MC SYLLABUS 20

T.M.T.COOLEN
P.W.HEMKER
P.J.VAN DER HOUWEN
E.SLAGT

ALGOL 60 PROCEDURES VOOR BEGIN- EN RANDWAARDEPROBLEMEN

MATHEMATISCH CENTRUM AMSTERDAM 1976

AMS(MOS) subject classification scheme (1970): 6560, 6561, 6565, 6566, 6567, 6568

ISBN 90 6196 094 0

Inhoud

Voorwoord

Deze syllabus dient als leidraad voor een werkweek "Numeriek oplossen van differentiaalvergelijkingen" georganiseerd door het Mathematisch Centrum. Behandeld worden een aantal numerieke oplossingsmethoden voor gewone, stijve en partiële differentiaalvergelijkingen. Getracht is een boekje te schrijven, dat als wegwijzer kan dienen voor hen die in de praktijk met het oplossen van dergelijke vergelijkingen geconfronteerd worden. Hiertoe zijn een flink aantal volledig uitgewerkte voorbeelden en complete ALGOL 60 programma's opgenomen.

In hoofdstuk I worden lineaire meerstapsmethoden behandeld. Een ALGOL 60 versie van deze methoden wordt gegeven in de procedure "multistep", waarna een uitgewerkt voorbeeld volgt.

In hoofdstuk II wordt een methode aangegeven om stijve differentiaalvergelijkingen efficiënt op te lossen. De procedure "exponential fitted runge kutta" wordt beschreven en aan de hand van een voorbeeld toegelicht.

In hoofdstuk III wordt aangegeven hoe hyperbolische en parabolische differentiaalvergelijkingen door middel van partiële discretisatie in een stelsel gewone differentiaalvergelijkingen omgezet kunnen worden, zodat gestabiliseerde Runge-Kutta methoden toepasbaar zijn. De procedure "modified runge kutta" wordt beschreven, terwijl ook hier uitgebreide voorbeelden zijn opgenomen.

In hoofdstuk IV komen tenslotte partiële differentiaalvergelijkingen van het elliptische type ter sprake. Hier wordt de versnelde methode van Richardson voor het oplossen gebruikt. De procedures "richardson" en "elimination" worden tezamen met een uitgewerkt voorbeeld uitvoerig behandeld. Aan het eind van ieder hoofdstuk zijn een aantal vraagstukken als oefenstof opgenomen.

De gebruiker van deze handleiding wordt er tenslotte nog op gewezen, dat de in dit boekje beschreven procedures, die in eerst instantie gemaakt zijn voor de X8 rekenmachine van het Mathematisch Centrum, geconverteerd zijn of worden voor de SARA rekenmachine, de Cyber 73, waarbij hier en daar nog enkele modificaties zijn aangebracht. De procedures zullen volledig gedocumenteerd in de bibliotheek NUMAL verkrijgbaar zijn.

Amsterdam, juli 1972

E. Slagt

MATHEMATICAL CENTRE SYLLABUS 20, 1975, I, 1-24.

LINEAIRE MEERSTAPSMETHODEN VOOR GEWONE DIFFERENTIAALVERGELIJKINGEN

P.W. HEMKER

Een vergelijking van de vorm

$$(1.1) \qquad F(x,y,\frac{dy}{dx},\frac{d^2y}{dx^2},\ldots,\frac{d^ny}{dx^n}) = 0$$

is een n-de orde gewone differentiaalvergelijking. De orde van de differentiaalvergelijking, n, is de orde van de hoogste afgeleide welke in (1.1) voorkomt. Als alle nevenvoorwaarden, zoals de waarden van $y,y',\ldots,y^{(n-1)}$, worden gegeven voor dezelfde waarde van de onafhankelijke variabele x_0, is het probleem een *beginwaardeprobleem*. Als de noodzakelijke nevenvoorwaarden niet allen voor hetzelfde punt gegeven worden noemen we het probleem een *randwaardeprobleem*.
Iedere n-de orde gewone differentiaalvergelijking kan geschreven worden als een stelsel eerste orde differentiaalvergelijkingen. Hiervoor definiëren we de variabelen

$$\begin{aligned} \frac{dy}{dx} &= v_1 \\ \frac{dv_1}{dx} &= v_2 = \frac{d^2y}{dx^2} \\ &\vdots \\ \frac{dv_{n-1}}{dx} &= v_n = \frac{d^ny}{dx} \end{aligned}$$

en substitueren we $v_1, v_2,\ldots, v_{n-1}$ en $d^ny/dx^n = dv_{n-1}/dx$ in de vergelijking (1.1). Hieruit blijkt dat we ons kunnen beperken tot het oplossen van stelsels eerste orde differentiaalvergelijkingen.

Opmerking

In enkele gevallen waarin vergelijking (1.1) een bijzondere structuur

bezit, kan het nuttig zijn deze reductie tot een stelsel niet uit te voeren, maar gebruik te maken van deze bijzondere structuur. Dit kan in het bijzonder het geval zijn als een aantal van de afgeleiden d^iy/dx^i $(0 \le i < n)$ niet expliciet in (1.1) voorkomt.

1.1. De constructie van een oplossing

Laten we ons eerst tot een enkele eerste orde differentiaalvergelijking beperken. Het zal later blijken dat uitbreiden tot een stelsel differentiaalvergelijkingen geen extra moeilijkheden oplevert. Zij gegeven de differentiaalvergelijking

$$\frac{dy}{dx} = f(x,y) , \qquad f: \mathbb{R} \times \mathbb{R} \to \mathbb{R} \tag{1.3}$$

We kunnen de functie f in een tekening voorstellen als een verzameling van richtingen voor iedere waarde van de onafhankelijke variabele x en van de afhankelijke variabele y. We nemen aan dat f een continue functie is van x en y, welke bovendien aan een Lipschitz-voorwaarde voldoet:

$$\exists k > 0 \quad \forall x, y_1, y_2 \quad |f(x,y_1) - f(x,y_2)| \le k |y_1 - y_2| . \tag{1.4}$$

Als nu een punt A gegeven is, kunnen we de oplossing eenvoudig tekenen door, met toenemende waarden van x, een baan in het x-y-vlak te volgen waarvan de richting gelijk is aan die welke voorgeschreven wordt door f.

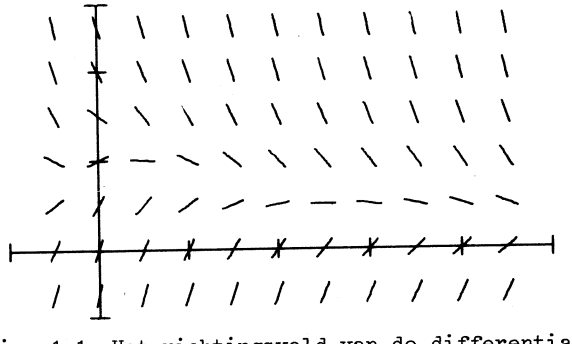


fig. 1.1. Het richtingsveld van de differentiaalvergelijking
$dy/dx = -2.5y + (5x+3)(x+1)^{-2}$

Numerieke methoden voor gewone beginwaardeproblemen komen er dan ook op neer dat we de oplossing van de differentiaalvergelijking benaderen met een kromme die zich dicht langs de richtingen van het richtingsveld aansluit. Deze kromme wordt bepaald door stap voor stap een volgend punt (x_{n+1},y_{n+1}) te berekenen.

Wanneer men te maken krijgt met een beginwaardeprobleem waarvan de structuur niet direct een eigenaardig karakter vertoont, is het een goed gebruik in eerste instantie te trachten dit probleem op te lossen met een standaardmethode. Een aantal algoritmen welke zich hier goed voor lenen, zijn bijvoorbeeld de Runge-Kutta methoden zoals ze staan beschreven in Zonneveld [1964]. (Tijdens deze cursus zullen we op deze methoden niet verder ingaan.) Een aantal moeilijkheden kan zich echter voordoen. Het meest voorkomende is het verschijnsel van de "*stijve differentiaalvergelijking*". Dit treedt op wanneer het gestelde probleem een analytische oplossing heeft waarvan sommige componenten een zeer stabiel karakter hebben. In dat geval zullen deze standaardmethoden, vanwege de eis van numerieke stabiliteit, gedwongen zijn de numerieke integratie voort te zetten met zeer kleine staplengte. De belangrijke begrippen *analytische (inherente) stabiliteit* en *numerieke stabiliteit*, welke hier naar voren komen, zullen we hieronder toelichten.

1.2. Analytische en numerieke stabiliteit

De oplossing $y(x)$ van een differentiaalvergelijking hangt, behalve van de functie f (zie (1.3)), ook af van een gegeven beginvoorwaarde: $y(a)= b$. Beschouwen we twee oplossingen y_1 en y_2 van een differentiaalvergelijking met beginvoorwaarden welke een weinig verschillen

$$\frac{d}{dx}y_1(x) = f(x,y), \qquad y_1(a)= b$$

$$\frac{d}{dx}y_2(x) = f(x,y), \qquad y_2(a)= b + \epsilon$$

dan noemen we de differentiaalvergelijking *analytisch (of inherent) stabiel op het interval* (a,c) als geldt

$$x \in (a,c) \Rightarrow |y_1(x) - y_2(x)| < \epsilon$$

Een belangrijke grootheid, welke de stabiliteit in de omgeving van een punt in het x-y vlak bepaalt (*locale stabiliteit*), is de partiële afgeleide f_y.

