\alpha v^2 \sin\phi \\ y'' = -g - \alpha v^2 \cos\phi \end{cases}$$

waarin

$v = \sqrt{(x')^2 + (y')^2}$ (de snelheid van het projectiel)
$\phi = \arctan(dy/dx)$ (de hoek met de horizontale richting)

zodat de baan van het projectiel beschreven wordt door het stelsel

$$(3.1.4.2)\quad \begin{cases} \dfrac{dy}{dx} = \tan\phi \\ \dfrac{dv}{dx} = g\tan(\phi)/v - \alpha v\cos(\phi) \\ \dfrac{d\phi}{dx} = -g/v^2 \end{cases}$$

Met de constanten g = 9.80665, α = 0.0007 en de randvoorwaarden

$$y = 0, v = 150 \text{ voor } x = 0 \quad ,$$
$$y = 0 \qquad \text{voor } x = 1500,$$

willen we $\phi(0)$ berekenen. We schatten de (overige) randwaarden als $\phi(0) = 1.15$; $\phi(1500) = -1.2$; $v(1500) = 135$.

3.1.4.2. Een parameter-schat probleem

Dit praktijkprobleem is ontleend aan de besliskunde. Om een curve $y(x)$ te berekenen die een optimale verkoopstrategie bepaalt, moeten we een parameter $p > 0$ bepalen, zodanig dat

$$y'(x) = 1 + p + \log(3)\exp(-n\ell y)$$

$$\times \left[y \sum_{j=0}^{n-2} \frac{1}{j!}(n\ell y)^j - \frac{1}{\ell} \sum_{j=0}^{n-1} \frac{1}{j!}(n\ell y)^j \right] \tag{3.1.4.3}$$

waarin $\ell = \ell(x) = \frac{1}{3x+0.01}$ en $n > 1$ een geheel getal.
Deze vergelijking geldt op het interval $[0,1]$; de randvoorwaarden zijn $y(0) = 0$, $y(1) = 2$. Omdat weliswaar $p > 0$, maar de orde van grootte p onbekend is, introduceren we de nieuwe parameter $P = \log(p)$. We bepalen de waarde van p voor $n = 10$ en $n = 20$.

3.1.4.3. Een niet-lineaire tweede orde vergelijking

We willen het volgende probleem oplossen met een schietmethode en met een globale methode

$$\begin{cases} y'' = e^y & \text{op } [0,1], \\ y(0) = y(1) = 0. \end{cases} \tag{3.1.4.4}$$

De analytische oplossing is bekend en luidt

$$y(x) = 2\ln\left(\frac{c}{\cos(e(x-0.5))}\right) + \ln 2,$$

(3.1.4.5)

$$c \approx 0.6680278475.$$

Voor de schietmethode schrijven we het probleem als

$$(3.1.4.6)\quad \begin{cases} y' = v \\ v' = e^{y} \end{cases},$$

$$y(0) = y(1) = 0.$$

Om het probleem op te lossen met een globale methode gebruiken we de procedure FEMLAG. Deze procedure is alleen geschikt voor lineaire problemen en daarom construeren we het iteratieve Newton-proces

$$(3.1.4.7)\quad -y''_{n+1} + [e^{y_n}]y_{n+1} = [e^{y_n}(y_n - 1)].$$

3.1.4.4. Een probleem dat niet opgelost kon worden met de beschikbare programmatuur

Vele tweepunts randwaardeproblemen leveren toch grote moeilijkheden op wanneer we alleen beschikken over de programmatuur die in sectie 3.1.3 genoemd is. Zo bleek het niet mogelijk om met een van de genoemde routines het volgende probleem op te lossen

$$\begin{cases} \varepsilon p y'' = g(y-z) \\ p z'' = -g(y-z) \end{cases} \qquad \text{op } [0,1]$$

(3.1.4.8)

$$y'(0) = 0,\ z(0) = 0,$$
$$y(1) = 1,\ z'(0) = 0,$$

waarin

$$g(x) = \exp(\tfrac{1}{3}\eta x) - \exp(-\tfrac{2}{3}\eta x)$$

$$(3.1.4.9)\quad \varepsilon = 0.143,\ p = 0.1743,\ \eta = 17.19.$$

Dit probleem heeft zo een duidelijk stijf karakter. Ook echter met de parameters (waarmee het probleem minder stijf is)

$$\varepsilon = 0.685,\ p = 4.028,\ \eta = 17.23$$

bleken de schietmethoden te falen. We wijten dit aan de sterke niet-lineariteit van het probleem die te grote moeilijkheden veroorzaakt voor de gebruikte integratiemethoden.

3.1.5. De algemene routines ontwikkeld door Pereyra en zijn medewerkers

Zoals in de inleiding tot dit hoofdstuk reeds werd aangegeven, bestaan er nauwelijks algemene routines voor het oplossen van tweepunts randwaardeproblemen.

Het is bekend, dat o.a. Keller en Pereyra, elk met hun medewerkers, aktief zijn op dit gebied. PEREYRA [1974] is echter de enige die een algemeen bruikbare routine (SYSSOL) gepubliceerd heeft; bovendien heeft hij reeds de beschikking over een experimentele, verbeterde versie (PASVAR), waarmee ook niet-uniforme roosters behandeld kunnen worden.

Bij zijn ontwikkeling van algemeen bruikbare routines voor het oplossen van randwaardeproblemen heeft Pereyra de bedoeling .. *to achieve the quality and high standards of the general purpose software available for initial value problems*. En ons inziens is hij daarin vrijwel geslaagd.

We zullen ons beperken tot tweepunts randwaardeproblemen van het type:

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

(3.1.5.1) $a \leq x \leq b$

$$A_1 y(a) + B_1 y(b) = C,$$

met

y en f: M-dimensionale vektorfunkties,
A en B: konstante (M×M) matrices,
C: konstante M-vektor.

We nemen aan dat de gezochte oplossing y van (3.1.5.1) bestaat en geïsoleerd is. Het gestelde probleem is dan:

Bereken een oplossing Y *die over het gehele interval* [a,b] *slechts met een bij voorbaat aangegeven tolerantie van de exakte oplossing* y *mag afwijken.*

De door Pereyra behandelde voorbeelden en enkele toepassingen van ons lijken de konklusie te wettigen dat de routines SYSSOL en vooral PASVAR het betrekkelijk automatisch en simpel oplossen van problemen als (3.1.5.1) mogelijk maken.

De struktuur van SYSSOL en PASVAR is globaal als volgt. De basis is een *discretiseringsmethode* (vgl. 3.1.2), waarbij gebruik gemaakt wordt van eindige differenties.

Dit betekent o.a. dat een bepaald rooster (partitie, mesh of grid) gegeven moet zijn. Bij een gegeven rooster leidt de discretisering tot een stelsel van eindig veel *niet-lineaire vergelijkingen*. De methode van Newton wordt toegepast om dit stelsel op te lossen, vgl. PEREYRA [1968, 1974, 1975], KELLER [1974]. Op een efficiënte wijze wordt door PEREYRA [1974] gebruik gemaakt van de bandstruktuur.

Na de oplossing van dit stelsel vergelijkingen is dus een *benadering* van de exakte oplossing op het gegeven rooster bekend. Maar hoe goed is deze benadering?

PEREYRA [1968] past nu de techniek van *Iterated Deferred Corrections (IDC)* toe om op het gegeven rooster een schatting van de fout te bepalen en vervolgens de benaderde oplossing (iteratief) te verbeteren.

N.B. Pereyra claimt dat IDC efficiënter is dan een techniek als Richardson extrapolatie, die b.v. door KELLER [1974] wordt toegepast.

Wanneer verder verbetering niet mogelijk of zinnig lijkt, wordt het *rooster verfijnd.*
In de gepubliceerde routine SYSSOL is het slechts mogelijk met een uniform rooster te werken; dit rooster wordt door halvering verfijnd.
In de versie PASVAR kan met een niet-uniform rooster worden gestart; verfijning van het rooster is i.h.a. niet uniform. Er wordt naar een zodanig rooster gestreefd, dat de geschatte fout van de oplossing uniform over het interval verdeeld is (dus b.v. veel roosterpunten bij grotere gradiënten): het begrip "equidistribution" wordt hier gehanteerd.

De overgangen tussen de verschillende genoemde fasen worden via een aantal strategieën geregeld. De "tuning" van deze overgangen is deels op basis van experimenten bepaald.

Afgezien van de gevallen waarin de subroutine op een abnormale wijze beëindigd wordt (b.v. onjuiste invoergegevens), leveren beide subroutines op "automatische" wijze een berekende oplossing, die in het algemeen aan de *door de gebruiker opgegeven nauwkeurigheid* voldoet.

De deklaratie van de subroutine SYSSOL is als volgt:

SUBROUTINE SYSSOL (M,N,A,B,C,A1,B1,TOL,DELEPS,X,Y,ABT,FF,JACOB,JERROR).

De parameters M,A,B,C,A1,B1,X en Y hebben dezelfde betekenis als in (3.1.5.1). N is het aantal roosterpunten (inclusief de beide randpunten) waarmee wordt begonnen. FF en JACOB zijn de door de gebruiker te definiëren subroutines voor de berekening van de rechterleden f van (3.1.5.1) en van de jacobiaan $(\partial f/\partial y)$ van f. TOL is de door de gebruiker op te geven absolute nauwkeurigheid.

Voor de betekenis van DELEPS en ABT en voor verdere details wordt verwezen naar PEREYRA [1974].

De subroutine is eenvoudig te gebruiken en snel uit de literatuur over te nemen.

Enkele voorbeelden

a. Een randwaardeprobleem uit de elektriciteitsleer.

$$\varepsilon p \frac{d^2y}{dt^2} = g(y-z) \ ,$$

$$p \frac{d^2z}{dt^2} = -g(y-z), \qquad 0 \le t \le 1,$$

$$t = 0 : \frac{dy}{dt} = 0, \ z = 0,$$

$$t = 1 : y = 1, \ \frac{dz}{dt} = 0,$$

$$g(x) = \exp(\tfrac{1}{3}\eta x) - \exp(-\tfrac{2}{3}\eta x),$$

$$\varepsilon = 0.143; \qquad p = 0.1743; \qquad \eta = 17.19.$$

De met PASVAR c.q. SYSSOL berekende oplossing is in de volgende tabel op genomen.

t	y	z	$\frac{dt}{dy}$	$\frac{dz}{dt}$
0.000	0.033	0.000	0.000	0.899
0.125	0.103	0.102	0.762	0.790
0.250	0.201	0.201	0.786	0.787
0.375	0.299	0.299	0.787	0.787
0.500	0.397	0.397	0.787	0.787
0.625	0.496	0.496	0.787	0.787
0.750	0.594	0.594	0.793	0.786
0.875	0.700	0.691	0.987	0.758
1.000	1.000	0.761	6.287	0.000

De grenslaagstruktuur bij t = 0, en vooral bij t = 1, is duidelijk te zien. Wanneer ε groter wordt gekozen gaan y en z meer uiteen; er ontstaat dan een minder uitgesproken grenslaagstruktuur.

Dit probleem is met behulp van (elementair) schieten, vgl. (3.1.2) niet of nauwelijks oplosbaar.

Dit probleem is zowel met SYSSOL als met PASVAR opgelost. Bij SYSSOL was een uniforme verfijning nodig tot 156 intervallen.
Bij PASVAR met gelijke nauwkeurigheidseis, werd een niet-uniforme verfijning toegepast tot 84 intervallen, waarbij 16 kleine op het deelinterval $[\frac{15}{16},1]$; dit geeft aan dat het rooster aangepast werd aan het karakter van de oplossing.

Een wat eenvoudiger probleem behoort bij de parameterwaarden $\varepsilon = 0.685$; $p = 4.028$; $\eta = 17.23$. De oplossing is dan:

t	y	z	$\frac{dy}{dt}$	$\frac{dz}{dt}$
0.000	0.348	0.000	0.000	1.001
0.125	0.365	0.114	0.247	0.832
0.250	0.406	0.211	0.401	0.726
0.375	0.464	0.296	0.518	0.646
0.500	0.536	0.373	0.629	0.570
0.625	0.621	0.439	0.740	0.494
0.750	0.722	0.495	0.882	0.397
0.875	0.844	0.537	1.085	0.258
1.000	1.000	0.555	1.461	0.000

Reeds bij 29 intervallen werd hier de gewenste nauwkeurigheid (meer dan de gegeven cijfers) bereikt.

b. Een probleem dat voortkomt uit de berekening van de warmte- en impuls-overdracht tussen een bewegende cylinder en een stilstaande omgeving

$$t\frac{d^3y}{dt^3} - \left(\frac{dt}{dt}\right)^2 + (y-1)\left(\frac{d^2y}{dt^2} - \frac{1}{t}\frac{dy}{dt}\right) = 0,$$

$$t\frac{d^2z}{dt^2} + (py+1)\frac{dz}{dt} = 0, \qquad t_0 \leq t \leq t_1,$$

$$t = t_0 : y = 0, \frac{dy}{dt} = t; z = 1,$$

$$t = t_1 : \frac{dy}{dt} = 0, z = 0,$$

$$t_0 = 0.005; t_1 = 20; p = 0.72.$$

De met PASVAR berekende oplossing is:

t	y	$\frac{dt}{dy}$	$\frac{d^2y}{dt^2}$	z	$\frac{dz}{dt}$
0.005	0.000	0.005	0.835	1.000	-28.056
0.010	0.000	0.009	0.716	0.898	-13.600
0.020	0.000	0.015	0.607	0.805	- 7.002
0.102	0.003	0.052	0.342	0.576	- 1.368
0.210	0.010	0.082	0.230	0.476	- 0.666
0.06	0.124	0.163	0.032	0.254	- 0.125
2.11	0.303	0.171	-0.007	0.167	- 0.057
3.16	0.477	0.158	-0.015	0.121	- 0.034
4.21	0.635	0.141	-0.016	0.092	- 0.023
5.27	0.775	0.125	-0.015	0.072	- 0.016
20.0	1.536	0.000	-0.006	0.000	- 0.001

Dit probleem kon ook met schieten opgelost worden.

Het is interessant te zien dat PASVAR een duidelijk niet-uniform rooster genereert. In het uiteindelijke rooster: $h_{min} = 0.0002125$ en $h_{max} = 1.05237$, dus een verhouding $h_{max}/h_{min} = 4950$.

Diskussie

Wanneer we eenvoud van gebruik als een der belangrijke criteria zien, dan blijken deze routines van Pereyra zeer duidelijk gunstig uit te steken boven alternatieve aanpakken als schieten. De routines lossen "automatisch" het gestelde probleem op en bovendien doen ze dat efficiënt.

Literatuur

LENTINI, M. & V. PEREYRA [1974], *A variable order finite difference method for nonlinear multipoint boundary value problems*, Math. Comp. 28, 981-1003.

KELLER, H.B. [1974], *Accurate difference methods for nonlinear two-point boundary value problems*, SIAM J. Numer. Anal. 11, 305-320.

PEREYRA, V. [1968], *Iterated deferred corrections for nonlinear boundary value problems*, Numer. Math. 11, 111-125.

PEREYRA, V. & E.G. SEWELL [1975], *Mesh selection for discrete solution of boundary problems in ordinary differential equations*, Numer. Math. 23, 261-268.