Schrijven we

$$\frac{d}{dx}(y_1(x) - y_2(x)) = f(x,y_1) - f(x,y_2) =$$

$$= f_y(x,y_1)(y_1-y_2) + o(y_1-y_2),$$

dan zien we dat het gedrag van de verschilfunctie $v(x)= y_1(x) - y_2(x)$ weer beschreven wordt door een differentiaalvergelijking en dat voor kleine waarden van $v(a) = \epsilon$ bij benadering geldt

$$\frac{d}{dx} v(x) = f_y(x,y)\, v(x) \tag{1.5}$$

ofwel

$$v(x) = v(a).e^{\int_a^x f_y(\xi,y(\xi))d\xi}.$$

Hieruit volgt direct dat een differentiaalvergelijking in ieder geval stabiel is op die plaats waar $f_y(x,y) < 0$

Voor een stelsel differentiaalvergelijkingen

$$\frac{d}{dx} y_i = f_i(x,y_1,y_2,\ldots,y_n), \qquad 1 \leq i \leq n,$$

laat het begrip *partiële afgeleide* zich uitbreiden tot het begrip *Jacobiaan*, de matrix van partiële afgeleiden

$$J(x,y) = (\partial f_i/\partial y_j)(x,y).$$

Het stabiliteitsgedrag wordt geheel bepaald door deze Jacobiaan. Het stelsel differentiaalvergelijkingen is nu stabiel als voor alle *eigenwaarden van de Jacobiaan* λ_i geldt Re $\lambda_i < 0$.

Wanneer de partiële afgeleide $f_y(x,y)$(of de Jacobiaan $J(x,y)$) onafhankelijk is van de argumenten x en y spreekt men van *lineaire* (een lineair stelsel) *differentiaalvergelijkingen*. Een niet-lineaire vergelijking heeft een stabiliteitsgedrag dat afhankelijk is van de plaats in het x-y vlak.

Numerieke stabiliteit is een wezenlijk ander begrip. Zegt inherente stabiliteit iets over de analytische oplossing, numerieke stabiliteit zegt iets over de numerieke berekening. Bij een rekenproces worden telkens fouten geïntroduceerd zoals *afbreekfouten* en *afrondfouten*. Een numeriek

proces heet nu numeriek stabiel wanneer het min of meer ongevoelig is voor deze fouten, zodat geen opeenhoping van gemaakte fouten ontstaat. We onderscheiden (1) *absolute stabiliteit*: de numerieke fout wordt in absolute waarde steeds kleiner, en (2) *relatieve stabiliteit*: de numerieke fout blijft klein ten opzichte van het gewenste resultaat van de berekening. Een numeriek proces heet instabiel wanneer de gemaakte fouten versterkt worden en daardoor op den duur het gewenste resultaat kunnen verdringen.

We zullen ons op deze plaats niet bezighouden met het analyseren van het stabiliteitsgedrag van de verschillende methoden voor het oplossen van gewone differentiaalvergelijkingen. We volstaan hiertoe met verwijzen naar een aantal boeken: Henrici [1962], Gear [1971] en MC Syllabus 15.1 [1972] Een enkele opmerking over numerieke stabiliteit moet hier echter toch gemaakt worden.

Bij een theoretische behandeling van de numerieke stabiliteit van methoden voor gewone differentiaalvergelijking, worden bijna uitsluitend locaal lineaire beschouwingen gegeven, d.w.z. men beperkt zich tot een klein gebiedje in het x-y vlak (juist dat gebiedje waar men de oplossing wil construeren) en men neemt aan dat de Jacobiaan in dat gebiedje praktisch constant is.

1.3. Het stabiliteitsgebied

Het stabiliteitsgedrag van een numerieke methode, bij het oplossen van een bepaald probleem, hangt ten nauwste samen (1) met het karakter van het op te lossen probleem, met name van de Jacobiaan in de omgeving van de oplossing, $J(x,y(x))$, en (2) met de staplengte h_n waarmee de integratie uitgevoerd wordt. De stabiliteit van een bepaalde methode wordt gekarakteriseerd door een gebied S in het complexe vlak. Een numeriek stabiel proces wordt verkregen wanneer voor iedere stap uit het integratieproces en voor iedere eigenwaarde λ_i van de Jacobiaan geldt dat $h_n\lambda_{i,n} \in S$

Een belangrijk onderscheid valt te maken tussen *expliciete* methoden en *impliciete* methoden om gewone differentiaalvergelijkingen op te lossen. Bij expliciete methoden wordt in iedere stap van het integratieproces y_{n+1} berekend door een vast aantal malen de functie $f(x,y)$ te evalueren waarbij tevoren de argumenten x en y bekend zijn. Bij een impliciete methode moet voor iedere integratiestap een (in het algemeen niet-lineaire) vergelijking opgelost worden van de vorm

$$G(x_{n+1}, y_{n+1}, f(x_{n+1}, y_{n+1})) = 0.$$

Waar expliciete methoden- zoals standaard Runge-Kutta - door hun (betrekkelijke) eenvoud de voorkeur schijnen te verdienen, blijken deze alle een begrensd stabiliteitsgebied te bezitten. Voor beginwaardeproblemen waar $|\lambda_i|$ grote waarden aanneemt, kan het voorkomen dat de integratie met kleinere stappen moet worden uitgevoerd dan men op grond van nauwkeurigheidscriteria wenst (stijve differentiaalvergelijkingen). Sommige impliciete methoden hebben dit nadeel niet.

1.4. Extra informatie voor efficiënter rekenen

Hoewel het in principe mogelijk is een beginwaardeprobleem numeriek te integreren met een algoritme die als enige informatie over het gestelde probleem de definiërende functie f en de bijbehorende startwaarden gebruikt, zullen we dikwijls aan de algoritme extra informatie willen verschaffen om de berekening efficiënter te kunnen uitvoeren. Informatie welke hiervoor o.a. in aanmerking komt is
(1) een analytische uitdrukking voor de Jacobiaan,
(2) de ligging van de eigenwaarden in het complexe vlak.
Dikwijls is een redelijke benadering van één van deze twee al voldoende om de efficiency van de berekening aanzienlijk te verhogen. De Jacobiaan wordt gebruikt in semi-impliciete methoden en wordt ook bij impliciete methoden gebruikt om de vergelijking G = 0 op te lossen. Wanneer de ligging van de eigenwaarden bekend is, kan dit gebruikt worden om de techniek van het "exponentieel fitten" toe te passen. Deze techniek die ook zeer geschikt is voor het oplossen van stelsels gewone differentiaalvergelijkingen zal behandeld worden in een volgend hoofdstuk.

1.5. Lineaire meerstapsmethoden

In dit hoofdstuk zullen we een beschrijving geven van de meerstapsmethoden volgens Adams-Moulton [1883,1926] en Curtiss-Hirschfelder [1952], zoals ze later zijn uitgewerkt door Nordsieck [1962] en Gear [1968]. Deze methoden behoren tot de lineaire meerstapsmethoden, een klasse van methoden waar een uitgebreide theorie over bestaat. (Henrici [1962] Gear [1971], MC Syllabus 15.1 [1972].) De nieuwe ontwikkelingen zijn vooral gericht op het efficiënt oplossen van *stelsels stijve* differenti-

aalvergelijkingen. We willen hier op de theoretische aspecten niet ingaan, deze kunnen gevonden worden in bovengenoemde boeken. We zullen hier de nadruk leggen op de structuur en het gebruik van procedures zoals ze worden gegeven door Gear [1971] (in FORTRAN) en door Hemker [1971](in ALGOL 60).

Beschouw het stelsel differentiaalvergelijkingen

$$\vec{y}' = \vec{f}(\vec{y}) = \vec{f}(y_1, y_2, \ldots, y_m), \tag{1.6}$$

geschreven in vectornotatie. Zij $h > 0$ een vast gekozen staplengte en laat

$$x_n = x_0 + nh, \qquad n = 0,1,2,\ldots,N,$$

een rij punten zijn met gelijke tussenruimte. We schrijven $\vec{y}_n$ voor de benadering van $\vec{y}(x_n)$ $(n = 0,1,2,\ldots,N)$, de oplossing van (1.6) met een gegeven beginvoorwaarde $\vec{y}_0$:

$$\vec{y}_n = \vec{y}(x_n) + \vec{\varepsilon}_n = [y_{1n}, y_{2n}, \ldots, y_{mn}].$$

Wanneer we aannemen dat $\vec{\varepsilon}_0 = \vec{\varepsilon}_1 = \ldots = \vec{\varepsilon}_{n-1} = 0$ dan heet $\vec{\varepsilon}_n$ de *locale discretiseringsfout* van de methode en de methode heet *nauwkeurig van de orde* p als geldt

$$\vec{\varepsilon}_n = O(h^{p+1}).$$

Een lineaire meerstapsmethode om (1.6) op te lossen, wordt gedefinieerd door de vectorvergelijking

$$\sum_{i=0}^{k} \alpha_i \vec{y}_{n-i} + h \sum_{i=0}^{k} \beta_k \vec{f}(\vec{y}_{n-i}) = \vec{0} \qquad (n \geq k) \tag{1.7}$$

Waarbij k startwaarden $\vec{y}_0, \vec{y}_1, \ldots, \vec{y}_{k-1}$ nodig zijn. Een methode wordt gedefinieerd door de keuze van de parameters α_i, β_i $(i=0,1,\ldots,k)$; er bestaan methoden die een hoge orde van nauwkeurigheid bezitten en stabiel zijn voor voldoend kleine h. Twee typen van deze methoden komen als bijzonder gunstig naar voren. Dit zijn (1) de methoden welke de waarden $f(\vec{y}_{n-i})$ $(0 \leq i \leq k)$ door een polynoom verbinden (de Adams-Moulton methoden; $\alpha_i = 0$ voor $i = 1,\ldots,k$). Deze bereiken de orde van nauwkeurigheid $p = k+1$ maar bezitten een - met hogere orde afnemend - begrensd stabiliteitsge-

bied. En daarnaast (2) de methoden welke de waarden y_{n-1} $(0 \le i \le k)$ met een polynoom verbinden (de Curtiss-Hirschfelder methoden; $\beta_i = 0$ voor $i = 1,2,\ldots,k$). Deze laatste methoden, met een orde van nauwkeurigheid $p = k$, zijn slechts stabiel voor $p \le 6$. Het stabiliteitsgebied van deze methoden strekt zich echter (voor $p \le 6$) uit over de gehele negatieve reële as.