Bijlagen

```
PROBLEEM 3.1.4.1
M.B.V. PEIDE (NUMAL)


"BEGIN" "INTEGER" M, NN, NOBS, NBP;
  "ARRAY" PAR [1 : 1], RES [1 : 1], JTJINV [1 : 1, 1 : 1], IN [0 : 6],
          OUT [1 : 8]; "INTEGER" "ARRAY" BP [0 : 1];
  "REAL" TIME, FA, G, D;

  "PROCEDURE" PEIDE (N, M, NO, NB, P, R, BP, J, I, O, D, JDY, JDP, CY,
                     DA, MO); "CODE" 34444;
  "PROCEDURE" COMMUNICATION (P, F, M, N, NO, NP, PA, R, BP, J, I, O, W,
                             NI); "CODE" 34445;

  "BOOLEAN" "PROCEDURE" JAC DFDP (PAR, Y, X, FP);
  "REAL" X; "ARRAY" PAR, Y, FP;
  "BEGIN" FP [1, 1]:= FP [2, 1]:= FP [3, 1]:= 0;
    JAC DFDP:= "TRUE"
  "END" JAC DFDP;

  "PROCEDURE" DATA (NOBS, TOBS, OBS, COBS);
  "VALUE" NOBS; "INTEGER" NOBS; "ARRAY" TOBS, OBS, COBS;
  "BEGIN"
    TOBS [0]:= 0; TOBS [1]:= 1500; COBS [1]:= 1; OBS [1]:= 0
  "END" DATA;

  "PROCECURE" CALL YSTART (PAR, Y, YMAX);
  "ARRAY" PAR, Y, YMAX;
  "BEGIN" Y [1]:= 0; Y [2]:= 150; Y [3]:= PAR [1];
    YMAX [1]:= 1000; YMAX [2]:= 150; YMAX [3]:= 1;
    Y [2]:= 1
  "END" CALL YSTART;

  "BOOLEAN" "PROCEDURE" DERIV (PAR, Y, X, DF);
  "REAL" X; "ARRAY" PAR, Y, DF;
  "BEGIN" "REAL" C, S, V;
    C:= COS (Y [3]);
    S:= SIN (Y [3]);
    V:= Y [2];
    DF [1]:= S / C;
    DF [2]:= -G * S / (C * V) - D * V / C;
    DF [3]:= -G / (V * V);
    DERIV:= C > 0
  "END" DERIV;

  "BOOLEAN" "PROCEDURE" JAC DFDY (PAR, Y, X, FY);
  "REAL" X; "ARRAY" PAR, Y, FY;
  "BEGIN" "INTEGER" I, J; "REAL" C, S, V;
    C:= COS (Y [3]);
    S:= SIN (Y [3]);
    V:= Y [2];
    "FOR" I:= 1 "STEP" 1 "UNTIL" 3 "DO"
    "FOR" J:= 1 "STEP" 1 "UNTIL" 3 "DO" FY [I, J]:= 0;
```

```
    FY [1, 3]:= 1 / (C * C);
    FY [2, 2]:= (G * S / (V * V) - D) / C;
    FY [2, 3]:= (-G / V - D * V * S) / (C * C);
    FY [3, 2]:= 2 * G / (V * V * V);
    JAC DFDY:= C > 0
  "END" JAC DFDY;

  "PROCEDURE" MONITOR (POST, NCOL, NROW, PAR, RES, WEIGHT, NIS);
  "VALUE" POST, NCOL, NROW, WEIGHT, NIS;
  "INTEGER" POST, NCOL, NROW, WEIGHT, NIS; "ARRAY" PAR, RES;
  OUTPUT (61, "("/5ZD, 5B, N/")", POST, PAR [1]);

  G:= 9.80665; D:= 0.00007;
  PAR [1]:= 1.15; NBP:= 0;
  IN [3]:= "-6; IN [4]:= "-6; IN [5]:= 50; IN [6]:= "-10;
  IN [1]:= "-4; IN [2]:= "-6; FA:= 0; IN [0]:= "-14;
  NN:= 3; M:= NOBS:= 1; BP [0]:= BP [1]:= 0;
  TIME:= CLOCK;
  COMMUNICATION (1, FA, NN, M, NOBS, NBP, PAR, RES, BP, JTJINV, IN,
                 OUT, 0, 0);

  PEIDE(NN, 1, 1, NBP, PAR, RES, BP, JTJINV, IN, OUT, DERIV, JAC DFDY,
        JAC DFDP, CALL YSTART, DATA, MONITOR);

  TIME:= CLOCK - TIME;
  COMMUNICATION (2, FA, NN, M, NOBS, NBP, PAR, RES, BP, JTJINV, IN,
                 OUT, 0, 0);
  OUTPUT (61, "("3/5B "("THE CALCULATION IN PEIDE CONSUMED")",
          BZZD.DDBB, "("SECONDS")",*")", TIME)
"END"
```

STARTING VALUES OF THE PARAMETERS
+,1150000" +1

NUMBER OF EQUATIONS : 3
NUMBER OF OBSERVATIONS: 1

MACHINE PRECISION	$+,1"-13
RELATIVE LOCAL ERROR BOUND FOR INTEGRATION	$+,1" -5
RELATIVE TOLERANCE FOR RESIDUE	$+,10" -5
ABSOLUTE TOLERANCE FOR RESIDUE	$+,10" -5
MAXIMUM NUMBER OF INTEGRATIONS TO PERFORM	$ 50
RELATIVE STARTING VALUE OF LAMBDA	$+,10" -9
RELATIVE MINIMAL STEPLENGTH	$+,10" -3

THERE ARE NO BREAK-POINTS

THE ALPHA-POINT OF THE F-DISTIBUTION : 0,00

1	+1,1500000000000"+000
2	+1,1500000000000"+000
1	+1,1500000000000"+000
1	+1,1548127086472"+000
1	+1,1549991219896"+000
1	+1,1550107739228"+000
1	+1,1550115120783"+000
1	+1,1550115588797"+000
1	+1,1550115618472"+000
1	+1,1550115620353"+000

```
NORMAL TERMINATION OF THE PROCESS

LAST INTEGRATION WAS PERFORMED WITHOUT BREAK-POINTS


EUCL. NORM OF THE LAST RESIDUAL VECTOR ;.8609400" -7
EUCL. NORM OF THE FIRST RESIDUAL VECTOR;.3300138" +2
NUMBER OF INTEGRATIONS PERFORMED       ;  8
LAST IMPROVEMENT OF THE EUCLIDEAN NORM ;.1275210" -5
CONDITON NUMBER OF  J'*J               ;.1000000" +1
LOCAL ERROR BOUND WAS EXCEEDED (MAXIM.);  0
```

THE LAST RESIDUAL VECTOR

I	RES[I]
1	+.8609" -7

THE CALCULATION IN PEIDE CONSUMED 43.77 SECONDS

```
C     PROBLEEM 3.1.4.1
C     M.B.V. D02ADF(NAG)
C
      PROGRAM MASTER (OUTPUT, TAPE 6 = OUTPUT)
      EXTERNAL DERIV, OUT
      DIMENSION U (3, 2), V (3, 2), E (3), Y (6, 3), G (3, 3), INT (3),
     1WSP (3, 5), WS (3), YO (3), EE (3), AA (3), BB (3), CC (3), DD (3)
      U (1, 1) = 0.0
      V (1, 1) = 0.0
      U (1, 2) = 0.0
      V (1, 2) = 0.0
      U (2, 1) = 150.0
      V (2, 1) = 0.0
      U (2, 2) = 135.0
      V (2, 2) = 1.0
      U (3, 1) = 1.15
      V (3, 1) = 1.0
      U (3, 2) = -1.2
      V (3, 2) = 1.0
      R = 750
      X = 0.0
      X1 = 1500.0
      E (1) = 1E-2
      E (2) = 1E-2
      E (3) = 5E-5
      H = 0.01
      N = 3
      M1 = 6
      M = 1
      WRITE (6, 98)
98    FORMAT (28H1   CURRENT PARAMETER VALUES,
     1 4X, 23HERRORS IN Y(I) AT X = R, 18X, 17HSUM (ERRORS ** 2))
      CALL D02ADF (U, V, E, Y, H, X, X1, R, N, M1, DERIV, OUT, G,
     1 INT, WSP, WS, YO, EE, AA, BB, CC, DD, M)
      IF (M .GT. 0) GOTO 3
      WRITE (6, 99)
99    FORMAT (16H0 FINAL SOLUTION)
      DO 2 I = 1, 6
      WRITE (6, 100) I, (Y (I, J), J = 1, 3)
100   FORMAT (1H, I2, 3F10.4)
2     CONTINUE
3     WRITE (6, 101) M
101   FORMAT (13H ERROR NUMBER, I3)
10    CONTINUE
      STOP
      END
```

```
      SUBROUTINE DERIV (G, Z, X)
      DIMENSION G (3), Z (3)
      GG = 9,80665
      D = 0,00007
      G (1) = SIN (Z (3)) / COS (Z (3))
      G (2) = -GG * G (1) / Z (2) - D * Z (2) / COS (Z (3))
      G (3) = -GG / (Z (2) * Z (2))
      RETURN
      END



      SUBROUTINE OUT (A, B, F, R, N)
      DIMENSION A (N, 2), B (N, 2), F (N), W (3)
      J = 1
      DO 3 K = 1, 2
      DO 2 I = 1, N
      IF (B (I, K) .EQ. 0.0) GOTO 2
      W (J) = A (I, K)
      J = J + 1
2     CONTINUE
3     CONTINUE
      WRITE (6, 100) (W (J), J = 1, 3), (F (I), I = 1, 3), R
100   FORMAT (1H0, 3F9.4, 3E13.4, E15.4)
      RETURN
      END
```

```
   CURRENT PARAMETER VALUES     ERRORS IN Y[I] AT X = R                    SUM (ERRORS ** 2)

   1.1500 135.0000  -1.2000    -.2927E+02   -.2037E+01    .1334E+00       .8609E+03

   1.1549 137.4487  -1.1905    -.5354E+00   -.6099E-01    .2766E-02       .2904E+00

   1.1551 137.5017  -1.1902     .9342E-01    .1467E-01    .3008E-03       .8942E-02

   1.1550 137.5021  -1.1902     .2198E-02   -.3529E-03   -.3118E-04       .4957E-05

 FINAL SOLUTION
 1     0.0000  150.0000     1.1550
 2   555.2828  100.8934      .9588
 3   848.4029   63.4423      .4800
 4   863.1451   59.9158     -.4053
 5   586.3348   93.1578     -.9592
 6     0.0000  137.5021    -1.1902
ERROR NUMBER  0
```

```
PROBLEEM 3.1.4.2
M.B.V. PEIDE (NUMAL)

"BEGIN" "INTEGER" I, J, M, NN, N, NOBS, NBP;
  "ARRAY" PAR [1 : 1], RES [1 : 1], JTJINV [1 : 1, 1 : 1], IN [0 : 6],
          OUT [1 : 8]; "INTEGER" "ARRAY" BP [0 : 1], FAC [0 : 25];
  "REAL" TIME, FA;

  "PROCEDURE" PEIDE (N, M, NO, NB, P, R, BP, J, I, O, D, JDY, JDP, CY,
                     DA, MO); "CODE" 34444;
  "PROCEDURE" COMMUNICATION (P, F, M, N, NO, NP, PA, R, BP, J, I, O, W,
                             NI); "CODE" 34445;

  "REAL" "PROCEDURE" SUM (I, L, U, T); "VALUE" U;
  "INTEGER" I, L, U; "REAL" T;
  "BEGIN" "REAL" S; S:= 0;
    "FOR" I:= L "STEP" 1 "UNTIL" U "DO"
    S:= S + T; SUM:= S
  "END" SUM;

  "BOOLEAN" "PROCEDURE" JAC DFDP (PAR, Y, X, FP);
  "REAL" X; "ARRAY" PAR, Y, FP;
  "BEGIN" FP [1, 1]:= EXP (PAR [1]);
    JAC DFDP:= "TRUE"
  "END" JAC DFDP;

  "PROCEDURE" DATA (NOBS, TOBS, OBS, COBS);
  "VALUE" NOBS; "INTEGER" NOBS; "ARRAY" TOBS, OBS, COBS;
  "BEGIN"
    TOBS [0]:= 0; TOBS [1]:= 1; COBS [1]:= 1; OBS [1]:= 2
  "END" DATA;

  "PROCECURE" CALL YSTART (PAR, Y, YMAX);
  "ARRAY" PAR, Y, YMAX;
  "BEGIN" Y [1]:= 0; YMAX [1]:= 4
  "END" CALL YSTART;

  "BOOLEAN" "PROCEDURE" DERIV (PAR, Y, X, DF);
  "REAL" X; "ARRAY" PAR, Y, DF;
  "BEGIN" "REAL" Y1, NLY, EMNLY;
    Y1:= Y [1]; NLY:= N * L (X) * Y1;
    EMNLY:= EXP (-NLY);
    DF [1]:= 1 + EXP (PAR [1])
    * LN (3) * (Y [1] * EMNLY * SUM (J, 0, N-2, NLY ** J / FAC [J])
    - EMNLY * SUM (J, 0, N-1, NLY ** J / FAC [J]) / L (X));
    DERIV:= "TRUE"
  "END" DERIV;

  "BOOLEAN" "PROCEDURE" JAC DFDY (PAR, Y, X, FY);
  "REAL" X; "ARRAY" PAR, Y, FY;
  "BEGIN" "REAL" Y1, NLY, EMNLY;
    Y1:= Y [1]; NLY:= N * L (X) * Y1;
    EMNLY:= EXP (-NLY);
    FY [1, 1]:= LN (3) * EMNLY * SUM (J, 0, N-1, NLY ** J / FAC [J]);
    JAC DFDY:= "TRUE"
  "END" JAC DFDY;
```

```
"PROCEDURE" MONITOR (POST, NCOL, NROW, PAR, RES, WEIGHT, NIS);
"VALUE" POST, NCOL, NROW, WEIGHT, NIS;
"INTEGER" POST, NCOL, NROW, WEIGHT, NIS; "ARRAY" PAR, RES;
OUTPUT (61, "("/5ZD,5B,N/")", POST, PAR [1]);

"REAL" "PROCEDURE" L (X); "VALUE" X; "REAL" X;
L:= 1 / (3 * X + 0.01);

FAC [0]:= M:= 1;
"FOR" I:=1 "STEP" 1 "UNTIL" 25 "DO" FAC [I]:= M:= M * I;

N:= 10;
PAR [1]:= 0; NBP:= 0;
IN [3]:= "-7; IN [4]:= "-7; IN [5]:= 50; IN [6]:= "-3;
IN [1]:= "-4; IN [2]:= "-7; FA:= 0; IN [0]:= "-14;
NN:= M:= NOBS:= 1; BP [0]:= BP [1]:= 0;
TIME:= CLOCK;
COMMUNICATION (1, FA, NN, M, NOBS, NBP, PAR, RES, BP, JTJINV, IN,
               OUT, 0, 0);

PEIDE(1, 1, 1, NBP, PAR, RES, BP, JTJINV, IN, OUT, DERIV, JAC DFDY,
      JAC DFDP, CALL YSTART, DATA, MONITOR);

TIME:= CLOCK - TIME;
COMMUNICATION (2, FA, NN, M, NOBS, NBP, PAR, RES, BP, JTJINV, IN,
               OUT, 0, 0);
OUTPUT (61, "("3/5B "("THE CALCULATION IN PEIDE CONSUMED")",
        BZZD.DDBB, "("SECONDS")",*")", TIME)
"END"
```

STARTING VALUES OF THE PARAMETERS
+.0000000" +0

NUMBER OF EQUATIONS : 1
NUMBER OF OBSERVATIONS: 1

MACHINE PRECISION	:+.1"-13
RELATIVE LOCAL ERROR BOUND FOR INTEGRATION	:+.1" -6
RELATIVE TOLERANCE FOR RESIDUE	:+.10" -6
ABSOLUTE TOLERANCE FOR RESIDUE	:+.10" -6
MAXIMUM NUMBER OF INTEGRATIONS TO PERFORM	:50
RELATIVE STARTING VALUE OF LAMBDA	:+.10" -2
RELATIVE MINIMAL STEPLENGTH	:+.10" -3

THERE ARE NO BREAK-POINTS

THE ALPHA-POINT OF THE F-DISTIBUTION : 0.00

1	+0.0000000000000"+000
2	+0.0000000000000"+000
1	+0.0000000000000"+000
1	+4.0853574590190"-001
1	+4.0522950596640"-001
1	+4.0486250266894"-001
1	+4.0482209170912"-001
1	+4.0481764823811"-001
1	+4.0481715984346"-001
1	+4.0481710617162"-001

```
NORMAL TERMINATION OF THE PROCESS

LAST INTEGRATION WAS PERFORMED WITHOUT BREAK-POINTS


EUCL. NORM OF THE LAST RESIDUAL VECTOR ;.1590742" -7
EUCL. NORM OF THE FIRST RESIDUAL VECTOR;.8490653" +0
NUMBER OF INTEGRATIONS PERFORMED       ;  8
LAST IMPROVEMENT OF THE EUCLIDEAN NORM ;.1288565" -6
CONDITON NUMBER OF  J'*J               ;.1000000" +1
LOCAL ERROR BOUND WAS EXCEEDED (MAXIM.);  0
```

THE LAST RESIDUAL VECTOR

I	RES[I]
1	+.1591" -7

THE CALCULATION IN PEIDE CONSUMED 15.31 SECONDS

```
C     PROBLEEM 3.1.4.2
C     M.B.V. D02AGF (NAG)
C
      PROGRAM MASTER2 (OUTPUT, TAPE 6 = OUTPUT)
      REAL MAT
      DIMENSION PARAM (1), PARERR (1), ERROR (1), C (2, 1),  MAT (1, 1),
     1 COPY (1, 1), WSPACE (1, 9), G (1), G1 (1)
      EXTERNAL AUX,PRSOL,RNAUX,BCAUX
      PARAM(1)=0.0
      N1=1
      N=1
      H=0.00001
      ERROR(1)=1.0E-7
      PARERR(1)=1.0E-7
      M1=2
      IFAIL=1
      CALL D02AGF (H,ERROR,PARERR,PARAM,C,N,N1,M1,AUX,BCAUX,
     1 RNAUX,PRSOL,MAT,COPY,WSPACE,G,G1,IFAIL)
      WRITE(6,9002)IFAIL
9002  FORMAT(1H0,7HIFAIL= ,I3/17H0FINAL PARAMETERS)
      WRITE(6,9003)(PARAM(I),I=1,N1)
9003  FORMAT(1H ,3E16.8)
      WRITE(6,9004)
9004  FORMAT(1H0,14HFINAL SOLUTION/26H X-VALUE AND COMPONENTS OF,
     19H SOLUTION)
      CALL RNAUX(X,X1,R,PARAM)
      H = X1 - X
      DO 1 I = 1, 2
      DUM=X+FLOAT(I-1)*H
      WRITE(6,9005)DUM,(C(I,J),J=1,N)
9005  FORMAT(1H ,F10.3,3E15.6)
1     CONTINUE
      STOP
      END


      SUBROUTINE PRSOL(PARAM,RESID,N1,ERR)
      DIMENSION PARAM(N1),ERR(N1)
      WRITE(6,9000)(PARAM(I),ERR(I))
9000  FORMAT(12H PARAMETER ,E13.7,/,12H RESIDU    ,E11.3,//)
      RETURN
      END
```

```
      SUBROUTINE AUX(F,Y,X,PARAM)
      REAL L, NLY
      DIMENSION F (1), Y (1), PARAM (1)
      N = 10
      L = 1 / (3 * X + 0.01)
      YY = Y (1)
      NLY = N * L * YY
      JFAC = 1
      SOM = 0.0
      NN = N - 2
      DO 1 J = 1, NN
      JFAC = JFAC * J
      SOM = SOM + NLY ** J / JFAC
    1 CONTINUE
      SOM1 = SOM + 1.0
      SOM2 = SOM1 + NLY ** (N - 1) / (JFAC * (N - 1))
      F (1) = 1.0 + EXP (PARAM (1)) + ALOG (3.0) * EXP (-NLY) *
     1 (YY * SOM1 - SOM2 / L)
      RETURN
      END

      SUBROUTINE RNAUX(X,X1,R,PARAM)
      DIMENSION PARAM(1)
      X = 0.0
      X1 = 1.0
      R = 1.0
      RETURN
      END

      SUBROUTINE BCAUX(G,G1,PARAM)
      DIMENSION G(1),G1(1),PARAM(1)
      G (1) = 0.0
      G1 (1) = 2.0
      RETURN
      END
```

```
PARAMETER :0.
RESIDU    :   .849E+00

PARAMETER : .4779490E+00
RESIDU    :  -.179E+00

PARAMETER : .3771742E+00
RESIDU    :   .658E-01

PARAMETER : .4050347E+00
RESIDU    :  -.528E-03

PARAMETER : .4048113E+00
RESIDU    :   .838E-05

PARAMETER : .4048149E+00
RESIDU    :  -.133E-06


IFAIL=   0

FINAL PARAMETERS
   .40481483E+00

FINAL SOLUTION
X-VALUE AND COMPONENTS OF SOLUTION
     0.000    0.
     1.000     .200000E+01
```

```
PROBLEEM 3.1.4.3
M.B.V. PEIDE (NUMAL)


"BEGIN" "INTEGER" M, NN, NOBS, NBP;
  "ARRAY" PAR [1 : 1], RES [1 : 1], JTJINV [1 : 1, 1 : 1], IN [0 : 6],
          OUT [1 : 8]; "INTEGER" "ARRAY" BP [0 : 1];
  "REAL" TIME, FA;

  "PROCEDURE" PEIDE (N, M, NO, NB, P, R, BP, J, I, O, D, JDY, JDP, CY,
                     DA, MO); "CODE" 34444;
  "PROCEDURE" COMMUNICATION (P, F, M, N, NO, NP, PA, R, BP, J, I, O, W,
                             NI); "CODE" 34445;

  "BOOLEAN" "PROCEDURE" JAC DFDP (PAR, Y, X, FP);
  "REAL" X; "ARRAY" PAR, Y, FP;
  "BEGIN" FP [1, 1]:= FP [2, 1]:= 0;
    JAC DFDP:= "TRUE"
  "END" JAC DFDP;

  "PROCEDURE" DATA (NOBS, TOBS, OBS, COBS);
  "VALUE" NOBS; "INTEGER" NOBS; "ARRAY" TOBS, OBS, COBS;
  "BEGIN"
    TOBS [0]:= 0; TOBS [1]:= 1; COBS [1]:= 1; OBS [1]:= 0
  "END" DATA;

  "PROCECURE" CALL YSTART (PAR, Y, YMAX);
  "ARRAY" PAR, Y, YMAX;
  "BEGIN" Y [1]:= 0; Y [2]:= PAR [1];
    YMAX [1]:= YMAX [2]:= 1;
    Y [14]:= 1
  "END" CALL YSTART;

  "BOOLEAN" "PROCEDURE" DERIV (PAR, Y, X, DF);
  "REAL" X; "ARRAY" PAR, Y, DF;
  "BEGIN"
    DF [1]:= Y [2];
    DF [2]:= EXP (Y [1]);
    DERIV:= "TRUE"
  "END" DERIV;

  "BOOLEAN" "PROCEDURE" JAC DFDY (PAR, Y, X, FY);
  "REAL" X; "ARRAY" PAR, Y, FY;
  "BEGIN"
    FY [1, 1]:= FY [2, 2]:= 0;
    FY [1, 2]:= 1; FY [2, 1]:= EXP (Y [1]);
    JAC DFDY:= "TRUE"
  "END" JAC DFDY;

  "PROCEDURE" MONITOR (POST, NCOL, NROW, PAR, RES, WEIGHT, NIS);
  "VALUE" POST, NCOL, NROW, WEIGHT, NIS;
  "INTEGER" POST, NCOL, NROW, WEIGHT, NIS; "ARRAY" PAR, RES;
  OUTPUT (61, "("/5ZD, 5B, N/")", POST, PAR [1]);

  PAR [1]:= 1; NBP:= 0;
  IN [3]:= "=8; IN [4]:= "=8; IN [5]:= 50; IN [6]:= "=10;
  IN [1]:= "=4; IN [2]:= "=8; FA:= 0; IN [0]:= "=14;
  NN:= 2; M:= NOBS:= 1; BP [0]:= BP [1]:= 0;
  TIME:= CLOCK;
```

```
   COMMUNICATION (1, FA, NN, M, NOBS, NBP, PAR, RES, BP, JTJINV, IN,
                  OUT, 0, 0);

   PEIDE(NN, 1, 1, NBP, PAR, RES, BP, JTJINV, IN, OUT, DERIV, JAC DFDY,
         JAC DFDP, CALL YSTART, DATA, MONITOR);

   TIME:= CLOCK - TIME;
   COMMUNICATION (2, FA, NN, M, NOBS, NBP, PAR, RES, BP, JTJINV, IN,
                  OUT, 0, 0);
   OUTPUT (61, "("3/5B "("THE CALCULATION IN PEIDE CONSUMED")",
           BZZD.DDBB, "("SECONDS")",*")", TIME)
"END"
```

STARTING VALUES OF THE PARAMETERS
+,1000000" +1

NUMBER OF EQUATIONS : 2
NUMBER OF OBSERVATIONS: 1

MACHINE PRECISION	:+,1"-13
RELATIVE LOCAL ERROR BOUND FOR INTEGRATION	:+,1" -7
RELATIVE TOLERANCE FOR RESIDUE	:+,10" -7
ABSOLUTE TOLERANCE FOR RESIDUE	:+,10" -7
MAXIMUM NUMBER OF INTEGRATIONS TO PERFORM	: 50
RELATIVE STARTING VALUE OF LAMBDA	:+,10" -9
RELATIVE MINIMAL STEPLENGTH	:+,10" -3

THERE ARE NO BREAK-POINTS

THE ALPHA-POINT OF THE F-DISTIBUTION : 0,00

1	+1,0000000000000"+000
2	+1,0000000000000"+000
1	+1,0000000000000"+000
1	-2,5397516194423"-001
1	-4,5315780618951"-001
1	-4,6307915408640"-001
1	-4,6360333442760"-001
1	-4,6363109710557"-001
1	-4,6363256767096"-001
1	-4,6363264556597"-001

```
NORMAL TERMINATION OF THE PROCESS

LAST INTEGRATION WAS PERFORMED WITHOUT BREAK-POINTS


EUCL. NORM OF THE LAST RESIDUAL VECTOR ;.5051082" -8
EUCL. NORM OF THE FIRST RESIDUAL VECTOR;.1841818" +1
NUMBER OF INTEGRATIONS PERFORMED       ;  8
LAST IMPROVEMENT OF THE EUCLIDEAN NORM ;.9030730" -7
CONDITON NUMBER OF  J'*J               ;.1000000" +1
LOCAL ERROR BOUND WAS EXCEEDED (MAXIM.);  0
```

THE LAST RESIDUAL VECTOR

I	RES[I]
1	+.5051" -8

THE CALCULATION IN PEIDE CONSUMED 18.96 SECONDS

```
PROBLEEM 3.1.4.3
M.B.V. FEMLAG (NUMAL)


"BEGIN" "INTEGER" N, III; "REAL" XXX, YYY, C;
  "ARRAY" E [1 : 6];

  "PROCEDURE" FEMLAG (X, Y, N, R, F, ORDE, E); "CODE" 33301;

  "FOR" N:= 5, 10, 20 "DO"
  "BEGIN" "INTEGER" ORDE, I, IT; "REAL" NORM;
    "ARRAY" XX, YY, YN [0 : N];

    "REAL" "PROCEDURE" Y (X); "VALUE" X; "REAL" X;
    "IF" X = XXX "THEN" Y:= YYY "ELSE"
    "BEGIN" "INTEGER" I; "REAL" XXI, YYI;
      "IF" X < XX [III] "THEN" I:= 0 "ELSE" I:= III;
      "FOR" I:= I "WHILE" X > XX [I + 1] "DO" I:= I + 1;
      XXX:= X; III:= I; XXI:= XX [I]; YYI:= YY [I];
      YYY:= YYI + (YY [I + 1] - YYI) * (XXX - XXI) / (XX [I + 1] - XXI);
      Y:= YYY
    "END" Y;

    "REAL" "PROCEDURE" R (X); "VALUE" X; "REAL" X;
    R:= EXP (Y (X));

    "REAL" "PROCEDURE" F (X); "VALUE" X; "REAL" X;
    "BEGIN" "REAL" YX;
      YX:= Y (X); F:= EXP (YX) * (YX - 1)
    "END" F;

    "REAL" "PROCEDURE" NORM 2 (X, Y); "ARRAY" X, Y;
    "BEGIN" "INTEGER" I; "REAL" S;
      S:= 0;
      "FOR" I:= 0 "STEP" 1 "UNTIL" N "DO" S:= S + (X [I] - Y [I]) ** 2;
      NORM 2:= S
    "END" NORM 2;

    ORDE:= 2; III:= 0; XXX:= "+300; C:= 0,66802784575;
    E [1]:= E [4]:= 1; E [2]:= E [3]:= E [5]:= E [6]:= 0;

    "FOR" I:= 0 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" XX [I]:= I / N; YN [I]:= 0 "END";

    "FOR" IT:= 0, IT + 1 "WHILE" NORM > "-9, -4, -6 "DO"
    "BEGIN" "IF" IT < 0 "THEN" ORDE:= -IT;
      "FOR" I:= 0 "STEP" 1 "UNTIL" N "DO" YY [I]:= YN [I];
      FEMLAG (XX, YN, N, R, F, ORDE, E);
      NORM:= SQRT (NORM 2 (YY, YN));
      OUTPUT (61, "(""("VERSCHIL MET VORIG RESULTAAT =")", 2ZD.9D,
              "(" ,ORDE V.D. METHODE = ")", ZD, /")", NORM, ORDE)
    "END";
    OUTPUT (61, "("//, 8B, "("X - WAARDE")", 9B, "("Y - WAARDE")",
          9B, "("EXACTE OPLOSSING")", 2/")");
    "FOR" I:= 0 "STEP" 1 "UNTIL" N "DO"
    OUTPUT (61, "("5B, 3 (-2ZD.9D, 5B), /")", XX [I], YY [I],
          2 * LN (C / COS (C * (XX [I] - 0,5))) + LN (2,0));
    OUTPUT (61, "("*")")
  "END"
"END"
```

```
VERSCHIL MET VORIG RESULTAAT =  0,184646567 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000836337 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000000016 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000000000 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000533397 ,ORDE V,D, METHODE =  4
VERSCHIL MET VORIG RESULTAAT =  0,000001004 ,ORDE V,D, METHODE =  6
```

X = WAARDE	Y = WAARDE	EXACTE OPLOSSING
0,000000000	0,000000000	=0,000000006
0,200000000	=0,073267595	=0,073268387
0,400000000	=0,109236582	=0,109237726
0,600000000	=0,109236582	=0,109237726
0,800000000	=0,073267595	=0,073268387
1,000000000	0,000000000	=0,000000006

```
VERSCHIL MET VORIG RESULTAAT =  0,261939722 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,001157893 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000000022 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000000000 ,ORDE V,D, METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000190826 ,ORDE V,D, METHODE =  4
VERSCHIL MET VORIG RESULTAAT =  0,000000091 ,ORDE V,D, METHODE =  6
```

X = WAARDE	Y = WAARDE	EXACTE OPLOSSING
0,000000000	0,000000000	=0,000000006
0,100000000	=0,041435594	=0,041435629
0,200000000	=0,073268332	=0,073268387
0,300000000	=0,095799784	=0,095799853
0,400000000	=0,109237649	=0,109237726
0,500000000	=0,113703582	=0,113703662
0,600000000	=0,109237649	=0,109237726
0,700000000	=0,095799784	=0,095799853
0,800000000	=0,073268332	=0,073268387
0,900000000	=0,041435594	=0,041435629
1,000000000	0,000000000	=0,000000006

```
VERSCHIL MET VORIG RESULTAAT =  0,370668240 ,ORDE V.D. METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,001629765 ,ORDE V.D. METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000000030 ,ORDE V.D. METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000000000 ,ORDE V.D. METHODE =  2
VERSCHIL MET VORIG RESULTAAT =  0,000067653 ,ORDE V.D. METHODE =  4
VERSCHIL MET VORIG RESULTAAT =  0,000000008 ,ORDE V.D. METHODE =  6
```

X = WAARDE	Y = WAARDE	EXACTE OPLOSSING
0,000000000	0,000000000	=0,000000006
0,050000000	=0,021940976	=0,021940983
0,100000000	=0,041435621	=0,041435629
0,150000000	=0,058531193	=0,058531201
0,200000000	=0,073268379	=0,073268387
0,250000000	=0,085681701	=0,085681709
0,300000000	=0,095799844	=0,095799853
0,350000000	=0,103645927	=0,103645936
0,400000000	=0,109237717	=0,109237726
0,450000000	=0,112587791	=0,112587801
0,500000000	=0,113703652	=0,113703662
0,550000000	=0,112587791	=0,112587801
0,600000000	=0,109237717	=0,109237726
0,650000000	=0,103645927	=0,103645936
0,700000000	=0,095799844	=0,095799853
0,750000000	=0,085681701	=0,085681709
0,800000000	=0,073268379	=0,073268387
0,850000000	=0,058531193	=0,058531201
0,900000000	=0,041435621	=0,041435629
0,950000000	=0,021940976	=0,021940983
1,000000000	0,000000000	=0,000000006

```
C     PROBLEEM 3.1.4.3
C     M.B.V. D02ADF (NAG)
C
      PROGRAM MASTER (OUTPUT, TAPE 6 = OUTPUT)
      EXTERNAL DERIV, OUT
      DIMENSION U (2, 2), V (2, 2), E (2), Y (11, 2), G (2, 3), INT (2),
     1WSP (2, 5), WS (2), YO (2), EE (2), AA (2), BB (2), CC (2), DD (2)
      U (1, 1) = 0.0
      V (1, 1) = 0.0
      U (1, 2) = 0.0
      V (1, 2) = 0.0
      U (2, 1) = 1.0
      V (2, 1) = 1.0
      U (2, 2) = -1.0
      V (2, 2) = 1.0
      C = 0.6680278475
      R = 0.5
      X = 0.0
      X1 = 1.0
      E (1) = 1E-7
      E (2) = 1E-7
      H = 0.000001
      N = 2
      M1 = 11
      IFAIL = 1
      WRITE (6, 98)
98    FORMAT (28H1   CURRENT PARAMETER VALUES,
     1 4X, 23HERRORS IN Y(I) AT X = R, 6X, 17HSUM (ERRORS ** 2))
      CALL D02ADF (U, V, E, Y, H, X, X1, R, N, M1, DERIV, OUT, G,
     1 INT, WSP, WS, YO, EE, AA, BB, CC, DD, IFAIL)
      IF (M .GT. 0) GOTO 3
      WRITE (6, 99)
99    FORMAT (16HO FINAL SOLUTION, 18X, 23H EXACT SOLUTION OF Y(1))
      DO 2 I = 1, 11
      X = (I - 1) / 10.0
      YY = 2 * ALOG (C / COS (C * (X - 0.5))) + ALOG (2.0)
      WRITE (6, 100) X, (Y (I, J), J = 1, 2), YY
100   FORMAT (1H, F4.1, 3F14.10)
2     CONTINUE
3     WRITE (6, 101) M
101   FORMAT (13H ERROR NUMBER, I3)
10    CONTINUE
      STOP
      END
```

```
      SUBROUTINE DERIV (G, Z, X)
      DIMENSION G (2), Z (2)
      G (1) = Z (2)
      G (2) = EXP (Z (1))
      RETURN
      END



      SUBROUTINE OUT (A, B, F, R, N)
      DIMENSION A (N, 2), B (N, 2), F (N), W (2)
      J = 1
      DO 3 K = 1, 2
      DO 2 I = 1, N
      IF (B (I, K) .EQ. 0.0) GOTO 2
      W (J) = A (I, K)
      J = J + 1
2     CONTINUE
3     CONTINUE
      WRITE (6, 100) (W (J), J = 1, 2), (F (I), I = 1, 2), R
100   FORMAT (1H0, 2F14.10, 2E13.4, E15.4)
      RETURN
      END
```