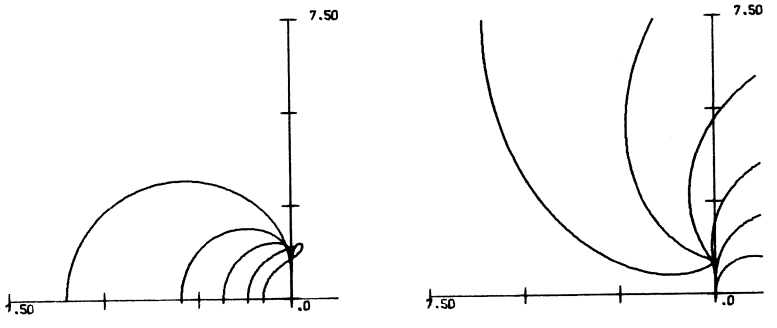


fig. 1.2. Stabiliteitsgebieden voor Adams-Moulton methoden (links) en voor Curtiss-Hirschfelder methoden (rechts)

De voorwaarden voor consistentie en het bepalen van stabiliteitsgebieden voor de lineaire meerstapsmethoden wordt uitgebreid behandeld in hoofdstuk IV van MC Syllabus 15.1 [1972].

We beschouwen de k-stapsmethoden (1.7) met $\alpha_0 \neq 0$, deze kunnen worden geschreven als

$$(1.8) \qquad \vec{y}_n = h\,\vec{f}(\vec{y}_n) + \vec{\Phi}, \qquad \beta = -\beta_0/\alpha_0,$$

waarin $\vec{\Phi}$ een lineare combinatie is van $\vec{y}_{n-1},\vec{y}_{n-2},\ldots,\vec{y}_{n-k},\vec{f}(\vec{y}_{n-1}),\ldots,\vec{f}(\vec{y}_{n-k})$. Als $\beta = 0$ is de methode expliciet en levert de berekening van y_n geen moeilijkheden op. Als $\beta \neq 0$ is de methode impliciet en is (1.8) een stelsel van m (niet-lineaire) vergelijkingen met m onbekenden. De gebruikelijke iteratieve methode om dit stelsel op te lossen wordt gegeven door

$$(1.9) \qquad {}_{r+1}\vec{y}_n = h\beta\vec{f}({}_r\vec{y}_n) + \vec{\Phi}, \qquad r = 0,1,2,\ldots$$

waarin ${}_0\vec{y}_n$ een beginapproximatie is van $\vec{y}_n$. Het is eenvoudig in te zien dat dit iteratieproces convergeert voor kleine waarden van $|h\lambda_i|$ ($\lambda_1,\lambda_2,\ldots,\lambda_n$ de eigenwaarden van de Jacobiaan). Voor stijve stelsels houdt dit echter in dat de staplengte klein gekozen moet worden. Een an-

dere, geschikte, iteratieve methode om (1.8) op te lossen is het gewijzigde Newton-Raphson iteratieproces, dat gedefinieerd wordt door

$$(1.10) \qquad {}_{r+1}\vec{y}_n = {}_r\vec{y}_n - (I-h\beta J^*)^{-1}\,({}_r\vec{y}_n - \vec{\phi} - h\beta \vec{f}({}_r\vec{y}_n)),$$

waarin J^* een benadering is van de Jacobiaan $J({}_r\vec{y}_n)$. Dit schema convergeert als alle eigenwaarden van de inverse matrix in absolute waarde kleiner dan één zijn. Zijn $\lambda_1,\lambda_2,\ldots,\lambda_m$ de eigenwaarden van J^* dan zijn

$$\frac{1}{1 - h\beta\lambda_i} \qquad i = 1,2,\ldots,m,$$

de eigenwaarden van $(1-h\beta J^*)^{-1}$. De iteratiemethode is dan geschikt voor stijve stelsels. We merken op dat men bij het uitvoeren van dit proces de beschikking moet hebben over een benadering van de Jacobiaan J.

De vectorfunctie $\vec{\phi}$, gedefinieerd door (1.8) is in het geval van de Curtiss-Hirschfelder methoden een lineaire combinatie van $\vec{y}_{n-i}$ $(0<i\leq k)$ en in het geval van de Adams-Moulton methoden een lineaire combinatie van y_{n-1} en $f(y_{n-i})$ $(0<i\leq k)$. Een lineaire transformatie voert bij de Curtiss-Hirschfelder methoden de vector $(y_{n-1},y_{n-2},\ldots,y_{n-k})$ over in de vector der achterwaartse differenties

$$(y_{n-1},\nabla y_{n-1},\nabla^2 y_{n-1}/2,\ldots,\nabla^{k-1} y_{n-1}/(k-1)!) \cong$$

$$\cong (y_{n-1},hy'_{n-1},\frac{h^2}{2}y''_{n-1},\ldots,h^{k-1}y_{n-1}^{(k-1)}/(k-1)!).$$

Bij de Adams-Moulton methode kan $(y_{n-1},f_{n-1},f_{n-2},\ldots,f_{n-k})$ worden overgevoerd in

$$(y_{n-1},f_{n-1},\nabla f_{n-1},\ldots,\nabla^{k-1} f_{n-1}/(k-1)!) \cong$$

$$\cong (y_{n-1},hy'_{n-1},\frac{h^2}{2}y''_{n-1},\ldots,h^k y_{n-1}^{(k)}/k!).$$

We kunnen $\vec{\phi}$ derhalve ook beschouwen als een lineaire combinatie van $h^i y_{n-1}^{(i)}/i!$ $(0\leq i\leq p-1)$

We kunnen nu eenvoudig, op expliciete wijze, beginschattingen ${}_0y_n, h\,{}_0y_n,\ldots,h^{p-1}\,{}_0y_n^{(p-1)}/(p-1)!$ berekenen door directe extrapolatie, bij voorbeeld met behulp van de formules

$$ {}_0y_n = y_{n-1} + hy'_{n-1} + h^2y''_{n-1}/2 + \ldots + h^{p-1}y^{(p-1)}_{n-1}/(p-1)! $$

en

$$ h\,{}_0y'_n = \quad hy'_{n-1} + h^2y''_{n-1} + \ldots + h^{p-1}y^{(p-1)}_{n-1}/(p-2)! $$

Nadat op deze wijze beginschattingen verkregen zijn, worden $\vec{y}_n$ en $h\vec{y}'_n$ iteratief verbeterd met behulp van (1.10); ${}_r\vec{y}'_n$ (r>0) wordt gedefinieerd door

$$ h\beta\ {}_r\vec{y}'_n \overset{d}{=} h\beta\vec{f}({}_{r-1}\vec{y}_n) = {}_r\vec{y}_n - \vec{\phi}. $$

Wanneer dit iteratieproces beëindigd is kunnen de waarden $h^i y^{(i)}_{n+1}/i!$ $(2\le i\le p)$ berekend worden door de beginschattingen te corrigeren met een term $a_i \times ({}_0y_n - y_n)$. De factor a_i is afhankelijk van de gekozen methode en van de gekozen orde. (Voor het bewijs hiervan en voor de berekening van a_i zij verwezen naar MC Syllabus 15.1 [1972].)

Een volledige rekenproces in één integratiestap luidt dus als volgt:

$$ \begin{pmatrix} {}_0\vec{y}_n \\ h\ {}_0\vec{y}'_n \\ h^2\ {}_0\vec{y}''_n/2 \\ \vdots \\ h^{p-1}\ {}_0\vec{y}^{(p-1)}_n/(p-1)! \end{pmatrix} = \left(A_{ij}\right) \begin{pmatrix} \vec{y}_{n-1} \\ h\ \vec{y}'_{n-1} \\ h^2\ \vec{y}''_{n-1}/2 \\ \vdots \\ h^{p-1}\ \vec{y}^{(p-1)}_{n-1}/(p-1)! \end{pmatrix} $$

$$ \text{met } A_{ij} = \begin{cases} \binom{j}{i} & \text{als } i\le j \\ 0 & \text{als } i>j \end{cases} $$

$$ \left.\begin{aligned} {}_r\vec{d} &= (I-h\beta J^*)^{-1}\,(h\vec{f}({}_r\vec{y}_n) - h\ {}_r\vec{y}'_n) \\ {}_{r+1}\vec{y}_n &= {}_r\vec{y}_n + \beta\ {}_r\vec{d} \\ h\cdot{}_{r+1}\vec{y}'_n &= h\cdot{}_r\vec{y}'_n + {}_r\vec{d} \end{aligned}\right\} \quad r = 0,1,\ldots,R-1 $$

$$(1.11)\quad \begin{cases} \vec{y}_n = R\vec{y}_n \\ h\,\vec{y}'_n = h\,R\vec{y}_n \\ h^i\,\vec{y}^{(i)}_n/i! = h^i\,{}_0\vec{y}^{(i)}_n + a_i(hy'_n - h\,{}_0y'_n) \qquad (2\le i\le k). \end{cases}$$

Wanneer de orde van de formule bij de volgende stap hoger gekozen wordt, wordt bovendien berekend

$$h^{k+1}\,\vec{y}^{(k+1)}/(k+1)! = a_k(h\vec{y}'_n - h\,{}_0y'_n)/(k+1).$$

1.6. Een strategie voor het toepassen van lineaire meerstapsformules

We hebben nog een groot aantal vrijheden als we dit proces willen toepassen. Zo moeten we nog beslissen

(1) wanneer we een nieuwe J^* zullen berekenen,
(2) welke methode we zullen toepassen: Adams-Moulton of Curtiss-Hirschfelder, en
(3) met welke orde en welke staplengte we zullen integreren.

In het kort zullen we de strategie vermelden die in de procedure MULTISTEP gerealiseerd is:

1. We kiezen een minimale en een maximale staplengte (een minimale staplengte is o.a. nodig om ons van een eindig proces te verzekeren).

2. Als we geen reden hebben om aan te nemen dat de vergelijking stijf is, integreren we in eerste instantie met de Adams-Moulton methode.

3 Bij niet-stijve differentiaalvergelijkingen (Adams-Moulton methode) nemen we eerst $J^* = 0$; blijkt dat het iteratieproces niet snel genoeg convergeert met de laatste J^* dan wordt de Jacobiaan ter plaatse geëvalueerd.

4. We starten de integratie met een eerste orde methode en met de minimale staplengte (een hogere orde methode kunnen we niet gebruiken omdat we niet over $h^i\,\vec{y}^{(i)}/i!$ $(i\ge2)$ beschikken).