CURRENT PARAMETER VALUES		ERRORS IN Y[I] AT X = R		SUM (ERRORS ** 2)
1,0000000000	-1,0000000000	,1549E-08	-,3372E+01	,1137E+02
-,4081784020	,4123032897	-,2144E-02	-,1193E+00	,1423E-01
-,4600197170	,4602415240	-,1152E-03	-,7816E-02	,6111E-04
-,4633930821	,4634068291	-,7141E-05	-,5192E-03	,2696E-06
-,4636166786	,4636175736	-,4649E-06	-,3452E-04	,1192E-08
-,4636325931	,4636325931	-,1616E-12	-,9497E-11	,9023E-22
-,4636325931	,4636325931	0,	-,1554E-14	,2416E-29

	FINAL SOLUTION		EXACT SOLUTION OF Y[I]
0,0	0,0000000000	-,4636325931	,0000000002
,1	-,0414356219	-,3657558958	-,0414356231
,2	-,0732683795	-,2713999484	-,0732683816
,3	-,0957998449	-,1795742504	-,0957998475
,4	-,1092377183	-,0893852455	-,1092377212
,5	-,1137036534	,0000000009	-,1137036563
,6	-,1092377183	,0893852455	-,1092377212
,7	-,0957998449	,1795742504	-,0957998475
,8	-,0732683795	,2713999484	-,0732683816
,9	-,0414356219	,3657558958	-,0414356231
1,0	0,0000000000	,4636325931	,0000000002

ERROR NUMBER 0

3. RANDWAARDEPROBLEMEN

3.2. Elliptische randwaardeproblemen voor partiële differentiaalvergelijkingen

door M. Bakker en P.J. van der Houwen
(Mathematisch Centrum)

3.2.1. Inleiding

Evenals het geval is voor beginwaardeproblemen bestaat er voor randwaardeproblemen voor partiële differentiaalvergelijkingen geen algemeen toepasbare programmatuur. Zoals gebruikelijk is in de numerieke wiskunde vervangt men het probleem door een ander probleem waarvoor men de gereedschappen wel beschikbaar heeft; de oplossing van het nieuwe probleem moet natuurlijk wel een beetje lijken op de oplossing van het oorspronkelijke probleem. We zullen twee methoden bespreken om elliptische differentiaalvergelijkingen in een stelsel van lineaire vergelijkingen (als de elliptische vergelijking zelf lineair is) of niet-lineaire vergelijkingen over te voeren. De eerste methode, welke het langste toepassing vindt, is de *eindige-differentiemethode*, de tweede methode is de *eindige-elementenmethode* welke de laatste jaren zeer in de belangstelling staat. Beide methoden reduceren het randwaardeprobleem tot een in het algemeen zeer groot stelsel vergelijkingen met een "ijle" coëfficiëntenmatrix of, in het geval van niet-lineaire problemen, een "ijle" Jacobiaan (dit is de matrix van partiële afgeleiden). Met ijl wordt bedoeld dat de matrix overwegend elementen gelijk 0 heeft. De oplossing van dergelijke grote ijle stelsels van duizenden vergelijkingen vraagt een geheugenzuinige numerieke oplossingsmethode. We zullen twee oplossingsmethoden aan de orde stellen: de *methode van Richardson* en de *geconjugeerde-gradiëntenmethode*. Beide kunnen volstaan met een aantal geheugenplaatsen dat evenredig met het aantal vergelijkingen is.

Een en ander zal toegelicht worden aan de hand van het volgende elliptische randwaardeprobleem

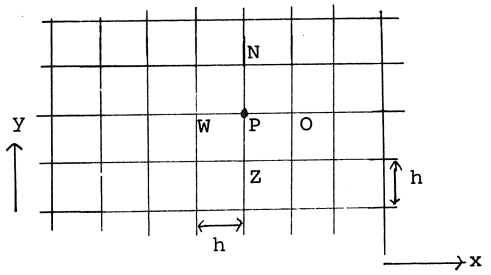
$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial x^2} - 10\,u = -1, \qquad 0 \le x, y \le 1,$$

$$u(x,0) = u(x,1) = u(0,y) = u(1,y) = 0. \tag{3.2.1.1}$$

3.2.2. Discretisering door middel van eindige differenties

In de eindige-differentiemethode worden differentiaties door differentiequotiënten vervangen. Dit proces wordt discretisering of discretisatie genoemd en is identiek aan de in paragraaf 2.2.3 beschreven semi-discreti-

satie van begin-randwaardeproblemen. In het bijzonder zullen we de discretisering van vergelijking (3.2.1.1) bespreken. Daartoe kiezen we in het (x,y)-vlak een uniform rooster met vierkante mazen met zijde h (zie figuur 3.2.1)



Figuur 3.2.1 Uniform rooster in het (x,y)-vlak

De eenvoudigste 2^e orde discretisering van $\partial^2u/\partial x^2 + \partial^2u/\partial y^2$ in een punt P wordt gegeven door (vierkante haken geeft discretisering aan)

$$(3.2.2.1) \qquad \left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right] = \frac{U_W - 2U_P + U_O}{h^2} + \frac{U_N - 2U_P + U_Z}{h^2},$$

waarin $\partial^2u/\partial x^2$ en $\partial^2u/\partial y^2$ dus eenvoudig door centrale differentiequotiënten zijn vervangen. Een handige notatie voor formules als (3.2.2.1) is de volgende:

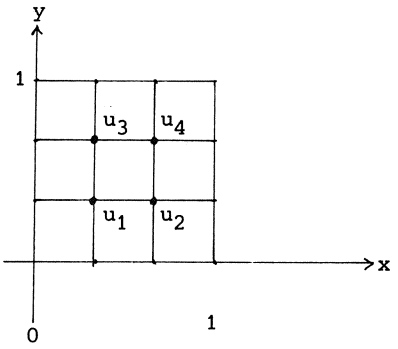
$$(3.2.2.1') \qquad \left[\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right] = \frac{1}{h^2} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Hierin noteert men de gewichten die de functie U in de verschillende roosterpunten uit de differentieformule heeft, in een array en wel zodanig dat de plaatsen in dit array corresponderen met de plaats van de roosterpunten in het rooster. Voorstelling (3.2.2.1') wordt ook wel de *molecuulvoorstelling* genoemd.

Vervangt men in (3.2.1.1) de differentiaaloperator $\partial^2/\partial x^2 + \partial^2/\partial y^2$ door de differentieoperator (3.2.2.1'), dan gaat het randwaardeprobleem over in een lineair stelsel

$$(3.2.2.2) \qquad A\vec{u} = \vec{f},$$

waarin $\vec{u}$ de benadering van de oplossing $u(x,y)$ in de roosterpunten voorstelt en $\vec{f}$ de inhomogene termen van de differentiaalvergelijking en de randvoorwaarden bevat. Als illustratie stellen we het stelsel (3.2.2.2) op voor het geval waarin het vierkant $0 \leq x \leq 1$, $0 \leq y \leq 1$ in 9 vierkantjes onderverdeeld is (zie figuur 3.2.2). Voor u_1 vinden we dan



Figuur 3.2.2 Rooster met h = 1/3

$$(\tfrac{1}{1/3})^2 [-4u_1 + u_3 + u_2] - 10u_1 = -1$$

en soortgelijke vergelijkingen voor u_2, u_3 en u_4. Aldus komen we tot het stelsel

$$(3.2.2.2') \quad \begin{pmatrix} -46 & 9 & 9 & 0 \\ 9 & -46 & 0 & 9 \\ 9 & 0 & -46 & 9 \\ 0 & 9 & 9 & -46 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = - \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

In de numerieke oplossingsmethoden voor stelsels afkomstig van randwaardeproblemen voor partiële differentiaalvergelijkingen speelt het spectrum van de coëfficiëntenmatrix vaak een belangrijke rol. In het bovenstaande geval van h = 1/3 zien we direct dat de eigenwaarden reëel zijn (A is symmetrisch) en, krachtens de bekende stelling van Gerschgorin, dat de eigenwaarden in een cirkel met middelpunt - 46 en straal 18 liggen. Derhalve liggen de eigenwaarden in het interval [-64, -28]. Voor een willekeurige maaswijdte h geldt ook dat A symmetrisch is en vinden we met Gerschgorin dat de eigenwaarden in het interval

$$\left[-\frac{8}{h^2} - 10, -10 \right]$$

liggen. Uiteraard is dit een wat ruwe schatting. Voor een nauwkeuriger analyse verwijzen we naar de literatuur (e.g. FORSYTHE & WASOW [1961]).

3.2.3. De eindige elementenmethode

De oplossing u van (3.2.1.1) minimaliseert de convexe functionaal

$$I[v] = \iint_R [v_x^2 + v_y^2 + 10v^2 - 2v]dxdy$$

over de ruimte V van functies die continu zijn op $R = [0,1] \times [0,1]$ en nul zijn op ∂R. Voor u geldt de betrekking

$$(u_x, v_x) + (u_y, v_y) + 10(u,v) = (1,v), \quad v \in V,$$

waarbij (,) het inproduct over R betekent. De Ritz-Galerkin-methode bestaat hierin dat we u benaderen door I[v] te minimaliseren over een eindig-dimensionale deelruimte S van V. Als $\{\phi_1,\ldots,\phi_N\}$ een basis is van S, dan wordt de approximatie $u_s \in S$ van u bepaald door

$$(3.2.3.1) \quad \left(\frac{\partial u_s}{\partial x}, \frac{\partial \phi_i}{\partial x}\right) + \left(\frac{\partial u_s}{\partial y}, \frac{\partial \phi_i}{\partial y}\right) + 10(u_s, \phi_i) = (1, \phi_i), \quad i = 1,\ldots,N.$$

Schrijven we $u_s = \Sigma_{i=1}^N q_i \phi_i(x,y)$, dan wordt $q = (q_1,\ldots,q_N)^T$ bepaald door

$$\vec{Aq} = \vec{b};$$

$$(3.2.3.2) \quad A = \left(\left(\frac{\partial \phi_i}{\partial x}, \frac{\partial \phi_j}{\partial x}\right) + \left(\frac{\partial \phi_i}{\partial y}, \frac{\partial \phi_j}{\partial y}\right) + 10(\phi_i, \phi_j)\right);$$

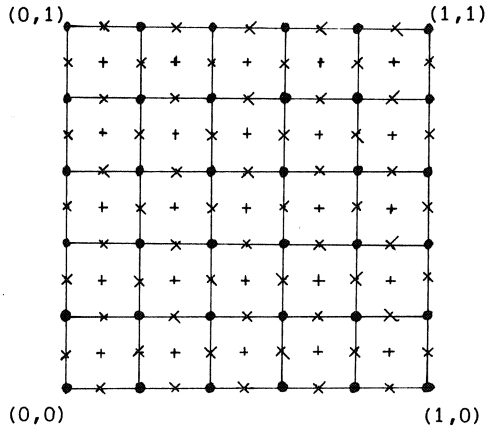
$$\vec{b} = ((1, \phi_i)).$$

De eindige-elementenmethode nu bestaat hierin dat S zo gekozen wordt dat

1. u_s voldoende nauwkeurig is;
2. het stelsel (3.2.3.2) een ijle matrix A heeft en dus ook voor zeer grote N nog oplosbaar is.

We verdelen R in $M = n^2$ vierkantjes van gelijke oppervlakte $\sigma = h^2$, h de maaswijdte. Als roosterpunten kiezen we

1. de hoekpunten van de vierkantjes: ● ;
2. de middens der zijden : x ;
3. de middens der vierkantjes : + .



Figuur 3.2.3 Uniforme elementen

Voor S kiezen we de ruimte van functies die

1. continu zijn op R en nul op ∂R;
2. op ieder deelvierkantje e_ℓ een bikwadratisch polynoom zijn d.w.z. een polynoom van de vorm $(a_0+a_1x+a_2x^2)(b_0+b_1y+b_2y^2)$.

Om een basis voor S te definiëren nummeren we de roosterpunten van 1 tot en met $N = (2n-1)^2$. De basisfuncties van S zijn bikwadratische polynomen $\phi_i(x,y)$ met de eigenschap

$$\phi_i(P_j) = \delta_{ij},\ 1 \le i,\ j \le N.$$

Het is gemakkelijk te verifiëren dat $\phi_i(x,y) \equiv 0$ op die vierkantjes waartoe P_i niet behoort. Bijgevolg is $a_{ij} = 0$ als P_i en P_j niet tot hetzelfde vierkantje behoren. Een blik op figuur 3.2.3 leert dat roosterpunten ● aan ten hoogste 24 andere roosterpunten zijn gekoppeld, roosterpunten x aan ten hoogste 14 en roosterpunten + aan ten hoogste 8 andere roosterpunten.

We werken in formule (3.2.3.2) uit voor de verschillende roosterpunten. Voor de integratie maken we gebruik van de tweedimensionale regel van Simpson. Voor een rechtvaardiging hiervan zie STRANG [1972].

Om compact aan te geven hoe de verschillende roosterpunten aan elkaar gekoppeld zijn, gebruiken we eveneens de molecuulnotatie. We krijgen dan voor de verschillende soorten roosterpunten de volgende moleculen.

Roosterpunten •

$$\frac{1}{9}\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -8 & 0 & 0 \\ 1 & -8 & 28+10h^2 & -8 & 1 \\ 0 & 0 & -8 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} ;$$

Roosterpunten x

$$\frac{2}{9}\begin{bmatrix} 0 & 0 & -4 & 0 & 0 \\ 1 & -8 & 22+10h^2 & -8 & 1 \\ 0 & 0 & -4 & 0 & 0 \end{bmatrix} \quad \text{of} \quad \frac{2}{9}\begin{bmatrix} 0 & 1 & 0 \\ 0 & -8 & 0 \\ -4 & 22+10h^2 & -4 \\ 0 & -8 & 0 \\ 0 & 1 & 0 \end{bmatrix} ;$$

Roosterpunten +

$$\begin{bmatrix} 0 & -4 & 0 \\ -4 & 16+10h^2 & -4 \\ 0 & -4 & 0 \end{bmatrix}.$$

Merk op dat er geen "kruiskoppeling" optreedt. Bijgevolg is de ijlheid van A aanzienlijk groter dan verwacht mocht worden. Het rechterlid van (3.2.3.2) is als volgt bepaald

$$b_i = \begin{cases} \frac{4}{9}h^2, & \text{voor + roosterpunten,} \\ \frac{2}{9}h^2, & \text{voor x roosterpunten,} \\ \frac{1}{9}h^2, & \text{voor • roosterpunten.} \end{cases}$$

3.2.4. De methode van Richardson

Stel dat we het niet-lineaire stelsel

$$(3.2.4.1)\quad \vec{F}(\vec{u}) = \vec{0}$$

willen oplossen, waarin $\vec{F}$ een gegeven vectorfunctie in $\vec{u}$ is. In het bijzonder beschouwen we stelsels waarvoor de Jacobiaan

$$(3.4.2.4)\quad J(\vec{u}) = \left(\frac{\partial F_r(\vec{u})}{\partial u_k}\right), \qquad \begin{matrix} r = \text{rij-index} \\ k = \text{kolomindex} \end{matrix}$$

een *ijle* matrix is met *positieve* eigenwaarden. Voorbeeld van een dergelijk stelsel is het door semi-discretisatie van probleem (3.2.1.1) verkregen stelsel (3.2.2.2). Dit stelsel is *lineair* zodat J = A. Een voorbeeld van een *niet-lineair* stelsel levert de discretisatie van het probleem

$$(3.2.4.3)\quad \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - 10 \sinh(10\, u) = 0,$$

met randvoorwaarden van de eerste soort langs de randen x = 0, x = 1, y = 0 en y = 1. Dit probleem is de twee-dimensionale vorm van een probleem van Troesch (zie ROBERTS en SHIPMAN [1972]). Discretisering van de Laplace-operator leidt tot het stelsel

$$(3.2.4.3')\quad A\vec{u} - \vec{g}(\vec{u}) = \vec{f},$$

waarin A een matrix is, $\vec{f}$ een constante vector en de componenten van $\vec{g}(\vec{u})$ door $10 \sinh(10\, u_j)$ gegeven worden. Het is duidelijk dat de Jacobiaan van dit stelsel gegeven wordt door

$$J(\vec{u}) = A - D(\vec{u}),$$

waarin $D(\vec{u})$ een diagonaalmatrix is met diagonaalelementen $d_{jj} = 100 \cosh(10\, u_j)$.