5. We nemen minstens k equidistante stappen met een k-stapsmethode.

6. We verlagen of verhogen de orde van nauwkeurigheid nooit met meer dan één tegelijk.

7. Na een aantal stappen met een p-de orde methode berekenen we $h^{(i)}$ (i=p-1,p,p+1), d.i. de staplengte die een voorgeschreven afbreekfout (eps) zal veroorzaken bij het gebruik van de i-de orde methode. De maximale $h^{(i)}$ wordt gekozen als nieuwe staplengte, in combinatie met de bijbehorende orde i. (Enige veiligheidsmarges zijn ingebouwd om overbodig heen- en terugspringen te verhinderen.) We maken er gebruik van dat de i-de orde afbreekfout evenredig is met

$$\begin{cases} \| h^p \, y_n^{(p)}/p! \| & \text{voor } i = p-1 \\ \| {}_R\vec{y}_n - {}_0\vec{y}_n \| & \text{voor } i = p, \\ \| ({}_R\vec{y}_n - {}_0\vec{y}_n) - ({}_R\vec{y}_{n-1} - {}_0\vec{y}_{n-1}) \| & \text{voor } i = p+1. \end{cases}$$

8. In een aantal gevallen zullen we gedwongen zijn een integratiestap te verwerpen en een stap met kleinere staplengte opnieuw uit te voeren. Dit kan optreden (A) als het proces niet (voldoende snel) convergeert terwijl J^* een goede benadering is van J of (B) als de verkregen afbreekfout groter is dan de gewenste.

9. Wanneer bij de minimale staplengte de berekende afbreekfout groter blijft dan de gewenste, wordt dit waarschijnlijk veroorzaakt door stijfheid van de differentiaalvergelijking *); wanneer dit optreedt bij het gebruik van een Adams-Moulton methode, wordt overgeschakeld op een Curtiss-Hirschfelder methode, wanneer dit optreedt bij een Curtiss-Hirschfelder methode wordt verder gerekend met de backward-Euler methode met de minimale staplengte. (N.B. de backward-Euler methode is de Curtiss-Hirschfelder methode van orde 1.) Wanneer in dit laatste geval niet aan het locale foutcriterium is voldaan, wordt dit in een output-parameter gemeld.

*) Stijfheid van de differentiaalvergelijking is een begrip dat zowel afhankelijk is van de vergelijking zelf, als van de eisen die de gebruiker aan de oplossing stelt: eps en hmin.

1.7. De procedure MULTISTEP

Tot slot geven we de gebruiksaanwijzing voor ALGOL 60 procedure MULTISTEP

Declaratie

```
procedure MULTISTEP (x,xend,y,hmin,hmax,ymax,eps,
          first,dd,fxyi,i,jacij,j,n,available,
          stiff);
value hmin,hmax,eps,xend,available,n;
Boolean available,stiff,first;
integer i,j,n;
real x,xend,hmin,hmax,eps,fxyi,jacij;
array y,ymax,dd;
<procedure body>;
```

Parameters

x : <variable>;
Deze wordt o.a. gebruikt als Jensen-parameter voor fxyi en jacij; de onafhankelijke variabele;
ingang: x0 de beginwaarde van het integratie interval;

xend : <expression>;
de eindwaarde van het integratie-interval (xend > x);

y : <array identifier>; array y [0:7,1:n];
de afhankelijke variabele;
ingang: de beginwaarden voor het stelsel
differentiaalvergelijkingen
y[0,i]:= y[i] (x0);
uitgang: y(xend);

hmin,hmax: <expression>;
de minimale resp. de maximale stap waarmee de integratie wordt uitgevoerd;

eps : <expression>;
de maximaal toegestane relatieve locale fout;

ymax : <array identifier>; array ymax[1:n];
ingang : de toegestane absolute locale fout/eps;

uitgang:ymax [i] is de maximale waarde welke y[i]
tijdens het integratie proces heeft aangenomen;

first : <identifier>;
bij een eerste aanroep van de procedure moet first:≡
true opdat gestart kan worden met een eerste orde
Adams-methode. Bij het verlaten van de procedure is
first:= false zodat bij het voortzetten van de
integratie de procedure voortgaat op de wijze die
vanaf de laatste aanroep met first:= true de beste
gebleken is (een hogere orde Adams- of Curtiss-methode);

dd : <array identifier>; array dd[0:7,0:n];
throughput en meldingen:
dd[0,0] = 0 Adams-methode gebruikt,
= 1 overgeschakeld op Curtiss-methode;
dd[1,0] = 0 geen fouten opgetreden tijdens executie,
= 1 bij de huidige hmin kan de nietlineariteit van het
probleem niet gevolgd worden;
dd[2,0] : aantal stappen waarbij met de huidige hmin
de vereiste locale fout niet binnen de
gestelde grenzen bleef;
dd[3,0] : if dd[2,0] = 0 then 0 else
schatting van de maximale locale fout;

i,j : <identifier>;
worden gebruikt als Jensen-parameter voor fxyi en
jacij;

fxyi : <expression>;
een uitdrukking afhankelijk van x,y,i welke de
waarde van dy_i/dx geeft;

jacij : <expression>;
een uitdrukking afhankelijk van x,y,i,j, welke
(ad lib.) de waarde van $\partial(dy_i/dx)/\partial y_j$ geeft
(de Jacobiaan van het stelsel);

available: <Boolean expression>;
een uitdrukking welke aangeeft of door jacij de Jacobiaan ter beschikking gesteld wordt;

stiff : <Boolean expression>;
als stiff:≡ true gebruikt de procedure direct methoden welke geschikt zijn voor het oplossen van stijve differentiaalvergelijkingen, zonder eerst de Adams-Moultonmethoden te proberen;

n : <expression>;
het aantal vergelijkingen waaruit het stelsel bestaat.

1.8. Voorbeeld van een aanroep

Als voorbeeld van het gebruik van de procedure MULTISTEP geven we een gedeelte uit een programma waarin met verschillende waarden van de parameters eps en hmin de oplossing wordt berekend van het beginwaardeprobleem

$$dy/dx = (-1000 \times (y+z-2) - 0.013) \times y$$
$$dz/dx = -2500 \times (y+z-2) \times z$$