We beschouwen het iteratieproces

$$(3.2.4.4)\quad \vec{u}_{n+1} = \alpha_n \vec{u}_n - \omega_n \vec{F}(\vec{u}_n) + (1-\alpha_n)\vec{u}_{n-1}, \qquad n = 1,2,\ldots .$$

De vectoren $\vec{u}_0$ en $\vec{u}_1$ moeten hierin als startvectoren opgegeven worden. De

parameters α_n en ω_n kiezen we zodanig dat de convergentiesnelheid zo groot mogelijk is. Daartoe substitueren we in (3.2.4.4)

$$\vec{u}_n = \vec{u} + \vec{v}_n,$$

waarin $\vec{u}$ de oplossing van (3.2.4.1) is en $\vec{v}_n$ de fout in de iterand $\vec{u}_n$. Voor $\vec{u}_n$ voldoende dichtbij de oplossing $\vec{u}$ geldt

$$(3.2.4.5) \quad \vec{v}_{n+1} = [\alpha_n - \omega_n J(\vec{u})]\vec{v}_n + (1-\alpha_n)\vec{v}_{n-1}.$$

Stel dat we $\vec{u}_1 = \vec{u}_0$, dus $\vec{v}_1 = \vec{v}_0$ gekozen zouden hebben. Uit (3.2.4.5) volgt dan dat

$$(3.2.4.5') \quad \vec{v}_{n+1} = P_n(J(\vec{u}))\vec{v}_0,$$

waarin P_n een polynoom van de graad n is met $P_n(0) = 1$.
Blijkbaar moeten we de parameters α_n en ω_n zodanig kiezen dat $\|P_n(J(\vec{u}))\|$ zo klein mogelijk is. In de practijk vervangt men deze opgave door een iets eenvoudiger probleemstelling: *kies* P_n *zodanig dat de eigenwaarden van* $P_n(J(\vec{u}))$ *minimaal zijn.* Zoals gezegd beperken we ons tot stelsels waarin $J(\vec{u})$ positieve eigenwaarden heeft. Stel dat men bovendien weet dat de eigenwaarden in een interval [a,b] liggen. We krijgen dan de opgaaf het polynoom van de graad n te construeren dat de kleinste supremumnorm op het interval [a,b] heeft met als nevenvoorwaarde $P_n(0) = 1$. In een heel ander verband werd dit minimaxprobleem al door Markoff in 1892 opgelost en is na hem nog vele malen herontdekt. In elk geval kende RICHARDSON [1910] de oplossing niet en stelde voor om de nulpunten gelijkelijk over het interval [a,b] te verdelen. Pas in 1950 kwamen Shortley en Flanders met de optimale verdeling van de nulpunten aandragen. Er geldt namelijk dat de optimale $P_n(z)$ gegeven wordt door

$$(3.2.4.6) \quad P_n(z) \equiv \frac{T_n\left(\frac{b+a-2z}{b-a}\right)}{T_n\left(\frac{b+a}{b-a}\right)},$$

waarin T_n het Chebyshev-polynoom van de graad n voorstelt. Uit deze identificatie kunnen de parameters α_n en ω_n berekend worden; we vinden

$$(3.2.4.7)\quad \alpha_n = 2q\,\frac{T_n(q)}{T_{n+1}(q)},\quad \omega_n = \frac{4}{b-a}\,\frac{T_n(q)}{T_{n+1}(q)},\qquad q = \frac{b+a}{b-a}.$$

Het aldus gedefinieerde iteratieproces wordt het *2e orde Richardson-proces* genoemd. Er bestaat ook een *1e orde Richardson-proces* waarin alle α_n's gelijk 1 zijn, dus in feite een éénstaps-iteratieproces. Hiervoor geldt echter dat aan (3.2.4.6) voor een vaste, vooraf gekozen waarde van n voldaan wordt; noem deze waarde N dan geldt voor de parameters ω_n

$$(3.2.4.8)\quad \omega_n = 2\left[a+b+(a-b)\cos\left(\frac{2n+1}{2N}\,\pi\right)\right]^{-1},\qquad n = 0,\ldots,N-1.$$

Alhoewel dit proces zuiniger geheugengebruik heeft, weegt het nadeel dat van te voren een waarde van N opgegeven moet worden zo zwaar, dat de 2^e orde versie verre de voorkeur verdient; bovendien is het door (3.2.4.8) gegenereerde proces numeriek instabiel.

3.2.5. Het eliminatieproces

We hebben gezien dat het Richardson-proces vraagt naar het eigenwaardeninterval [a,b]. Hoe scherper dit interval opgegeven kan worden, des te efficiënter werkt de methode. Immers uit (3.2.4.6) volgt voor grote waarden van n

$$|P_n(z)| \le T_n^{-1}\left(\frac{b+a}{b-a}\right) \simeq \frac{1}{2}\exp n\left[\operatorname{arcosh}\left(\frac{b+a}{b-a}\right)\right].$$

Aangezien arcosh(x) een stijgende functie van x is, moet men

$$\frac{b+a}{b-a} = 1 + \frac{2a}{b-a}$$

zo groot mogelijk kiezen, d.w.z. het eigenwaardeninterval zo scherp mogelijk bepalen.

Het eliminatieproces is er op gebaseerd de waarde van a door $\tilde{a} > a$ te vervangen. Het gevolg is dat de eigenvectoren in de fout $\vec{v}_n$ behorende bij de eigenwaarden binnen het interval $[\tilde{a},b]$ des te sneller verdwijnen, maar dat de eigenvectoren behorende bij eigenwaarden uit het interval $[a,\tilde{a}]$ blijven hangen en wel in het bijzonder de eigenvector behorende bij de kleinste eigenwaarde δ_{min} (zie figuur 3.2).

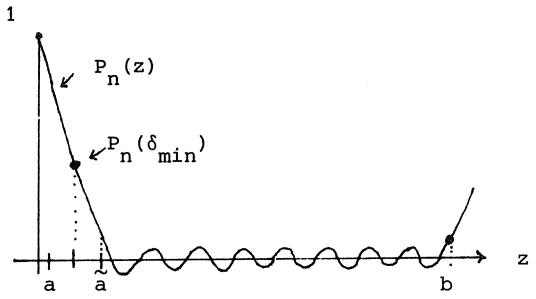


Fig. 3.2. Spectrum van $P_n(J(\vec{u}))$.

Het is mogelijk deze dominante eigenwaarde uit de successieve iteratieresultaten $\vec{u}_n$ te berekenen. Beschikken we eenmaal over δ_{min} dan kan de bijbehorende eigenvector geëlimineerd worden door de parameters α_n en ω_n zo te kiezen dat $P_n(z)$ een nulpunt in δ_{min} heeft. De uitwerking van dit eliminatieproces is nogal technisch en valt buiten het kader van deze syllabus. We vermelden alleen dat de convergentieversnelling aanzienlijk is en verwijzen voor verdere details naar MC Tract 20.

3.2.6. Numerieke experimenten met de procedure RICHARDSON

Probleem (3.2.1.1) werd door middel van de benadering (3.2.2.1) getransformeerd tot een lineair stelsel van de vorm (3.2.2.2). Achtereenvolgens werd de maaswijdte h = .025, .05 en .1 gekozen. Op deze stelsels werd het in paragraaf 3.2.4 beschreven proces van Richardson toegepast, waarvan een ALGOL 60 implementatie in de bibliotheek NUMAL aanwezig is (zie de procedure RICHARDSON gedocumenteerd in section 5.2.1.2.2.1.2 van de NUMAL-manual). Voor de grenzen van het spectrum van de coëfficientenmatrix geldt volgens paragraaf 3.2.2, na vermenigvuldiging met $-h^2$ om positieve eigenwaarden te krijgen,

$$a = 10h^2, \qquad b = 8 + 10h^2.$$

Een iets scherpere analyse dan die met Gerschgorin cirkels leidt tot de wat nauwere grenzen

$$a = 12h^2, \qquad b = 8(1+h^2).$$

Aan het eind van dit hoofdstuk zijn het gebruikte programma en de verkregen resultaten weergegeven; het iteratieproces werd beëindigd wanneer de maximumnorm van het residu $h^2(A\vec{u}_n-\vec{f})$ gezakt was beneden 10^{-5}. Een onderlinge vergelijking van deze resultaten laat zien dat de numerieke oplossing met $h = .1$ in 1 à 2 cijfers, en die met $h = .05$ in 2 à 3 cijfers overeenstemt met de voor $h = .025$ verkregen oplossing.

3.2.7. De geconjugeerde-gradiëntenmethode

De geconjugeerde gradiëntenmethode (af te korten als CG-methode) is een proces waardoor het n-dimensionale lineaire stelsel

(3.2.7.1) $A\vec{u} = \vec{b}$, A symmetrisch positief definiet,

iteratief wordt opgelost in ten hoogste n stappen. Een van de varianten, ontleend aan REID [1971] is de volgende:

$$\begin{aligned}
&\vec{r}_0 = \vec{p}_0 = A\vec{u}_0 - \vec{b} &&,\\
&\vec{u}_{\ell+1} = \vec{u}_\ell - \alpha_\ell \vec{p}_\ell &&,\\
&\alpha_\ell = (\vec{r}_\ell,\vec{p}_\ell)/(A\vec{p}_\ell,\vec{p}_\ell) &&,\\
&\vec{r}_{\ell+1} = A\vec{u}_{\ell+1} - \vec{b} = \vec{r}_\ell - \alpha_\ell A\vec{p}_\ell &&,\\
&\vec{p}_{\ell+1} = \vec{r}_{\ell+1} - \beta_\ell \vec{p}_\ell &&,\\
&\beta_\ell = (\vec{r}_{\ell+1},A\vec{p}_\ell)/(A\vec{p}_\ell,\vec{p}_\ell) &&, \quad \ell = 0,1,\ldots,
\end{aligned} \tag{3.2.7.2}$$

waarbij $\vec{p}_0,\vec{p}_1,\ldots$ de richtingsvectoren en $\vec{r}_0,\vec{r}_1,\ldots$ de residuvectoren heten. Aangezien de richtingsvectoren een A-orthogonaal stelsel vormen ($(A\vec{p}_i,\vec{p}_j) = 0$, $i \neq j$) en de residuvectoren een orthogonaal stelsel (zie REID [1971]), is het direct duidelijk dat in elk geval $\vec{r}_n = \vec{0}$, dus $\vec{u} = \vec{u}$. Uiteraard is het mogelijk dat dit proces eerder afbreekt.

Wat deze methode aantrekkelijk maakt om grote ijle stelsels, voortkomend uit elliptische randwaardeproblemen, op te lossen, is het volgende:

1. het aantal iteratiestappen is eindig;
2. blijkens formule (3.2.7.2) hoeft de matrix A niet expliciet meegegeven te worden als inputparameter, maar impliciet in de vorm van een routine

die bij gegeven vector v de vector $\vec{w} = A\vec{v}$ berekent.

TABEL 1

Beschikbare routines voor de CG methode

Programmatheek	Routine	Geheugen
ACCULIB	CG	4n
NUMAL	CONJ GRAD	2n

3.2.7.1. CONJ GRAD (NUMAL)

Deze routine is ontleend aan REID [1971]. Als inputparameters moeten o.m. worden meegegeven:

1. een beginschatting van $\vec{u}$;
2. het rechterlid van (3.2.7.1);
3. een routine die uit $\vec{v}$ de vector $A\vec{v}$ berekent;
4. een stuurparameter in de vorm van een boolean expression.

Gedurende het iteratieproces kan de gebruiker aan de hand van het aantal uitgevoerde iteratiestappen en het kwadraat van de residunorm bepalen of het proces doorgaat.

Voorbeeld. Met behulp van de aanroep

```
conj grad (matvec,u,b,1,n,
iterate < 10 "and" residunorm >=1.0" - 10,
iterate, residunorm);
```

bepaalt de gebruiker dat het proces wordt afgebroken als er 10 iteraties zijn uitgevoerd, of als de kwadraatnorm van het residu kleiner dan 10^{-10} is.

Uitvoer na m iteratiestappen:

1. een approximatie $\vec{u}^{(m)}$ van $\vec{u}$;
2. de residuvector $\vec{r}^{(m)}$, overgeschreven op $\vec{b}$;
3. de kwadraatnorm $\|\vec{r}^{(m)}\|^2$.

3.2.7.2. CG (ACCULIB)

Deze routine lost het stelsel

(3.2.7.3) $A\vec{u} + \vec{b} = 0$, A symmetrisch positief definiet,

op met behulp van een variant van de CG-methode, ontleend aan GINSBURG [1971]. De gebruiker is hier niet verplicht een beginschatting van u te geven. Wil hij dat toch doen, dan maakt hij dat kenbaar door de bovengrens van het array u een minteken te geven. Een ander verschil met CONJ GRAD is dat CG voortijdig wordt afgebroken als A niet positief definiet is of te slecht geconditioneerd.

Invoerparameters zijn o.m.

1. een beginschatting van $\vec{u}$ (niet verplicht);
2. de vector $\vec{b}$ van (3.2.7.3);
3. een routine die bij gegeven vector $\vec{v}$ de vector $\vec{w} = A\vec{v}$ berekent;
4. de bovengrens van het array $\vec{u}$;
5. een label waarheen de gebruiker wordt gestuurd bij voortijdige beëindiging van CG;
6. het maximum aantal interatiestappen.

Voorbeeld. Met de aanroep

```
          it := 20;
          CG(-100,20,b,matvec,x, it,q,e,exit);
          "goto" after exit;
       exit:
          output(61,"("/,"(" process ended prematurely ")",/")");
       after exit:
          < afdrukken gewenste resultaten >
```

maakt de gebruiker kenbaar dat

1. een beginschatting gegeven is;
2. het iteratieproces ten hoogste 20 stappen bedraagt;
3. melding wordt gemaakt van een eventuele voortijdige beëindiging.

Uitvoer:

1. een approximatie van $\vec{u}$;
2. het aantal iteratiestappen;
3. de vectoren $\vec{q}$ en $\vec{e}$; dit zijn twee p-dimensionale vectoren; voor de betekenis verwijzen we naar GINSBURG [1971].

3.2.8. Oplossing van (3.2.1.1) d.m.v. CONJ GRAD en CG

Een van de aantrekkelijke eigenschappen van de eindige elementenmethode is dat de matrix A en het rechterlid $\vec{b}$ van (3.2.3.2) vierkantsgewijs geëvalueerd kunnen worden (zie FELIPPA & CLOUGH [1970]):

$$A = \sum_{\ell=1}^{M} A_\ell; \qquad \vec{b} = \sum_{\ell=1}^{M} \vec{b}_\ell;$$

$$(3.2.8.1) \qquad A_\ell = \left(\iint_{e_\ell} \left[\frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial x} + \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial y} + 10 \phi_i \phi_j \right] dxdy \right);$$

$$\vec{b}_\ell = \left(\iint_{e_\ell} \phi_i \, dxdy \right); \qquad \ell = 1,\ldots,M.$$

Het gebruik van CG en CONJ GRAD vereist dat bij gegeven vector $\vec{v}$ de vector $\vec{Av}$ wordt berekend door middel van een routine matvec. Een en ander suggereert de segmentsgewijze evaluatie van $\vec{Av}$:

$$\vec{w} = \sum_{\ell=1}^{M} \vec{w}_\ell;$$

$$(3.2.8.2) \qquad \vec{w}_\ell = A_\ell \vec{v} \quad , \qquad \ell = 1,\ldots,M.$$

Aangezien ieder vierkantje ten hoogste negen roosterpunten bevat, bestaat A_ℓ overwegend uit nullen: ten hoogste 81 elementen zijn $\neq 0$ en door het ontbreken van kruiskoppeling wordt dit aantal zelfs tot 45 gereduceerd. $\vec{Aw}$ wordt nu als volgt berekend:

1. eerst wordt $\vec{w} = \vec{Av}$ op 0 gesteld;
2. vervolgens worden op ieder vierkantje e_ℓ de negen (of minder als e_ℓ aan de rand ligt) relevante componenten van $\vec{w}_\ell$ berekend en op de juiste plaats opgeteld.