met de beginwaarden

$$y(0) = 1; \quad z(0) = 1.$$

De resultaten worden telkens afgedrukt voor x = 0.005 en x = 50

```
1   BEGIN COMMENT  VOORLOOPBAND MULT+STEP (NR 128/72) (ER 3.3.8, MEI 1972);
2
3           PROCEDURE MULT+STEP(X,XEND,Y,HMIN,HMAX,YMAX,EPS, FIRST,DD,
4                               FXYI,II,JACIJ,JJ,N,AVAILABLE,STIFF);
5           VALUE HMIN,HMAX,EPS,XEND,AVAILABLE,N;
6           BOOLEAN AVAILABLE,STIFF,FIRST; INTEGER II,JJ,N;
7           REAL X,XEND,HMIN,HMAX,EPS,FXYI,JACIJ; ARRAY Y,YMAX,DD;
8
9           BEGIN OWN BOOLEAN WITH JACOBIAN,ADAMS;
10                OWN INTEGER KOLD; OWN REAL XOLD,HOLD;
11                BOOLEAN EVALUATE,EVALUATED,CONV;
12                INTEGER I,J,L,K,KNEW,MAXORDER,FAILS,SAME;
13                REAL H,CH,CHNEW,C,TOLCONV,TOLUP,TOL,TOLDWN,ERROR,DF;
14                ARRAY CONST[1:56],A[0:7],DELTA,LAST DELTA,DF[1:N],JAC[1:N,1:N];
15                INTEGER ARRAY P[1:N];
16
17                PROCEDURE METHOD;
18                BEGIN WITH JACOBIAN:= ¬ADAMS;
19                      MAXORDER:= IF ADAMS THEN 7 ELSE 6;
20                      II:= KI:= 1;
21                      IF ADAMS THEN
22                      BEGIN FOR CONST[I]:= 1,1,1/2,1,.5,1/3,.5,1/4,1/2,1,5/12,1,.75,
23                            1/6,37.89,24.2,.375,1.11/12,1/3,1/24,53.33,37.89,1,
24                            251/720,1.25/24,35/72,5/48,1/120,70.08,53.33,.3158,
25                            95/288,1,137/120,.625,17/96,.025,1/720,87.97,70.08,
26                            .07407,19087/60480,1.1.225,203/270,49/192,7/144,
27                            7/1440,1/5040,106.9,87.97,.0139 DO II:= I + 1
28                      END ELSE
29                      BEGIN FOR CONST[I]:= 1,1,.3,2,1,2/3,1/1/3,6/4.5,1,6/11,1,
30                            6/11,1/11,9.167,7.333,0.5,.48,1,.7,,2,,.02,12.5,
31                            10.42,.1667,120/274,1,225/274,85/274,15/274,1/274,
32                            15.98,13.7,.04167,180/441,1,58/63,5/12,25/252,
33                            3/252,1/1764,19.6,17.15,.008333 DO II:= I + 1
34                      END
35                END METHOD;
36
37                PROCEDURE ORDER;
38                BEGIN IF K>MAX ORDER THEN BEGIN DD[1,0]:= 2; GOTO RETURN END;
39                      JI:= (K-1) * (K+8) / 2 + 1;
40                      FOR II:= 0 STEP 1 UNTIL K DO A[I]:= CONST[I+J];
41                      TOLUP := (EPS*CONST[J+K+1])↑2;
42                      TOL   := (EPS*CONST[J+K+2])↑2;
43                      TOLDWN:= (EPS*CONST[J+K+3])↑2;
44                      TOLCONV:= EPS/(2*N*(K+2));
45                      EVALUATE:= WITH JACOBIAN;
46                      SAME:= K+1
47                END ORDER;
48
49                PROCEDURE EVALUATE JACOBIAN;
50                BEGIN REAL R;
51                      EVALUATE:= FALSE;
52                      IF AVAILABLE THEN
53                      BEGIN R:= -A[0] * H;
54                            FOR II:= 1 STEP 1 UNTIL N DO
55                                FOR JJ:= 1 STEP 1 UNTIL N DO JAC[II,JJ]:= JACIJ * R;
56                      END ELSE
```

```
57                BEGIN REAL D; ARRAY FIXDY,FIX Y[1:N];
58                      FOR I:=1 STEP 1 UNTIL N DO
59                            BEGIN FIX Y[I]:= Y[0,I]; FIXDY[I]:= FXY1 END;
60                      FOR J:=1 STEP 1 UNTIL N DO
61                      BEGIN D:= IF EPS>ABS(FIX Y[J]) THEN EPS*EPS
62                                ELSE EPS*ABS(FIX Y[J]);
63                            Y[0,J]:= Y[0,J] + D;
64                            R:= - A[0] * H/D;
65                            FOR I:= 1 STEP 1 UNTIL N DO
66                                  JAC[I,J]:=(FXY1 - FIXDY[I]) * R;
67                            Y[0,J]:= FIX Y[J]
68                      END
69                END;
70                FOR I:= 1 STEP 1 UNTIL N DO JAC[I,I]:= JAC[I,I] + 1;
71                DET(JAC,N,P);
72                EVALUATED:= TRUE
73          END EVALUATE JACOBIAN;
74
75          PROCEDURE CALCULATE STEP AND ORDER;
76          BEGIN REAL A1,A2,A3;
77                SAME:= 10;
78                A1:= IF K=1 THEN 0 ELSE
79                     0.75*(TOLDWN/SUM(I,1,N,(Y[K,I]/YMAX[I])↑2))↑(0.5/K);
80                A2:= 0.80*(TOL    /ERROR ) ↑ (0.5/(K+1));
81                A3:= IF K=MAX ORDER  V  FAILS=0 THEN 0 ELSE
82                     0.70*(TOLUP /SUM(I,1,N,((DELTA[I]-LAST DELTA[I])/
83                           YMAX[I])↑2))↑(0.5/(K+2));
84                IF A1>A2 ∧ A1>A3 THEN BEGIN KNEW:=K-1; CHNEW:=A1 END ELSE
85                IF A2>A3         THEN BEGIN KNEW:=K   ; CHNEW:=A2 END ELSE
86                                      BEGIN KNEW:=K+1; CHNEW:=A3 END
87          END CALCULATE STEP AND ORDER;
88
89          PROCEDURE SET;
90          BEGIN XOLD:= X; HOLD:= H; KOLD:= K; CH:= 1;
91                FOR I:= 1 STEP 1 UNTIL N DO
92                FOR J:= 0 STEP 1 UNTIL K DO DD[J,I]:= Y[J,I]
93          END SET;
94
95
96          PROCEDURE RESET STEP;
97          BEGIN REAL C;
98                IF CH < HMIN/HOLD THEN CH:= HMIN/HOLD ELSE
99                IF CH > HMAX/HOLD THEN CH:= HMAX/HOLD;
100               X:= XOLD; H:= HOLD * CH; C:= 1;
101               FOR J:=0 STEP 1 UNTIL K DO
102               BEGIN FOR I:=1 STEP 1 UNTIL N DO Y[J,I]:= DD[J,I] * C;
103                     C:= C * CH
104               END;
105               SAME:= K + 1
106         END RESET STEP;
107
108         PROCEDURE BEGIN;
109         BEGIN FAILS:= 0; H:= HMIN;
110               FOR I:= 1 STEP 1 UNTIL N DO Y[1,I]:= FXY1 * H;
111               K:= 1; ORDER; SET
112         END BEGIN;
113
114
115         IF FIRST THEN
116         BEGIN FIRST:= FALSE; ADAMS:= ¬STIFF; METHOD;
```

```
117                 BEGIN; FOR I:= 1,2,3 DO DD[I,0]:= 0
118           END ELSE
119           BEGIN METHOD; K:= KOLD; ORDER; CH:= 1; RESET STEP END;
120
121           FOR L:= 0 WHILE X<XEND DO
122           BEGIN IF X+H<XEND THEN X:= X+H ELSE
123                 BEGIN CH:= (XEND-X)/H; RESET STEP; X:= XEND END;
124
125                 COMMENT PREDICTION;
126                 FOR I:=0 STEP 1 UNTIL K-1 DO
127                 FOR J:= K-1 STEP -1 UNTIL 1 DO
128                 ELMROW(1,N,J,J+1,Y,Y,1);
129                 FOR I:= 1 STEP 1 UNTIL N DO DELTA[I]:= 0;
130
131
132                 COMMENT CORRECTION AND ESTIMATION LOCAL ERROR;
133                 FOR L:=1,2,3 DO
134                 BEGIN FOR I:=1 STEP 1 UNTIL N DO DF[I]:= FXYI*H - Y[1,I];
135                       IF EVALUATE THEN EVALUATE JACOBIAN;
136                       IF WITH JACOBIAN THEN SOL(JAC,N,P,DF);
137
138                       CONV:= TRUE;
139                       FOR I:= 1 STEP 1 UNTIL N DO
140                       BEGIN DF[I]:= DF[I];
141                             Y[0,I]:= Y[0,I] + A[0]*DF[I]
142                             Y[1,I]:= Y[1,I] +      DF[I]
143                             DELTA[I]:= DELTA[I] +  DF[I]
144                             CONV:= CONV ^ ABS(DF[I]) < TOLCONV * YMAX[I]
145                       END;
146                       IF CONV THEN
147                       BEGIN ERROR:= SUM(I,1,N,(DELTA[I]/YMAX[I])↑2);
148                             EVALUATED:= FALSE; GOTO CONVERGENCE
149                       END
150                 END;
151
152                 COMMENT ACCEPTANCE OR REJECTION;
153                 IF ¬CONV THEN NO CONVERGENCE;
154                 BEGIN IF WITH JACOBIAN ^ ¬ EVALUATED THEN ELSE
155                       IF H>HMIN*1.0001 THEN
156                       BEGIN WITH JACOBIAN:= WITH JACOBIAN ∪ AVAILABLE;
157                             CH:= CH/4
158                       END ELSE
159                       IF ADAMS THEN GOTO TRY CURTISS ELSE
160                       BEGIN DD[1,0]:= 1; GOTO RETURN END;
161
162                       EVALUATE:= WITH JACOBIAN; RESET STEP
163                 END ELSE CONVERGENCE;
164
165                 IF ERROR>TOL THEN ERROR TEST NOT OK;
166                 BEGIN FAILS:= FAILS + 1;
167                       IF H>HMIN*1.0001 THEN
168                       BEGIN IF FAILS>2 THEN
169                             BEGIN K:= 0; RESET STEP; BEGIN
170                             END ELSE
171                             BEGIN CALCULATE STEP AND ORDER;
172                                   IF KNEW≠K THEN BEGIN K:= KNEW; ORDER END;
173                                   CH:= CH*CHNEW/FAILS; RESET STEP
174                             END
175                       END ELSE
176                       IF ADAMS THEN TRY CURTISS;
```

```
177                         BEGIN ADAMS:= FALSE; METHOD; ORDER; RESET STEP END ELSE
178                         IF K#1 THEN
179                         BEGIN K:=1; ORDER; RESET STEP END ELSE
180
181                         BEGIN COMMENT VIOLATE EPS CRITERION;
182                               C:= EPS * SQRT( ERROR/TOL );
183                               IF C>DD[3,0] THEN DD[3,0]:= C;
184                               DD[2,0]:= DD[2,0] + 1;
185                               GOTO ERROR TEST OK
186                         END
187                   END ELSE
188
189                   ERROR TEST OK;
190                   BEGIN FAILS:= 0;
191                         IF K>2 THEN BEGIN FOR I:=1 STEP 1 UNTIL N DO
192                               ELMCOLVEC(2,K,I,Y,A,DELTA[I]) END;
193                         FOR I:= 1 STEP 1 UNTIL N DO IF ABS(Y[0,I])>YMAX[I]
194                               THEN YMAX[I]:=ABS(Y[0,I]);
195                         SAME:= SAME - 1;
196                         IF SAME=1 THEN BEGIN FOR I:=1 STEP 1 UNTIL N DO
197                               LAST DELTA[I]:= DELTA[I] END ELSE
198                         IF SAME=0 THEN
199                         BEGIN CALCULATE STEP AND ORDER;
200                               IF CHNEW>1.1 THEN
201                               BEGIN SAME:= K + 1;
202                                     IF KNEW#K THEN
203                                     BEGIN IF KNEW>K THEN
204                                           BEGIN FOR I:=1 STEP 1 UNTIL N DO
205                                                 Y[KNEW,I]:= DELTA[I]*A[K]/KNEW
206                                           END;
207                                           K:= KNEW; ORDER
208                                     END;
209                                     IF CHNEW> HMAX/H THEN CHNEW:= HMAX/H;
210                                     H:= H * CHNEW; C:= 1;
211                                     FOR J:=1 STEP 1 UNTIL K DO
212                                     BEGIN C:= C * CHNEW;
213                                           FOR I:=1 STEP 1 UNTIL N DO
214                                           Y[J,I]:= Y[J,I]*C
215                                     END
216                               END
217                         END;
218                         SET
219                   END
220             END STEP;
221       RETURN: DD[0,0]:= IF ADAMS THEN 0 ELSE 1; DD[4,0]:= K
222       END MULTISTEP;
223
224
225
226
227
228
229
230
231
232
233
234
235
236
```

```
237
238         COMMENT  HIERVOOR STAAT DE DECLARATIE VAN PROCEDURE MUW@+8@5P;
239
240         BOOLEAN FIRST; INTEGER I,J,CF,CJ;
241         REAL X,XEND,HMIN,EPS,R,TIM;
242         ARRAY Y[0:7,1:2],YMAX[1:2],D[0:7,0:2];
243
244         REAL PROC FXY;
245         BEGIN   CF:= CF + 1; FXY:= IF I= 1
246                 THEN - Y[0,1] *(0.013 + 1000 *(Y[0,1] + Y[0,2]-2))
247                 ELSE - 2500 * Y[0,2] * (Y[0,1] + Y[0,2]-2);
248         END;
249
250         REAL PROC JAC;
251         BEGIN   CJ:= CJ + 1; JAC:= IF I= 1
252                 THEN - 1000 * Y[0,1]
253                      -(IF J = 2 THEN 0 ELSE (0.013 + 1000 *(Y[0,1] + Y[0,2] -2)))
254                 ELSE - 2500 * Y[0,2]
255                      -(IF J = 1 THEN 0 ELSE (2500 *(Y[0,1] + Y[0,2]-2)));
256         END;
257
258         NLCR; PRINTTEXT(
259 $ HMIN       EPS       X              Y1                  Y2              FUS    JACS  D[0,1] D[0,2]   ORDER     TIME    MAX.LOC.ERROR$);
260
261
262
263         FOR HMIN:= -4,-5,-6,-7 DO
264         FOR EPS := -4,-6,-8,-10 DO
265         BEGIN   IF EPS=-4 THEN NLCR; TIM:= TIME;
266                 FIRST:= TRUE; CJ:= CF:= 0;
267                 X:= 0; Y[0,1]:= Y[0,2]:= 1; YMAX[1]:= YMAX[2]:= 2;
268                 FOR XEND:= 0.005,50 DO
269         BEGIN   NLCR; IF XEND>1 THEN SPACE(20) ELSE FOR R:= HMIN,EPS DO
270                 BEGIN PLOT(1,1,R); SPACE(3) END;
271
272                 MUW@+8@5P(X,XEND,Y,HMIN,(XEND-X)/20,YMAX,
273                         EPS,FIRST,D,FXY,I,JAC,J,2,TRUE,TRUE);
274
275                 PLOT(1,1,X); FOR R:= Y[0,1],Y[0,2] DO BEGIN SPACE(3); FIXT(1,12,R) END; SPACE(3);
276                 FOR R:= CF/2,CJ/4,D[1,0],D[2,0],D[4,0] DO ABSFIXT(5,0,R); ABSFIXT(5,2,TIME-TIM);SPACE(3);
277                 IF D[2,0] # 0 THEN PLOT(5,2,D[3,0]);
278                 IF D[1,0] # 0 THEN GOTO AA;
279         END;
280         AA: NLCR
281
282         END
283 END
284
```

HMIN	EPS	X	Y1	Y2	FUB	JACB	D[0,1]	D[0,2]	ORDER	TIME	MAX,LOC,ERROR
+.1₁₀-3	+.1₁₀-3	+.5₁₀-2	+.999952046592	+i.000044239492	23	1	0	0	1	.75	
		+.5₁₀+2	+.597692985894	+1.402305120568	60	7	0	0	3	2.09	
+.1₁₀-3	+.1₁₀-5	+.5₁₀-2	+.999952046573	+i.000044239445	30	1	0	0	1	.86	
		+.5₁₀+2	+.597653770477	+1.402344336140	99	8	0	0	4	3.12	
+.1₁₀-3	+.1₁₀-7	+.5₁₀-2	+.999951584187	+i.000044701768	50	5	0	7	1	1.45	+.64822₁₀- 7
		+.5₁₀+2	+.597654039217	+1.402344067399	158	15	0	7	5	5.07	+.64822₁₀- 7
+.1₁₀-3	+.1₁₀-9	+.5₁₀-2	+.999952046538	+i.000044239408	79	1	0	22	1	1.80	+.64822₁₀- 7
		+.5₁₀+2	+.597654368058	+1.402343738555	364	38	0	22	4	11.53	+.64822₁₀- 7
+.1₁₀-4	+.1₁₀-3	+.5₁₀-2	+.999952037725	+i.000042591788	27	3	0	0	1	.88	
		+.5₁₀+2	+.597704340240	+1.402293766167	98	12	0	0	2	2.86	
+.1₁₀-4	+.1₁₀-5	+.5₁₀-2	+.999952510829	+1.000043775126	29	2	0	0	1	.87	
		+.5₁₀+2	+.597656376877	+1.402341729730	85	8	0	0	5	2.98	
+.1₁₀-4	+.1₁₀-7	+.5₁₀-2	+.999952510806	+i.000043775133	62	9	0	0	1	2.00	
		+.5₁₀+2	+.597654698834	+1.402343407779	160	17	0	0	5	5.33	
+.1₁₀-4	+.1₁₀-9	+.5₁₀-2	+.999952510802	+i.000043775137	110	8	0	2	6	4.18	+.84550₁₀- 9
		+.5₁₀+2	+.597654698547	+1.402343408066	443	46	0	2	5	15.73	+.84550₁₀- 9
+.1₁₀-5	+.1₁₀-3	+.5₁₀-2	+.999952500846	+i.000043750095	25	3	0	0	1	.86	
		+.5₁₀+2	+.597871631695	+1.402126474080	77	11	0	0	2	2.95	
+.1₁₀-5	+.1₁₀-5	+.5₁₀-2	+.999952510829	+i.000043775124	29	2	0	0	1	.87	
		+.5₁₀+2	+.597656417520	+1.402341689087	84	8	0	0	5	2.97	
+.1₁₀-5	+.1₁₀-7	+.5₁₀-2	+.999952510802	+1.000043775141	61	9	0	0	1	2.00	
		+.5₁₀+2	+.597654692590	+1.402343414025	176	18	0	0	5	5.74	
+.1₁₀-5	+.1₁₀-9	+.5₁₀-2	+.999952510793	+1.000043775141	112	8	0	0	4	4.19	
		+.5₁₀+2	+.597654699774	+1.402343406840	472	54	0	0	4	15.87	
+.1₁₀-6	+.1₁₀-3	+.5₁₀-2	+.999952500102	+1.000043748221	25	3	0	0	1	.85	
		+.5₁₀+2	+.597870070746	+1.402128035035	77	11	0	0	2	2.49	
+.1₁₀-6	+.1₁₀-5	+.5₁₀-2	+.999952510831	+1.000043775124	29	2	0	0	1	.87	
		+.5₁₀+2	+.597656366770	+1.402341739837	85	8	0	0	5	3.43	
+.1₁₀-6	+.1₁₀-7	+.5₁₀-2	+.999952510813	+1.000043775137	64	7	0	0	1	1.95	
		+.5₁₀+2	+.597654714180	+1.402343392434	173	17	0	0	5	5.58	
+.1₁₀-6	+.1₁₀-9	+.5₁₀-2	+.999952510795	+1.000043775148	116	8	0	0	3	4.17	
		+.5₁₀+2	+.597654699056	+1.402343407557	404	43	0	0	5	13.63	

1.9. Opgaven

1. Zij gegeven het beginwaardeprobleem

$$\dot{s} = -(1-c)s + qc,$$
$$\dot{c} = M((1-c)s - pc,$$
$$s(0) = 1, \quad c(0) = 0,$$
$$M = 1000; \quad q = 0.99; \quad p = 1.00.$$

Bereken met een standaard Runge-Kutta procedure s(t) en c(t) voor $t = 1,2,3,4,5$.
Bereken met behulp van de procedure MULTISTEP s(t) en c(t) voor $t = 1,2,3,4,5,10,15,20,25,50$.

Voer de berekening een aantal malen uit met verschillende nauwkeurigheid.
Hoe nauwkeurig is het verkregen resultaat?
Ga telkens na hoeveel functie-evaluaties en hoeveel evaluatie van de Jacobiaan nodig zijn.
Is hier sprake van een stijve differentiaalvergelijking?

2. Twee gekoppelde chemische reacties worden beschreven door de chemische vergelijkingen

$$ABB \underset{k_2}{\overset{k_1}{\rightleftharpoons}} AB + B,$$

$$AB \underset{k_4}{\overset{k_3}{\rightleftharpoons}} A + B,$$

met $k_1 = 0.795$, $k_2 = 0.845$, $k_3 = 0.893$, $k_4 = 0.940$.

Het verloop van de concentraties [ABB] = y en [AB] = z wordt beschreven door het stelsel differentiaalvergelijkingen

$$dy/dt = -k_1 y + k_2 z(b-z-2y),$$

$$dz/dt = -k_3 z + k_4(b-z-2y)(a-z-y) - dy/dt$$

Gegeven is:

$$a = 1; \quad b = 2; \quad y(0) = 0.25; \quad z(0) = 0.5.$$

Bereken $y(t)$ en $z(t)$ voor $t = 0.333, 0.672, 1.012, 100$
Gebruik de procedure MULTISTEP met

$$hmin = 0.002, 0.005, 0.01, 0.02$$
$$eps = 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$$
stiff:= false.

Wordt bij de integratie een Adams-Moulton of een Curtiss-Hirschfelder gebruikt?
Welke uitspraak kan men doen omtrent de nauwkeurigheid van de resultaten?

3. Zij gegeven het beginwaardeprobleem

$$\dot{x} = -0.04x + 10^4yz,$$
$$\dot{y} = 0.04x + 10^4yz - 3\,10^7y^2,$$
$$\dot{z} = 3\,10^7y^2,$$
$$x(0) = z(0) = 0; \quad y(0) = 1.$$

Laat zien dat dit probleem equivalent is met

$$\dot{u} = 0.3v^2,$$

$$\dot{v} = 400(1-u-0.0001v) - 10000v(u+0.3v),$$

$$u(0) = v(0) = 0.$$

Is hier sprake van een stijve differentiaalvergelijking?
Bereken $u(0.4)$, $v(0.4)$, $u(10)$, $v(10)$ en ga na hoeveel functie-evaluaties en hoeveel evaluaties van de Jacobiaan nodig zijn.