Voor de boekhoudkundige aspecten van deze operatie verwijzen we naar FELIPPA & CLOUGH [1970].

Voor de oplossing van (3.2.1.1) werd R in resp. 25, 100 en 400 vierkantjes met maaswijdten 0.2, 0.1 en 0.05 verdeeld. Om de segmentsgewijze evaluatie van $\vec{w} = A\vec{v}$ meer efficiënt te kunnen programmeren werden de randpunten ook als roosterpunten beschouwd. Het aantal roosterpunten werd daardoor 121, resp. 441 en 1681.

3.2.8.1. CONJ GRAD

Indien een nauwkeurigheid is vereist van ca. 1% wordt bepaald dat het iteratieproces (3.2.7.2) wordt afgebroken als de residunorm kleiner dan 10^{-5} is. Een onderlinge vergelijking leert dat de oplossing voor h = 0.2 in 2 à 3 cijfers en dat de oplossing voor h = 0.1 in 4 à 5 cijfers overeenkomt met de oplossing voor h = 0.05. Hieronder drukken we het aantal iteratiestappen m en $\|A\vec{u}_m - \vec{b}\|$ af.

TABEL 2

h	m	$\|A\vec{u}_m - \vec{b}_m\|$
0.20	14	$4.77_{10}{}^{-6}$
0.10	31	$5.63_{10}{}^{-6}$
0.05	63	$8.11_{10}{}^{-6}$

3.2.8.2. CG

Aangezien deze routine geen eenvoudige mogelijkheden geeft om het iteratieproces tussentijds te sturen, wordt als bovengrens van het aantal iteratiestappen de dimensie van het array u gegeven.

TABEL 3

h	m
0.20	15
0.10	48
0.05	80

Literatuur

FORSYTHE, G. & W. WASOW [1960], *Finite Difference Methods for Partial Differential Equations*, Wiley, New York.

GINSBURG, T. [1971], *The Conjugate Gradient Method*, Uit: J.H. Wilkinson & C. Reinsch (editors), Linear Algebra II, Springer-Verlag, Berlijn, Heidelberg, New York.

FELIPPA, C.A. & R.W. CLOUGH [1970], *The finite element method in solid mechanics*, Uit: G. Birkhoff (editor), Numerical Solution of Field Problems in Continuum Physics, Duke University, SIAM-AMS Proceedings, Vol. 2.

REID, J.K. (editor) [1971], *Large sparse sets of linear equations*, p. 231 ff., Academic Press, London, New York.

RICHARDSON, L.F. [1910], *The Approximate Arithmetical Solution by Finite Differences of Physical Problems Involving Differential Equations*, Phil. Trans. Roy. Soc. London, p. 307 ff.

ROBERTS, S. & J. SHIPMAN [1972], *Solution of Troesch's two-point boundary value problem by a combination of techniques*, J. Comp. Phys. 10, p. 232 ff.

STRANG, G. [1972], *Variational crimes in the finite element method*, Uit: A.K. Aziz (editor).
The Mathematical Foundations of the Finite Element Method..., Academic Press, London, New York.

Bijlagen

```
"BEGIN" "INTEGER" N; "REAL" K2;
     K2:=-10; "FOR"N:=10,20,40 "DO"
"BEGIN" "INTEGER" N1, I, J, K, ISTEP, IT;
    "REAL" H, H2, H2K2, RATE, DOMEIGVAL;
    "ARRAY" U[0:N,0:N], DISCR[1:2];

    "PROCEDURE" RICHARDSON(U,LJ,UJ,LL,UL,INAP,RESIDUAL,A,B,N,
          DISCR,K,RATE,DOMEIGVAL,OUT); "CODE" 33170;

    "PROCEDURE" INIMAT(A,B,C,D,E,F); "CODE" 31011;

    "PROCEDURE" DUPMAT(A,B,C,D,E,F); "CODE" 31035;

    "PROCEDURE" RESIDUAL(U); "ARRAY" U;
    "BEGIN" "INTEGER" I, J; "ARRAY" AU[0:N,0:N];
      INIMAT(0,N,0,N,AU,0); DUPMAT(1,N1,1,N1,AU,U);
      "FOR"I:= 1 "STEP" 1 "UNTIL" N1"DO"
      "FOR"J:= 1 "STEP" 1 "UNTIL" N1"DO"
      U[I,J]:= (4-H2K2)*AU[I,J]-(AU[I,J-1]+AU[I,J+1]+AU[I-1,J]
      +AU[I+1,J])-H2
    "END";

    "PROCEDURE" OUT(K); "VALUE" K; "INTEGER" K;
    "BEGIN"    "END";

    N1:=N-1; H:=1/N; H2:=H*H; H2K2:=H2*K2;  INIMAT(0,N,0,N,U,0);
    RICHARDSON(U,1,N1,1,N1,"TRUE",RESIDUAL,12*H2,6*(1 + H2),
    "IF" DISCR[1]<"-5 "THEN" 0 "ELSE" 150,DISCR,K,RATE,DOMEIGVAL,OUT);
    OUTPUT(61,"("*,"("AANTAL AANROEPEN VAN RESIDUAL;")"ZZD,
      4B"("CONVERGENTIESNELHEID; ")"D,4D"-D,
      4B"("RESIDUNORM; ")"D,4D"-D,4B"("MAASWIJDTE; ")",3D,3/")",
      K,RATE,DISCR[2],H); ISTEP:= N//10;
    "FOR" I:=0"STEP" ISTEP "UNTIL" N "DO"
    "BEGIN" OUTPUT(61,
"("//,"(" X = 0.0, 0.1, ..., 1.0; Y = ")"D.D,//")",(I//ISTEP)*0.1);
         "FOR"J:=0 "STEP" ISTEP "UNTIL" N "DO"
         OUTPUT(61,"("2B-D.4D"-D")",U[I,J])
    "END"
"END"
"END"
```

AANTAL AANROEPEN VAN RESIDUAL: 37 CONVERGENTIESNELHEID: 2,3588"-1 RESIDUNORM: 2,8203"-6 MAASWIJDTE: ,100

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 9,8224"-3 1,5136"-2 1,8073"-2 1,9565"-2 2,0023"-2 1,9565"-2 1,8073"-2 1,5136"-2 9,8224"-3 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5136"-2 2,4163"-2 2,9398"-2 3,2120"-2 3,2964"-2 3,2120"-2 2,9398"-2 2,4163"-2 1,5136"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8073"-2 2,9398"-2 3,6178"-2 3,9768"-2 4,0889"-2 3,9768"-2 3,6178"-2 2,9398"-2 1,8073"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9565"-2 3,2120"-2 3,9768"-2 4,3860"-2 4,5145"-2 4,3860"-2 3,9768"-2 3,2120"-2 1,9565"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0023"-2 3,2964"-2 4,0889"-2 4,5145"-2 4,6484"-2 4,5145"-2 4,0889"-2 3,2964"-2 2,0023"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9565"-2 3,2120"-2 3,9768"-2 4,3860"-2 4,5145"-2 4,3860"-2 3,9768"-2 3,2120"-2 1,9565"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8073"-2 2,9398"-2 3,6178"-2 3,9768"-2 4,0889"-2 3,9768"-2 3,6178"-2 2,9398"-2 1,8073"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5136"-2 2,4163"-2 2,9398"-2 3,2120"-2 3,2964"-2 3,2120"-2 2,9398"-2 2,4163"-2 1,5136"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 9,8224"-3 1,5136"-2 1,8073"-2 1,9565"-2 2,0023"-2 1,9565"-2 1,8073"-2 1,5136"-2 9,8224"-3 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

AANTAL AANROEPEN VAN RESIDUAL: 71 CONVERGENTIESNELHEID: 1,1382"-1 RESIDUNORM: 1,3834"-6 MAASNIJDTE: ,050

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0018"-2 1,5332"-2 1,8249"-2 1,9730"-2 2,0184"-2 1,9730"-2 1,8249"-2 1,5332"-2 1,0018"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5332"-2 2,4422"-2 2,9667"-2 3,2386"-2 3,3228"-2 3,2386"-2 2,9667"-2 2,4422"-2 1,5332"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8249"-2 2,9667"-2 3,6484"-2 4,0085"-2 4,1209"-2 4,0085"-2 3,6484"-2 2,9667"-2 1,8249"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9730"-2 3,2386"-2 4,0085"-2 4,4200"-2 4,5490"-2 4,4200"-2 4,0085"-2 3,2386"-2 1,9730"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0184"-2 3,3228"-2 4,1209"-2 4,5490"-2 4,6836"-2 4,5490"-2 4,1209"-2 3,3228"-2 2,0184"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9730"-2 3,2386"-2 4,0085"-2 4,4200"-2 4,5490"-2 4,4200"-2 4,0085"-2 3,2386"-2 1,9730"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8249"-2 2,9667"-2 3,6484"-2 4,0085"-2 4,1209"-2 4,0085"-2 3,6484"-2 2,9667"-2 1,8249"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5332"-2 2,4422"-2 2,9667"-2 3,2386"-2 3,3228"-2 3,2386"-2 2,9667"-2 2,4422"-2 1,5332"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0018"-2 1,5332"-2 1,8249"-2 1,9730"-2 2,0184"-2 1,9730"-2 1,8249"-2 1,5332"-2 1,0018"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

AANTAL AANROEPEN VAN RESIDUAL: 135 CONVERGENTIESNELHEID: 5,4688"-2 RESIDUNORM: 6,2767"-7 MAASWIJDTE: ,025

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0074"-2 1,5385"-2 1,8298"-2 1,9776"-2 2,0229"-2 1,9776"-2 1,8298"-2 1,5385"-2 1,0074"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5385"-2 2,4495"-2 2,9744"-2 3,2462"-2 3,3302"-2 3,2462"-2 2,9744"-2 2,4495"-2 1,5385"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8298"-2 2,9744"-2 3,6573"-2 4,0176"-2 4,1300"-2 4,0176"-2 3,6573"-2 2,9744"-2 1,8298"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9776"-2 3,2462"-2 4,0176"-2 4,4296"-2 4,5587"-2 4,4296"-2 4,0176"-2 3,2462"-2 1,9776"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0229"-2 3,3302"-2 4,1300"-2 4,5587"-2 4,6933"-2 4,5587"-2 4,1300"-2 3,3302"-2 2,0229"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9776"-2 3,2462"-2 4,0176"-2 4,4296"-2 4,5587"-2 4,4296"-2 4,0176"-2 3,2462"-2 1,9776"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8298"-2 2,9744"-2 3,6573"-2 4,0176"-2 4,1300"-2 4,0176"-2 3,6573"-2 2,9744"-2 1,8298"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5385"-2 2,4495"-2 2,9744"-2 3,2462"-2 3,3302"-2 3,2462"-2 2,9744"-2 2,4495"-2 1,5385"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0074"-2 1,5385"-2 1,8298"-2 1,9776"-2 2,0229"-2 1,9776"-2 1,8298"-2 1,5385"-2 1,0074"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

```
"BEGIN" "INTEGER" N; "REAL" K2;
 K2:= 10; "FOR" N:= 5, 10, 20 "DO"
"BEGIN" "INTEGER" NSQ, M, M2, M3, M4, MSQ, I, J, IT, ISTEP, JSTEP;
"REAL" H, H2, H2K2, EPS; "ARRAY" U, F[1:(2*N+1)**2];

"PROCEDURE" CONJ GRAD(A, X, B, L, U, GO ON, IT, EPS); "CODE" 34220;

"PROCEDURE" INIVEC(L,U,A,X); "CODE" 31010;

"PROCEDURE" MATVEC(U,AU); "ARRAY" AU, U;
"BEGIN" "INTEGER" I, EL, K1, K2, L; "ARRAY" AUX[1:M4];
    INIVEC(1,MSQ,AU,0);
    "COMMENT" DE RANDWAARDEN WORDEN OP AUX OVERGESCHREVEN
       EN WORDEN TIJDELIJK OP NUL GESTELD;
    "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
    "BEGIN" AUX[I]:= U[I]; AUX[I+M]:= U[I+MSQ-M];
         AUX[I+M2]:= U[M*(I-1)+1]; AUX[I+M3]:= U[I*M];
      U[I]:= U[M*(I-1)+1]:= U[I*M]:= U[I+MSQ-M]:= 0
    "END";
    "COMMENT" DE BIJDRAGEN VAN AU WORDEN NU VIERKANT VOOR
      VIERKANT BEREKEND EN OPGETELD;
    "FOR" EL:= 1 "STEP" 1 "UNTIL" NSQ "DO"
    "BEGIN" K1:= EL//N; K2:= EL - K1*N;
      L:= "IF" K2 = 0 "THEN" (K1 - 1)*M2 + M - 2
             "ELSE" 2*(K1*M + K2) - 1;
      AU[L]:= AU[L]+(7+H2K2/4)*U[L]
             -4*(U[L+1]+U[L+M])+(U[L+M2]+U[L+2])/2;
      AU[L+2]:= AU[L+2]+(7+H2K2/4)*U[L+2]
             -4*(U[L+1]+U[L+M+2])+(U[L+M2+2]+U[L])/2;
      AU[L+M2]:= AU[L+M2]+(7+H2K2/4)*U[L+M2]
             -4*(U[L+M]+U[L+M2+1])+(U[L+M2+2]+U[L])/2;
      AU[L+M2+2]:= AU[L+M2+2]+(7+H2K2/4)*U[L+M2+2]
             -4*(U[L+M+2]+U[L+M2+1])+(U[L+M2]+U[L+2])/2;
      AU[L+1]:= AU[L+1]+(22+H2K2)*U[L+1]
             -4*(U[L]+U[L+2])-16*U[L+M+1]+2*U[L+M2+1];
      AU[L+M]:= AU[L+M]+(22+H2K2)*U[L+M]
             -4*(U[L]+U[L+M2])-16*U[L+M+1]+2*U[L+M+2];
      AU[L+M2+1]:= AU[L+M2+1]+(22+H2K2)*U[L+M2+1]
             -4*(U[L+M2+2]+U[L+M2])-16*U[L+M+1]+2*U[L+1];
      AU[L+M+2]:= AU[L+M+2]+(22+H2K2)*U[L+M+2]
             -4*(U[L+M2+2]+U[L+2])-16*U[L+M+1]+2*U[L+M];
      AU[L+M+1]:= AU[L+M+1]+(64+H2K2*4)*U[L+M+1]
             -16*(U[L+1]+U[L+M]+U[L+M+2]+U[L+M2+1])
    "END";
    "COMMENT" DE RANDWAARDEN WORDEN NU IN REKENING GEBRACHT;
    "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
    "BEGIN" L:= I;          AU[L]:= U[L]:= AUX[I];
      L:= I + MSQ - M  ;    AU[L]:= U[L]:= AUX[I+M];
      L:= M*(I - 1) + 1;    AU[L]:= U[L]:= AUX[I+M2];
      L:= I*M;              AU[L]:= U[L]:= AUX[I+M3]
    "END"
"END" MATVEC;
```

```
M:= N + N + 1; M2:= M+M; M3:= M+M2; M4:= M+M3;
MSQ:= M*M; NSQ:= N*N; H:= 1/N; H2:= H*H; H2K2:= H2*K2;
INIVEC(1,MSQ,U,0); INIVEC(1,MSQ,F,0);
"COMMENT" EVALUATIE VAN HET RECHTERLID;
"FOR" I:= M+2 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-3 "DO" F[J]:= 4*H2;
"FOR" I:= M+3 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-5 "DO" F[J]:= 2*H2;
"FOR" I:= M2+2 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-3 "DO" F[J]:= 2*H2;
"FOR" I:= M2+3 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-5 "DO" F[J]:= H2;

CONJ GRAD(MATVEC, U, F, 1, MSQ, EPS > 1,0"-10, IT, EPS);
OUTPUT(61,"("*,"("RESIDUNORM:")"-D,4D"-ZD,
  4B"(" AANTAL AANROEPEN VAN MATVEC:")"ZZD,
  4B"("AANTAL ROOSTERPUNTEN:")"ZZZD,
  4B"("MAASWIJDTE:")",DD,5/")",SQRT(EPS),IT,MSQ,H);
JSTEP:= N//5; ISTEP:= JSTEP*M;
"FOR" I:= 1 "STEP" ISTEP "UNTIL" MSQ - M + 1 "DO"
"BEGIN" OUTPUT(61,
  "("//,"("X = 0,0, 0,1, ..., 1,0; Y = ")"D,D,//")",(I//ISTEP)*0,1);
  "FOR" J:= I "STEP" JSTEP "UNTIL" I + M - 1 "DO"
  OUTPUT(61,"("BB-D,4D"-D")",U[J])
"END"
"END"
"END"
```