1.10. Literatuur

BASHFORTH, F. & ADAMS, J.C. *Theories of capillary action*, Cambridge Univ. Press. 1883

CURTISS, C.F. & HIRSCHFELDER J.O., *Integration of stiff equations*, Proc. Nat. Acad. Sci. U.S., 38 235. -1952

DEKKER, T.J., HEMKER, P.W. & HOUWEN, P.J. VAN DER, *Colloquium Stijve Differentiaalvergelijkingen*, MC Syllabus 15.1, Mathematisch Centrum, Amsterdam. 1972

GEAR, C.W., *The automatic integration of stiff ordinary differential equations*, Proc. IFIP Congr. 1968, pp. 187

GEAR, C.W., *Numerical initial value problems in ordinary differential equations*, Prentice-Hall Inc., Englewood Cliffs, N.J. 1971

HEMKER, P.W., *An ALGOL 60 procedure for the solution of stiff differential equations*, Report MR 128/71, Mathematisch Centrum, Amsterdam, 1971

HENRICI, P., *Discrete variable methods in ordinary differential equations*, John Wiley and Sons, New York. 1962

MOULTON, F.R., *New methods in exterior ballistics*, Univ. Chicago Press 1926

NORDSIECK, A., *On numerical integration of ordinary differential equations*, Math. Comp., 16 22-1962

ZONNEVELD, J.A., *Automatic numerical integration*, Math. Centre Tracts 8, Mathematisch Centrum, Amsterdam 1964

OPMERKING

Een onlangs (1973) verbeterde en gedocumenteerde versie van de procedure MULTISTEP is te vinden in:
NUMAL, a library of numerical procedures in ALGOL 60 (C. den Heijer, P.W. Hemker, P.J. van der Houwen, N. Temme, D.T. Winter eds.)
Mathematisch Centrum, Amsterdam.

MATHEMATICAL CENTRE SYLLABUS 20, 1975, II, 25-48.

EXPONENTIEEL AANGEPASTE RUNGE-KUTTA METHODEN VOOR STIJVE DIFFERENTIAALVERGELIJKINGEN

P.J. van der HOUWEN

Op het Mathematisch Centrum zijn de laatste jaren nieuwe Runge-Kutta formules gevonden waarvan de stabiliteitseigenschappen die van de standaard Runge-Kutta formules verre overtreffen. Deze formules zijn dan ook geschikt voor de integratie van stijve differentiaalvergelijkingen. Bovendien blijken ze heel goed bruikbaar voor de integratie van beginwaardeproblemen voor parabolische en hyperbolische differentiaalvergelijkingen. In dit hoofdstuk zullen we nader ingaan op de integratie van stijve differentiaalvergelijkingen. In hoofdstuk 3 komen partiële differentiaalvergelijkingen aan de orde.

2.1. Methode van Euler

De meest eenvoudige formule voor de integratie van het beginwaardeprobleem

$$(2.1) \qquad \vec{y}' = \vec{f}(x,\vec{y}), \quad \vec{y}(x_0) = \vec{y}_0$$

is de formule van Euler:

$$(2.2) \qquad \vec{y}_{n+1} = \vec{y}_n + h_n\vec{f}(x_n,\vec{y}_n), \quad n = 0,1,\ldots .$$

Deze formule is tevens de eenvoudigste vorm van een Runge-Kutta methode.

2.2. Algemene Runge-Kutta methoden

Een generalisatie van de Euler-formule werd door Runge [1895] gegeven. Voeren we de afkorting

$$j\vec{f}_{n+1} = \vec{f}({}_jx_{n+1}, {}_j\vec{y}_{n+1})$$

in, dan kan de algemene m-punts Runge-Kutta formule geschreven worden als

$$
(2.3)\qquad \begin{cases}
{}_jx_{n+1} = x_n + \mu_j\, h_n, \quad j = 0,1,\ldots,m-1, \\
{}_0\vec{y}_{n+1} = \vec{y}_n, \\
{}_1\vec{y}_{n+1} = \vec{y}_n + \lambda_{1,0}\, h_n\, {}_0\vec{f}_{n+1}, \\
{}_2\vec{y}_{n+1} = \vec{y}_n + \lambda_{2,0}\, h_n\, {}_0\vec{f}_{n+1} + \lambda_{2,1}\, h_n\, {}_1\vec{f}_{n+1} \\
\qquad \ldots \\
{}_m\vec{y}_{n+1} = \vec{y}_n + \lambda_{m,0}\, h_n\, {}_0\vec{f}_{n+1} + \lambda_{m,1}\, h_n\, {}_1\vec{f}_{n-1} + \ldots + \lambda_{m,m-1}\, {}_{m-1}\vec{f}_{n+1}, \\
\vec{y}_{n+1} = {}_m\vec{y}_{n+1}.
\end{cases}
$$

Hierin zijn μ_j en $\lambda_{j,l}$ nader te bepalen parameters.

Schema (2.3) berekent uit het resultaat $\vec{y}_n$ in het punt $x = x_n$ via m-1 tussenresultaten ${}_j\vec{y}_{n+1}$ in de tussenpunten $x = {}_jx_{n+1}$ een nieuwe vector $\vec{y}_{n+1}$ in het punt $x = x_{n+1}$ (zie figuur 2.1).

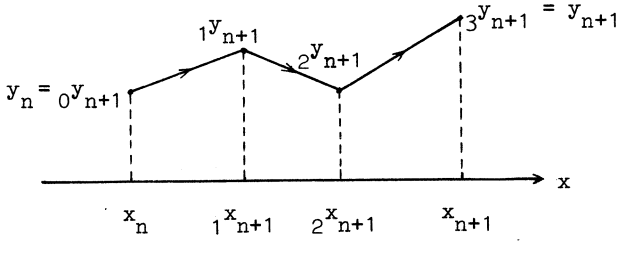


fig. 2.1. 3-punts Runge-Kutta-proces voor een scalar-vergelijking

Een Runge-Kutta formule wordt volledig vastgelegd door de (m×m) driehoeksmatrix $(\lambda_{j,l})$ en de vector (μ_j). In de praktijk kiest men altijd

$$
(2.4)\qquad \mu_0 = 0, \quad \mu_j = \lambda_{j,0} + \lambda_{j,1} + \ldots + \lambda_{j,j-1}, \quad j = 1,\ldots,m-1,
$$

zodat nu de matrix $(\lambda_{j,l})$ op een eenduidige wijze een Runge-Kutta proces genereert.

2.3. Standaard Runge-Kutta formule

De meest bekende Runge-Kutta formule werd door Kutta [1901] gegeven. In de notatie (2.3) luidt deze formule

$$\begin{cases} {}_0x_{n+1} = x_n, \quad {}_1x_{n+1} = {}_2x_{n+1} = x_n + \frac{1}{2}h_n, \quad {}_3x_{n+1} = x_n + h_n, \\ {}_0\vec{y}_{n+1} = \vec{y}_n, \\ {}_1\vec{y}_{n+1} = y_n + \frac{1}{2}h_n\, {}_0f_{n+1}, \\ {}_2\vec{y}_{n+1} = \vec{y}_n + \frac{1}{2}h_n\, {}_1\vec{f}_{n+1}, \\ {}_3\vec{y}_{n+1} = \vec{y}_n + h_n\, {}_2\vec{f}_{n+1} \\ \vec{y}_{n+1} = {}_4\vec{y}_{n+1} = \vec{y}_n + h_n\left[\frac{1}{6}\, {}_0\vec{f}_{n+1} + \frac{1}{3}\, {}_1\vec{f}_{n+1} + \frac{1}{3}\, {}_2\vec{f}_{n+1} + \frac{1}{6}\, {}_3\vec{f}_{n+1}\right], \end{cases}$$

of in compacte vorm

$$(\lambda_{j,l}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{bmatrix}$$

2.4. Orde van nauwkeurigheid

Stel dat het integratieproces gevorderd is tot het punt $(x_n,\vec{y}_n)$ in het $(x,\vec{y})$-vlak. Door dit punt gaat een oplossing $\vec{z}(x_n,\vec{y}_n;x)$ van de differentiaalvergelijking. Dus zou men exact integreren dan zou men in het punt $x = x_{n+1} = x_n + h_n$ de vector $\vec{z}(x_n,\vec{y}_n;x_{n+1})$ vinden. De numerieke integratieformule levert echter de benadering $\vec{y}_{n+1}$. Het verschil

$$\vec{\rho}_n = \vec{z}(x_n,\vec{y}_n;x_{n+1}) - \vec{y}_{n+1} \tag{2.5}$$

wordt de *locale discretiseringsfout genoemd* (ook wel *afbreekfout* in het punt $(x_n, \vec{y}_n)$). Wanneer

$$\vec{\rho}_n = O(h_n^{p+1}) \quad \text{voor } h_n \to 0$$

dan heet de integratieformule *nauwkeurig van de orde p* (vergelijk hoofdstuk 1, blz. 7).

Stelling 2.1

De Runge-Kutta formules gegenereerd door de diagonaalmatrix

$$(2.6) \qquad (\lambda_{j,l}) = \begin{bmatrix} \lambda_{1,0} & 0 & \cdots & 0 \\ 0 & \lambda_{2,1} & \ddots & \\ \vdots & & \ddots & 0 \\ 0 & & \lambda_{m-1,m-2} & 1 \end{bmatrix},$$

zijn nauwkeurig van de orde p = 1.

Als geldt

$$(2.7) \qquad \lambda_{m-1,m-2} = \frac{1}{2}$$

dan is de orde van nauwkeurigheid p = 2.

Bewijs

Zie MC Syllabus 15.1 [1972]. □

Stelling 2.2

De Runge-Kutta formules gegenereerd door de matrix

$$(2.8)\qquad (\lambda_{j,l}) = \begin{bmatrix} \lambda_{1,0} & 0 & \cdots & & & 0 \\ 1/4 & \lambda_{2,1} & & & & \\ 1/4 & 0 & \lambda_{3,2} & & & \\ \vdots & & \ddots & \ddots & & \vdots \\ 1/4 & 0 & \cdots & 0 & \lambda_{m-3,m-4} & \\ 1/4 & 0 & \cdots & & 0 \quad 17/60 & \\ 1/4 & 0 & \cdots & & 0 \quad 5/12 & 0 \\ 1/4 & 0 & \cdots & & 0 & 3/4 \end{bmatrix}$$

zijn naurkeurig van de orde p = 3

Bewijs

Zie MC Syllabus 15.1 [1972]. □

De door deze 2 stellingen gedefinieerde klassen van Runge-Kutta formules zijn onderwerp van uitvoerige studie geweest op het MC. In het bijzonder is nagegaan hoe de nog vrije parameters $\lambda_{j,l}$ gekozen kunnen worden om efficiënte en betrouwbare integratieformules te krijgen. Hierbij werd in het bijzonder gezocht naar formules met gunstige stabiliteits-eigenschappen. Hiernaast zijn ook gestabiliseerde vierde en vijfde orde Runge-Kutta formules bestudeerd. De geïnteresseerde lezer verwijzen we hiervoor naar de referenties: Beentjes & Dekker [1972] en van der Houwen [1971a, 1971b, 1972].