RESIDUNORM: 4,7659" -6 AANTAL AANROEPEN VAN MATVEC: 14 AANTAL ROOSTERPUNTEN: 121 MAASWIJDTE: ,20

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 9,9253"-3 1,5347"-2 1,8253"-2 1,9751"-2 2,0197"-2 1,9751"-2 1,8253"-2 1,5347"-2 9,9253"-3 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5347"-2 2,4523"-2 2,9741"-2 3,2480"-2 3,3305"-2 3,2480"-2 2,9741"-2 2,4523"-2 1,5347"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8253"-2 2,9741"-2 3,6542"-2 4,0170"-2 4,1276"-2 4,0170"-2 3,6542"-2 2,9741"-2 1,8253"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9751"-2 3,2480"-2 4,0170"-2 4,4314"-2 4,5586"-2 4,4314"-2 4,0170"-2 3,2480"-2 1,9751"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0197"-2 3,3305"-2 4,1276"-2 4,5586"-2 4,6914"-2 4,5586"-2 4,1276"-2 3,3305"-2 2,0197"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9751"-2 3,2480"-2 4,0170"-2 4,4314"-2 4,5586"-2 4,4314"-2 4,0170"-2 3,2480"-2 1,9751"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8253"-2 2,9741"-2 3,6542"-2 4,0170"-2 4,1276"-2 4,0170"-2 3,6542"-2 2,9741"-2 1,8253"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5347"-2 2,4523"-2 2,9741"-2 3,2480"-2 3,3305"-2 3,2480"-2 2,9741"-2 2,4523"-2 1,5347"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 9,9253"-3 1,5347"-2 1,8253"-2 1,9751"-2 2,0197"-2 1,9751"-2 1,8253"-2 1,5347"-2 9,9253"-3 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

RESIDUNORM: 5,6341" -6 AANTAL AANROEPEN VAN MATVEC: 31 AANTAL ROOSTERPUNTEN: 441 MAASWIJDTE: ,10

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0094"-2 1,5400"-2 1,8308"-2 1,9783"-2 2,0236"-2 1,9783"-2 1,8308"-2 1,5400"-2 1,0094"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5400"-2 2,4510"-2 2,9756"-2 3,2472"-2 3,3312"-2 3,2472"-2 2,9756"-2 2,4510"-2 1,5400"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8308"-2 2,9756"-2 3,6583"-2 4,0185"-2 4,1309"-2 4,0185"-2 3,6583"-2 2,9756"-2 1,8308"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9783"-2 3,2472"-2 4,0185"-2 4,4305"-2 4,5596"-2 4,4305"-2 4,0185"-2 3,2472"-2 1,9783"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0236"-2 3,3312"-2 4,1309"-2 4,5596"-2 4,6943"-2 4,5596"-2 4,1309"-2 3,3312"-2 2,0236"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9783"-2 3,2472"-2 4,0185"-2 4,4305"-2 4,5596"-2 4,4305"-2 4,0185"-2 3,2472"-2 1,9783"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8308"-2 2,9756"-2 3,6583"-2 4,0185"-2 4,1309"-2 4,0185"-2 3,6583"-2 2,9756"-2 1,8308"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5400"-2 2,4510"-2 2,9756"-2 3,2472"-2 3,3312"-2 3,2472"-2 2,9756"-2 2,4510"-2 1,5400"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0094"-2 1,5400"-2 1,8308"-2 1,9783"-2 2,0236"-2 1,9783"-2 1,8308"-2 1,5400"-2 1,0094"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

```
RESIDUNORM: 8,1133" -6    AANTAL AANROEPEN VAN MATVEC: 63    AANTAL ROOSTERPUNTEN:1681    MAASWIJDTE:,05

X = 0,0; 0,1; ...; 1,0; Y = 0,0
  0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0   0,0000" 0
X = 0,0; 0,1; ...; 1,0; Y = 0,1
  0,0000" 0   1,0091"-2   1,5399"-2   1,8308"-2   1,9783"-2   2,0235"-2   1,9783"-2   1,8308"-2   1,5399"-2   1,0091"-2   0,0000"1
```

```
"BEGIN" "INTEGER" N; "REAL" K2;
     K2:=-10; "FOR"N:=10,20,40 "DO"
"BEGIN" "INTEGER" N1, I, J, K, ISTEP, IT;
    "REAL" H, H2, H2K2, RATE, DOMEIGVAL;
    "ARRAY" U[0:N,0:N], DISCR[1:2];

    "PROCEDURE" RICHARDSON(U,LJ,UJ,LL,UL,INAP,RESIDUAL,A,B,N,
          DISCR,K,RATE,DOMEIGVAL,OUT);  "CODE" 33170;

    "PROCEDURE" INIMAT(A,B,C,D,E,F); "CODE" 31011;

    "PROCEDURE" DUPMAT(A,B,C,D,E,F); "CODE" 31035;

    "PROCEDURE" RESIDUAL(U); "ARRAY" U;
    "BEGIN" "INTEGER" I, J; "ARRAY" AU[0:N,0:N];
      INIMAT(0,N,0,N,AU,0); DUPMAT(1,N1,1,N1,AU,U);
      "FOR"I:= 1 "STEP" 1 "UNTIL" N1"DO"
      "FOR"J:= 1 "STEP" 1 "UNTIL" N1"DO"
      U[I,J]:= (4-H2K2)*AU[I,J)-(AU[I,J-1]+AU[I,J+1]+AU[I-1,J]
      +AU[I+1,J])-H2
    "END";

    "PROCEDURE" OUT(K); "VALUE" K; "INTEGER" K;
    "BEGIN"    "END";

    N1:=N-1; H:=1/N; H2:=H*H; H2K2:=H2*K2;  INIMAT(0,N,0,N,U,0);
    RICHARDSON(U,1,N1,1,N1,"TRUE",RESIDUAL,12*H2,8*(1 + H2),
    "IF" DISCR[1]<"-5 "THEN" 0 "ELSE" 150,DISCR,K,RATE,DOMEIGVAL,OUT);
    OUTPUT(61,"("*,"("AANTAL AANROEPEN VAN RESIDUAL:")"ZZD,
      4B"("CONVERGENTIESNELHEID: ")"D,4D"-D,
      4B"("RESIDUNORM: ")"D,4D"-D,4B"("MAASWIJDTE: ")",3D,3/")",
      K,RATE,DISCR[2],H); ISTEP:= N//10;
    "FOR" I:=0"STEP" ISTEP "UNTIL" N "DO"
    "BEGIN" OUTPUT(61,
"("//,"(" X = 0,0, 0,1, ...., 1,0: Y = ")"D,D,//")",(I//ISTEP)*0,1);
         "FOR"J:=0 "STEP" ISTEP "UNTIL" N "DO"
         OUTPUT(61,"("2B-D,4D"-D")",U[I,J])
    "END"
"END"
"END"
```

AANTAL AANROEPEN VAN RESIDUAL: 37 CONVERGENTIESNELHEID: 2,3988*-1 RESIDUNORM: 2,8203*-6 MAASWIJDTE: ,100

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000* 0 9,8224*-3 1,5136*-2 1,8073*-2 1,9565*-2 2,0023*-2 1,9565*-2 1,8073*-2 1,5136*-2 9,8224*-3 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000* 0 1,5136*-2 2,4163*-2 2,9398*-2 3,2120*-2 3,2964*-2 3,2120*-2 2,9398*-2 2,4163*-2 1,5136*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000* 0 1,8073*-2 2,9398*-2 3,6178*-2 3,9768*-2 4,0889*-2 3,9768*-2 3,6178*-2 2,9398*-2 1,8073*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000* 0 1,9565*-2 3,2120*-2 3,9768*-2 4,3860*-2 4,5145*-2 4,3860*-2 3,9768*-2 3,2120*-2 1,9565*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000* 0 2,0023*-2 3,2964*-2 4,0889*-2 4,5145*-2 4,6484*-2 4,5145*-2 4,0889*-2 3,2964*-2 2,0023*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000* 0 1,9565*-2 3,2120*-2 3,9768*-2 4,3860*-2 4,5145*-2 4,3860*-2 3,9768*-2 3,2120*-2 1,9565*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000* 0 1,8073*-2 2,9398*-2 3,6178*-2 3,9768*-2 4,0889*-2 3,9768*-2 3,6178*-2 2,9398*-2 1,8073*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000* 0 1,5136*-2 2,4163*-2 2,9398*-2 3,2120*-2 3,2964*-2 3,2120*-2 2,9398*-2 2,4163*-2 1,5136*-2 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000* 0 9,8224*-3 1,5136*-2 1,8073*-2 1,9565*-2 2,0023*-2 1,9565*-2 1,8073*-2 1,5136*-2 9,8224*-3 0,0000* 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0 0,0000* 0

AANTAL AANROEPEN VAN RESIDUAL: 71 CONVERGENTIESNELHEID: 1,1382"-1 RESIDUNORM: 1,3834"-6 MAASWIJDTE: ,050

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0018"-2 1,5332"-2 1,8249"-2 1,9730"-2 2,0184"-2 1,9730"-2 1,8249"-2 1,5332"-2 1,0018"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5332"-2 2,4422"-2 2,9667"-2 3,2386"-2 3,3228"-2 3,2386"-2 2,9667"-2 2,4422"-2 1,5332"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8249"-2 2,9667"-2 3,6484"-2 4,0085"-2 4,1209"-2 4,0085"-2 3,6484"-2 2,9667"-2 1,8249"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9730"-2 3,2386"-2 4,0085"-2 4,4200"-2 4,5490"-2 4,4200"-2 4,0085"-2 3,2386"-2 1,9730"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0184"-2 3,3228"-2 4,1209"-2 4,5490"-2 4,6836"-2 4,5490"-2 4,1209"-2 3,3228"-2 2,0184"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9730"-2 3,2386"-2 4,0085"-2 4,4200"-2 4,5490"-2 4,4200"-2 4,0085"-2 3,2386"-2 1,9730"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8249"-2 2,9667"-2 3,6484"-2 4,0085"-2 4,1209"-2 4,0085"-2 3,6484"-2 2,9667"-2 1,8249"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5332"-2 2,4422"-2 2,9667"-2 3,2386"-2 3,3228"-2 3,2386"-2 2,9667"-2 2,4422"-2 1,5332"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0018"-2 1,5332"-2 1,8249"-2 1,9730"-2 2,0184"-2 1,9730"-2 1,8249"-2 1,5332"-2 1,0018"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

AANTAL AANROEPEN VAN RESIDUAL: 135 CONVERGENTIESNELHEID: 5,4688"-2 RESIDUNORM: 6,2767"-7 MAASWIJDTE: ,025

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0074"-2 1,5385"-2 1,8298"-2 1,9776"-2 2,0229"-2 1,9776"-2 1,8298"-2 1,5385"-2 1,0074"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5385"-2 2,4495"-2 2,9744"-2 3,2462"-2 3,3302"-2 3,2462"-2 2,9744"-2 2,4495"-2 1,5385"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8298"-2 2,9744"-2 3,6573"-2 4,0176"-2 4,1300"-2 4,0176"-2 3,6573"-2 2,9744"-2 1,8298"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9776"-2 3,2462"-2 4,0176"-2 4,4296"-2 4,5587"-2 4,4296"-2 4,0176"-2 3,2462"-2 1,9776"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0229"-2 3,3302"-2 4,1300"-2 4,5587"-2 4,6933"-2 4,5587"-2 4,1300"-2 3,3302"-2 2,0229"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9776"-2 3,2462"-2 4,0176"-2 4,4296"-2 4,5587"-2 4,4296"-2 4,0176"-2 3,2462"-2 1,9776"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8298"-2 2,9744"-2 3,6573"-2 4,0176"-2 4,1300"-2 4,0176"-2 3,6573"-2 2,9744"-2 1,8298"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5385"-2 2,4495"-2 2,9744"-2 3,2462"-2 3,3302"-2 3,2462"-2 2,9744"-2 2,4495"-2 1,5385"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0074"-2 1,5385"-2 1,8298"-2 1,9776"-2 2,0229"-2 1,9776"-2 1,8298"-2 1,5385"-2 1,0074"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

```
"BEGIN" "INTEGER" N; "REAL" K2;
 K2:= 10; "FOR" N:= 5, 10, 20 "DO"
"BEGIN" "INTEGER" NSQ, M, M2, M3, M4, MSQ, I, J, IT, ISTEP, JSTEP;
"REAL" H, H2, H2K2, EPS; "ARRAY" U, F[1:(2*N+1)**2];

"PROCEDURE" CONJ GRAD(A, X, B, L, U, GO ON, IT, EPS); "CODE" 34220;

"PROCEDURE" INIVEC(L,U,A,X); "CODE" 31010;

"PROCEDURE" MATVEC(U,AU); "ARRAY" AU, U;
"BEGIN" "INTEGER" I, EL, K1, K2, L; "ARRAY" AUX[1:M4];
    INIVEC(1,MSQ,AU,0);
    "COMMENT" DE RANDWAARDEN WORDEN OP AUX OVERGESCHREVEN
       EN WORDEN TIJDELIJK OP NUL GESTELD;
    "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
    "BEGIN" AUX[I]:= U[I]; AUX[I+M]:= U[I+MSQ-M];
         AUX[I+M2]:= U[M*(I-1)+1]; AUX[I+M3]:= U[I*M];
      U[I]:= U[M*(I-1)+1]:= U[I*M]:= U[I+MSQ-M]:= 0
    "END";
    "COMMENT" DE BIJDRAGEN VAN AU WORDEN NU VIERKANT VOOR
      VIERKANT BEREKEND EN OPGETELD;
    "FOR" EL:= 1 "STEP" 1 "UNTIL" NSQ "DO"
    "BEGIN" K1:= EL//N; K2:= EL - K1*N;
      L:= "IF" K2 = 0 "THEN" (K1 - 1)*M2 + M - 2
                "ELSE" 2*(K1*M + K2) - 1;
      AU[L]:= AU[L]+(7+H2K2/4)*U[L]
              -4*(U[L+1]+U[L+M])+(U[L+M2]+U[L+2])/2;
      AU[L+2]:= AU[L+2]+(7+H2K2/4)*U[L+2]
              -4*(U[L+1]+U[L+M+2])+(U[L+M2+2]+U[L])/2;
      AU[L+M2]:= AU[L+M2]+(7+H2K2/4)*U[L+M2]
              -4*(U[L+M]+U[L+M2+1])+(U[L+M2+2]+U[L])/2;
      AU[L+M2+2]:= AU[L+M2+2]+(7+H2K2/4)*U[L+M2+2]
              -4*(U[L+M+2]+U[L+M2+1])+(U[L+M2]+U[L+2])/2;
      AU[L+1]:= AU[L+1]+(22+H2K2)*U[L+1]
              -4*(U[L]+U[L+2])-16*U[L+M+1]+2*U[L+M2+1];
      AU[L+M]:= AU[L+M]+(22+H2K2)*U[L+M]
              -4*(U[L]+U[L+M2])-16*U[L+M+1]+2*U[L+M+2];
      AU[L+M2+1]:= AU[L+M2+1]+(22+H2K2)*U[L+M2+1]
              -4*(U[L+M2+2]+U[L+M2])-16*U[L+M+1]+2*U[L+1];
      AU[L+M+2]:= AU[L+M+2]+(22+H2K2)*U[L+M+2]
              -4*(U[L+M2+2]+U[L+2])-16*U[L+M+1]+2*U[L+M];
      AU[L+M+1]:= AU[L+M+1]+(64+H2K2*4)*U[L+M+1]
              -16*(U[L+1]+U[L+M]+U[L+M+2]+U[L+M2+1])
    "END";
    "COMMENT" DE RANDWAARDEN WORDEN NU IN REKENING GEBRACHT;
    "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
    "BEGIN" L:= I;         AU[L]:= U[L]:= AUX[I];
      L:= I + MSQ - M  ;   AU[L]:= U[L]:= AUX[I+M];
      L:= M*(I - 1) + 1;   AU[L]:= U[L]:= AUX[I+M2];
      L:= I*M;             AU[L]:= U[L]:= AUX[I+M3]
    "END"
"END" MATVEC;
```

```
M:= N + N + 1; M2:= M+M; M3:= M+M2; M4:= M+M3;
MSQ:= M*M; NSQ:= N*N; H:= 1/N; H2:= H*H; H2K2:= H2*K2;
INIVEC(1,MSQ,U,0); INIVEC(1,MSQ,F,0);
"COMMENT" EVALUATIE VAN HET RECHTERLID;
"FOR" I:= M+2 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-3 "DO" F[J]:= 4*H2;
"FOR" I:= M+3 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-5 "DO" F[J]:= 2*H2;
"FOR" I:= M2+2 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-3 "DO" F[J]:= 2*H2;
"FOR" I:= M2+3 "STEP" M2 "UNTIL" MSQ-M "DO"
"FOR" J:=I "STEP" 2 "UNTIL" I+M-5 "DO" F[J]:= H2;

CONJ GRAD(MATVEC, U, F, 1, MSQ, EPS > 1,0"-10, IT, EPS);
OUTPUT(61,"("*,"("RESIDUNORM:")"-D,4D"-ZD,
  4B"(" AANTAL AANROEPEN VAN MATVEC:")"ZZD,
  4B"("AANTAL ROOSTERPUNTEN:")"ZZZD,
  4B"("MAASWIJDTE:")",DD,5/")",SQRT(EPS),IT,MSQ,H);
JSTEP:= N//5; ISTEP:= JSTEP*M;
"FOR" I:= 1 "STEP" ISTEP "UNTIL" MSQ - M + 1 "DO"
"BEGIN" OUTPUT(61,
  "("//,"("X = 0,0, 0,1, ..., 1,0; Y = ")"D,D,//")",(I//ISTEP)*0,1);
  "FOR" J:= I "STEP" JSTEP "UNTIL" I + M - 1 "DO"
  OUTPUT(61,"("BB-D,4D"-D")",U[J])
"END"
"END"
"END"
```

RESIDUNORM: 4,7659" -6 AANTAL AANROEPEN VAN MATVEC: 14 AANTAL ROOSTERPUNTEN: 121 MAASWIJDTE:,20

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 9,9253"-3 1,5347"-2 1,8253"-2 1,9751"-2 2,0197"-2 1,9751"-2 1,8253"-2 1,5347"-2 9,9253"-3 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5347"-2 2,4523"-2 2,9741"-2 3,2480"-2 3,3305"-2 3,2480"-2 2,9741"-2 2,4523"-2 1,5347"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8253"-2 2,9741"-2 3,6542"-2 4,0170"-2 4,1276"-2 4,0170"-2 3,6542"-2 2,9741"-2 1,8253"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9751"-2 3,2480"-2 4,0170"-2 4,4314"-2 4,5586"-2 4,4314"-2 4,0170"-2 3,2480"-2 1,9751"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0197"-2 3,3305"-2 4,1276"-2 4,5586"-2 4,6914"-2 4,5586"-2 4,1276"-2 3,3305"-2 2,0197"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9751"-2 3,2480"-2 4,0170"-2 4,4314"-2 4,5586"-2 4,4314"-2 4,0170"-2 3,2480"-2 1,9751"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8253"-2 2,9741"-2 3,6542"-2 4,0170"-2 4,1276"-2 4,0170"-2 3,6542"-2 2,9741"-2 1,8253"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5347"-2 2,4523"-2 2,9741"-2 3,2480"-2 3,3305"-2 3,2480"-2 2,9741"-2 2,4523"-2 1,5347"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 9,9253"-3 1,5347"-2 1,8253"-2 1,9751"-2 2,0197"-2 1,9751"-2 1,8253"-2 1,5347"-2 9,9253"-3 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

RESIDUNORM: 5,6341" -6 AANTAL AANROEPEN VAN MATVEC: 31 AANTAL ROOSTERPUNTEN: 441 MAASWIJDTE: ,10

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0094"-2 1,5400"-2 1,8308"-2 1,9783"-2 2,0236"-2 1,9783"-2 1,8308"-2 1,5400"-2 1,0094"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5400"-2 2,4510"-2 2,9756"-2 3,2472"-2 3,3312"-2 3,2472"-2 2,9756"-2 2,4510"-2 1,5400"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8308"-2 2,9756"-2 3,6583"-2 4,0185"-2 4,1309"-2 4,0185"-2 3,6583"-2 2,9756"-2 1,8308"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9783"-2 3,2472"-2 4,0185"-2 4,4305"-2 4,5596"-2 4,4305"-2 4,0185"-2 3,2472"-2 1,9783"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0236"-2 3,3312"-2 4,1309"-2 4,5596"-2 4,6943"-2 4,5596"-2 4,1309"-2 3,3312"-2 2,0236"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9783"-2 3,2472"-2 4,0185"-2 4,4305"-2 4,5596"-2 4,4305"-2 4,0185"-2 3,2472"-2 1,9783"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8308"-2 2,9756"-2 3,6583"-2 4,0185"-2 4,1309"-2 4,0185"-2 3,6583"-2 2,9756"-2 1,8308"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5400"-2 2,4510"-2 2,9756"-2 3,2472"-2 3,3312"-2 3,2472"-2 2,9756"-2 2,4510"-2 1,5400"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0094"-2 1,5400"-2 1,8308"-2 1,9783"-2 2,0236"-2 1,9783"-2 1,8308"-2 1,5400"-2 1,0094"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

RESIDUNORM: 8,1133" -6 AANTAL AANROEPEN VAN MATVEC: 63 AANTAL ROOSTERPUNTEN:1681 MAASWIJDTE:,05

X = 0,0, 0,1, ..., 1,0; Y = 0,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,1

0,0000" 0 1,0091"-2 1,5399"-2 1,8308"-2 1,9783"-2 2,0235"-2 1,9783"-2 1,8308"-2 1,5399"-2 1,0091"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,2

0,0000" 0 1,5399"-2 2,4509"-2 2,9755"-2 3,2471"-2 3,3311"-2 3,2471"-2 2,9755"-2 2,4509"-2 1,5399"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,3

0,0000" 0 1,8308"-2 2,9755"-2 3,6582"-2 4,0185"-2 4,1308"-2 4,0185"-2 3,6582"-2 2,9755"-2 1,8308"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,4

0,0000" 0 1,9783"-2 3,2471"-2 4,0185"-2 4,4304"-2 4,5595"-2 4,4304"-2 4,0185"-2 3,2471"-2 1,9783"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,5

0,0000" 0 2,0235"-2 3,3311"-2 4,1308"-2 4,5595"-2 4,6942"-2 4,5595"-2 4,1308"-2 3,3311"-2 2,0235"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,6

0,0000" 0 1,9783"-2 3,2471"-2 4,0185"-2 4,4304"-2 4,5595"-2 4,4304"-2 4,0185"-2 3,2471"-2 1,9783"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,7

0,0000" 0 1,8308"-2 2,9755"-2 3,6582"-2 4,0185"-2 4,1308"-2 4,0185"-2 3,6582"-2 2,9755"-2 1,8308"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,8

0,0000" 0 1,5399"-2 2,4509"-2 2,9755"-2 3,2471"-2 3,3311"-2 3,2471"-2 2,9755"-2 2,4509"-2 1,5399"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 0,9

0,0000" 0 1,0091"-2 1,5399"-2 1,8308"-2 1,9783"-2 2,0235"-2 1,9783"-2 1,8308"-2 1,5399"-2 1,0091"-2 0,0000" 0

X = 0,0, 0,1, ..., 1,0; Y = 1,0

0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0 0,0000" 0

UITGAVEN IN DE SERIE MC SYLLABUS

Onderstaande uitgaven zijn verkrijgbaar bij het Mathematisch Centrum, 2e Boerhaavestraat 49 te Amsterdam-1005, tel. 020-947272.

MCS 1.1 F. GÖBEL & J. VAN DE LUNE, *Leergang Besliskunde, deel 1: Wiskundige basiskennis*, 1965. ISBN 90 6196 014 2.

MCS 1.2 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 2: Kansberekening*, 1965. ISBN 90 6196 015 0.

MCS 1.3 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 3: Statistiek*, 1966. ISBN 90 6196 016 9.

MCS 1.4 G. DE LEVE & W. MOLENAAR, *Leergang Besliskunde, deel 4: Markovketens, en wachttijden*, 1966. ISBN 90 6196 017 7.

MCS 1.5 J. KRIENS & G. DE LEVE, *Leergang Besliskunde, deel 5: Inleiding tot de mathematische besliskunde*, 1966. ISBN 90 6196 018 5.

MCS 1.6a B. DORHOUT & J. KRIENS, *Leergang Besliskunde, deel 6a: Wiskundige programmering 1*, 1968. ISBN 90 6196 032 0.

MCS 1.7a G. DE LEVE, *Leergang Besliskunde, deel 7a: Dynamische programmering 1*, 1968. ISBN 90 6196 033 9.

MCS 1.7b G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7b: Dynamische programmering 2*, 1970. ISBN 90 6196 055 X.

MCS 1.7c G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7c: Dynamische programmering 3*, 1971. ISBN 90 6196 066 5.

MCS 1.8 J. KRIENS, F. GÖBEL & W. MOLENAAR, *Leergang Besliskunde, deel 8: Minimaxmethode, netwerkplanning, simulatie*, 1968. ISBN 90 6196 034 7.

MCS 2.1 G.J.R. FÖRCH, P.J. VAN DER HOUWEN & R.P. VAN DE RIET, *Colloquium stabiliteit van differentieschema's, deel 1*, 1967. ISBN 90 6196 023 1.

MCS 2.2 L. DEKKER, T.J. DEKKER, P.J. VAN DER HOUWEN & M.N. SPIJKER, *Colloquium stabiliteit van differentieschema's, deel 2*, 1968. ISBN 90 6196 035 5.

MCS 3.1 H.A. LAUWERIER, *Randwaardeproblemen, deel 1*, 1967. ISBN 90 6196 024 X.

MCS 3.2 H.A. LAUWERIER, *Randwaardeproblemen, deel 2*, 1968. ISBN 90 6196 036 3.

MCS 3.3 H.A. LAUWERIER, *Randwaardeproblemen, deel 3*, 1968. ISBN 90 6196 043 6.

MCS 4 H.A. LAUWERIER, *Representaties van groepen*, 1968. ISBN 90 6196 037 1.

MCS 5 J.H. VAN LINT, J.J. SEIDEL & P.C. BAAYEN, *Colloquium discrete wiskunde*, 1968. ISBN 90 6196 044 4.

MCS 6 K.K. KOKSMA, *Cursus ALGOL 60*, 1969. ISBN 90 6196 045 2.

MCS 7.1 *Colloquium Moderne rekenmachines, deel 1*, 1969. ISBN 90 6196 046 0.

MCS 7.2 *Colloquium Moderne rekenmachines, deel 2*, 1969. ISBN 90 6196 047 9.

MCS 8 H. BAVINCK & J. GRASMAN, *Relaxatietrillingen*, 1969. ISBN 90 6196 056 8.

MCS 9.1 T.M.T. COOLEN, G.J.R. FÖRCH, E.M. DE JAGER & H.G.J. PIJLS, *Elliptische differentiaalvergelijkingen, deel 1*, 1970. ISBN 90 6196 048 7.

MCS 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. ISBN 90 6196 049 5.

MCS 10 J. FABIUS & W.R. VAN ZWET, *Grondbegrippen van de waarschijnlijkheidsrekening*, 1970. ISBN 90 6196 057 6.

MCS 11 H. BART, M.A. KAASHOEK, H.G.J. PIJLS, W.J. DE SCHIPPER & J. DE VRIES, *Colloquium Halfalgebra's en positieve operatoren*, 1971. ISBN 90 6196 067 3.

MCS 12 T.J. DEKKER, *Numerieke algebra*, 1971. ISBN 90 6196 068 1.

MCS 13 F.E.J. KRUSEMAN ARETZ, *Programmeren voor rekenautomaten; De MC ALGOL 60 vertaler voor de EL X8*, 1971. ISBN 90 6196 069 X.

MCS 14 H. BAVINCK, W. GAUTSCHI & G.M. WILLEMS, *Colloquium Approximatietheorie*, 1971. ISBN 90 6196 070 3.

MCS 15.1 T.J. DEKKER, P.W. HEMKER & P.J. VAN DER HOUWEN, *Colloquium Stijve differentiaalvergelijkingen, deel 1*, 1972. ISBN 90 6196 078 9.

MCS 15.2 P.A. BEENTJES, K. DEKKER, H.C. HEMKER, S.P.N. VAN KAMPEN & G.M. WILLEMS, *Colloquium Stijve differentiaalvergelijkingen, deel 2*, 1973. ISBN 90 6196 079 7.

MCS 15.3 P.A. BEENTJES, K. DEKKER, P.W. HEMKER & M. VAN VELDHUIZEN, *Colloquium Stijve differentiaalvergelijkingen, deel 3*, 1975. ISBN 90 6196 118 1.

MCS 16.1 L. GEURTS, *Cursus Programmeren, deel 1: De elementen van het programmeren*, 1973. ISBN 90 6196 080 0.

MCS 16.2 L. GEURTS, *Cursus Programmeren, deel 2: De programmeertaal ALGOL 60*, 1973. ISBN 90 6196 087 8.

MCS 17.1 P.S. STOBBE, *Lineaire algebra, deel 1*, 1974. ISBN 90 6196 090 8.

MCS 17.2 P.S. STOBBE, *Lineaire algebra, deel 2*, 1974. ISBN 90 6196 091 6.

MCS 17.3 N.M. TEMME, *Lineaire algebra, deel 3*, 1976. ISBN 90 6196 123 8.

MCS 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*, 1974. ISBN 90 6196 092 4.

MCS 19 A. HORDIJK, R. POTHARST & J.TH. RUNNENBURG, *Optimaal stoppen van Markovketens*, 1974. ISBN 90 6196 093 2.

MCS 20 T.M.T. COOLEN, P.W. HEMKER, P.J. VAN DER HOUWEN & E. SLAGT, *ALGOL 60 procedures voor begin- en randwaardeproblemen*, 1976. ISBN 90 6196 094 0.

MCS 21 J.W. DE BAKKER (red.), *Colloquium Programmacorrectheid*, 1975. ISBN 90 6196 103 3.

* MCS 22 R. HELMERS, F.H. RUYMGAART, M.C.A. VAN ZUYLEN & J. OOSTERHOOF, *Asymptotische methoden in de toetsingstheorie toepassingen van naburigheid*, 1976. ISBN 90 6196 104 1.

MCS 23.1 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 1*, 1976. ISBN 90 6196 105 X.

* MCS 23.2 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 2*, 1976. ISBN 90 6196 115 7.

* MCS 24.1 P.J. VAN DER HOUWEN, *Numerieke integratie van differentiaalvergelijkingen, deel 1: Eenstapsmethoden* , 1974. ISBN 90 6196 106 8.

MCS 25 *Colloquium Structuur van Programmeertalen*, 1976. ISBN 90 6196 116 5.

MCS 26.1 N.M. TEMME (red.), *Nonlinear Analysis, volume 1*, 1976. ISBN 90 6196 117 3.

MCS 26.2 N.M. TEMME (red.), *Nonlinear Analysis, volume 2*, 1976. ISBN 90 6196 121 1.

MCS 27 M. BAKKER, P.W. HEMKER, P.J. VAN DER HOUWEN, S.J. POLAK & M. VAN VELDHUIZEN, *Colloquium Discreteringmethoden*, 1976. ISBN 90 6196 124 6.

* MCS 28 O. DIEKMAN, N.M. TEMME (EDS.), *Nonlinear Diffusion Problems*, 1976. ISBN 90 6196 126 2.

MCS 29.1 J.C.P. BUS (red.), *Numerieke Programmatuur, deel 1A, deel 1B*, 1976. ISBN 90 6196 128 9.

* MCS 29.2 J.C.P. Bus (red.), *Numerieke Programmatuur, deel 2*, ISBN 90 6196 144 0.

MCS 31 J.H. VAN LINT (red.), *Inleiding in de Coderingstheorie*, 1976. ISBN 90 6196 136 X.

MCS 32 L. GEURTS (red.), *Colloquium Bedrijfssystemen*, 1976. ISBN 90 6196 137 8.

* MCS 33 P.J. VAN DER HOUWEN, *Differentieschema's voor de berekening van waterstanden in zeeën en rivieren*, ISBN 90 6196 138 6.

* MCS 34 J. HEMELRIJK, *Orienterende cursus mathematische statistiek*, IBSN 90 6196 139 4.

De met een * gemerkte uitgaven moeten nog verschijnen.