2.5. Toepassing op lineaire differentiaalvergelijkingen

Om het karakter van een Runge-Kutta formule te leren kennen passen we deze toe op de lineaire vector-differentiaalvergelijking

$$(2.9)\qquad \vec{y}' = J\vec{y},$$

waarin J een matrix voorstelt met niet van x of $\vec{y}$ afhangende matrixelementen. Bijvoorbeeld de formule gedefinieerd door (2.6) geeft

$$(2.10)\qquad \vec{y}_{n+1} = I + h_n J(I + \lambda_{m-1,m-2} h_n J(I + \lambda_{m-2,m-3} J(I + \ldots$$

$$\ldots + \lambda_{2,1} h_n J(I + \lambda_{1,0} h_n J))\ldots)\vec{y}_n.$$

Hierin stelt I de eenheidsmatrix voor. Introduceren we de nieuwe parameters β_j, $j = 2,3,\ldots,m$ gedefinieerd door

$$(2.11)\qquad \begin{cases} \beta_2 = \lambda_{m-1,m-2}, \\ \beta_j = \lambda_{m+1-j,m-j}\ \beta_{j-1}, & j = 3,\ldots,m, \end{cases}$$

dan kan (2.10) geschreven worden als

$$(2.12)\qquad y_{n+1} = [I + h_n J + \beta_2(h_n J)^2 + \ldots + \beta_m(h_n J)^m]\ \vec{y}_n =$$

$$= P_m(h_n J)\ \vec{y}_n.$$

Dus de Runge-Kutta formule (2.6) toegepast op de lineaire vergelijking (2.9) vereenvoudigt zich tot het toepassen van een polynoom operator $P_m(h_n J)$ van de graad m in $h_n J$ op de vector $\vec{y}_n$. Het is eenvoudig in te zien dat dit voor elke Runge-Kutta formule (2.3) geldt, uiteraard met andere polynomen $P_m(z)$, d.w.z. andere coëfficiënten β_j. Zo leidt de derde orde nauwkeurige Runge-Kutta formule (2.8) na toepassing op vergelijking (2.9) tot formule (2.12) waarin de coëfficiënten β_j nu echter gegeven worden door

$$(2.13)\qquad \begin{cases} \beta_2 = \frac{1}{2}, \quad \beta_3 = \frac{1}{6}, \\ \beta_j = \dfrac{1 + 4\lambda_{m-j+1,m-j}}{1 + 4\lambda_{m-j+2,m-j+1}}\ \lambda_{m-j+2,m-j+1}\cdot\beta_{j-1}, & j = 4,5,\ldots,m-1 \\ \beta_m = \dfrac{4\lambda_{2,1}}{1 + 4\lambda_{2,1}}\ \beta_{m-1}. \end{cases}$$

We merken nog op dat voor een Runge-Kutta formule die nauwkeurig van de orde p is de eerste p coëfficiënten altijd gegeven worden door

$$(2.14)\qquad \beta_j = \frac{1}{j!}, \qquad j = 1,2,\ldots,p,$$

dus

$$(2.15)\qquad P_m(z) = 1 + z + \ldots + \frac{1}{p!}\ z^p + \beta_{p+1}\ z^{p+1} + \ldots + \beta_m\ z^m.$$

2.6. Stabiliteit

Stel dat we in plaats van $\vec{y}_n$ een verstoorde waarde $\vec{y}_n^*$ verkregen hebben. Dan zullen we ook in plaats van $\vec{y}_{n+1}$ een verstoorde waarde $\vec{y}_{n+1}^*$ vinden. Een goede integratieformule zal de verstoring $\vec{y}_n^* - \vec{y}_n$ niet groter laten worden. Dus men zou willen eisen dat de integratieformule zodanig is dat

$$(2.16) \qquad \| \vec{y}_{n+1}^* - \vec{y}_{n+1} \| < \| \vec{y}_n^* - \vec{y}_n \| .$$

Hierin stelt $\| \ \|$ de een of andere vectornorm voor; bijvoorbeeld de maximum norm of de Euclidische norm. We zullen laten zien dat de nog vrije parameters $\lambda_{j,l}$ in de formules (2.6) en (2.8) zo gekozen kunnen worden dat aan deze stabiliteitsvoorwaarde voor een grote klasse van differentiaalvergelijkingen voldaan kan worden.

Beschouw de vergelijking voor het tussenpunt ${}_j\vec{y}_{n+1}$ uit schema (2.3):

$${}_j\vec{y}_{n+1} = \vec{y}_n + \lambda_{j,0}\, h_n\, {}_0\vec{f}_{n+1} + \ldots + \lambda_{j,j-1}\, h_n\, {}_{j-1}\vec{f}_{n+1} \ ;$$

beschouw ook de vergelijking voor het tussenpunt ${}_j\vec{y}_{n+1}^*$ behorende bij de verstoorde beginwaarde $\vec{y}_n^*$:

$${}_j\vec{y}_{n+1}^* = \vec{y}_n^* + \lambda_{j,0}\, h_n\, {}_0\vec{f}_{n+1}^* + \ldots + \lambda_{j,j-1}\, h_n\, {}_{j-1}\vec{f}_{n+1}^* .$$

Uit deze relaties volgt met behulp van de middelwaardestelling

$${}_j\vec{y}_{n+1}^* - {}_j\vec{y}_{n+1} \cong \vec{y}_n^* - y_n + \lambda_{j,0}\, h_n\, J_n({}_0\vec{y}_{n+1}^* - {}_0\vec{y}_{n+1}) + \ldots$$

$$\ldots + \lambda_{j,j-1}\, h_n\, J_n({}_{j-1}\vec{y}_{n+1}^* - {}_{j-1}\vec{y}_{n+1}) .$$

Door recursie vinden we voor het verschil $\vec{y}_{n+1}^* - \vec{y}_{n+1}$ de formule (vergelijk (2.10))

$$(2.17) \qquad \vec{y}_{n+1}^* - \vec{y}_{n+1} \cong P_m(h_n J_n)\,(\vec{y}_n^* - \vec{y}_n),$$

waarin $P_m(z)$ juist het door (2.12) gedefinieerde polynoom is. Dus de stabiliteitsvoorwaarde (2.16) wordt voor Runge-Kutta formules

(2.18) $\quad \| P_m(h_n J_n) \| < 1.$

Vergelijking (2.17) is als het ware de *variatievergelijking* van de Runge-Kutta methode. Het polynoom $P_m(z)$ wordt het *stabiliteitspolynoom* genoemd. De operator $P_m(h_n J_n)$ zullen we de *amplificatieoperator* in het punt $(x_n, \vec{y}_n)$ noemen. Het gebied S in het complexe Z-vlak waar $P_m(z)$ in modulus kleiner dan 1 is wordt het *stabiliteitsgebied* genoemd, dus

(2.19) $\quad S = \{z \mid |P_m(z)| < 1\}.$

2.7. Interne stabiliteit

Wanneer tijdens de berekeningen, nodig om de stap $\vec{y}_n \to \vec{y}_{n+1}$ uit te voeren, geen afrondfouten gemaakt worden, dan beschrijft de amplificatie-operator $P_m(h_n J_n)$ de mate waarin een verstoring van $\vec{y}_n$ in $\vec{y}_{n+1}$ doorwerkt. We zullen nu nagaan hoe afrondfouten gemaakt bij de berekening van de tussenpunten ${}_j\vec{y}_{n+1}$, $j = 1,\ldots,m$, doorwerken in $\vec{y}_{n+1}$. Hierbij beperken we ons tot Runge-Kutta processen waarin de matrix $(\lambda_{j,l})$ uitsluitend nullen bevat behalve op de hoofddiagonaal en in de eerste kolom (merk op dat de door stelling 2.1 en 2.2 gedefinieerde processen een dergelijke parameter-matrix hebben). We hebben dus te maken met een rekenschema van de vorm:

$$
(2.20)\quad \begin{cases} {}_0\vec{y}_{n+1} = \vec{y}_n, \\ {}_1\vec{y}_{n+1} = \vec{y}_n + \lambda_{1,0}\, h_n\, {}_0\vec{f}_{n+1}, \\ {}_j\vec{y}_{n+1} = \vec{y}_n + \lambda_{j,1}\, h_n\, {}_0\vec{f}_{n+1} + \lambda_{j,j-1}\, h_n\, {}_{j-1}\vec{f}_{n+1}, \quad j = 2,3,\ldots,m, \\ \vec{y}_{n+1} = {}_m\vec{y}_{n+1}. \end{cases}
$$

Stel dat de berekening van de vector ${}_j\vec{y}_{n+1}$ aanleiding geeft tot een fout ${}_j\vec{\rho}_n^{\,*}$ en laten we de door dergelijke fouten verstoorde vectoren ${}_j\vec{y}_{n+1}$ aangeven met ${}_j\vec{y}^{\,*}_{n+1}$. Dan krijgen we in plaats van (2.20) het rekenschema

$$(2.20^*)\quad \begin{cases} {}_1\vec{y}^*_{n+1} = \vec{y}_n + \lambda_{1,0}\, h_n\, {}_0\vec{f}^*_{n+1} + {}_1\vec{\rho}^*_n, \\ {}_j\vec{y}^*_{n+1} = \vec{y}_n + \lambda_{j,0}\, h_n\, {}_0\vec{f}^*_{n+1} + \lambda_{j,j-1}\, h_n\, {}_{j-1}\vec{f}^*_{n+1} + {}_j\vec{\rho}^*_n, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad j = 2,3,\ldots,m, \\ \vec{y}_{n+1} = {}_m\vec{y}^*_{n+1}. \end{cases}$$

Met behulp van de middelwaardestelling volgt uit (2.20) en (2.20^*)

$$\begin{cases} {}_1\vec{y}^*_{n+1} - {}_1\vec{y}_{n+1} = {}_1\vec{\rho}^*_n, \\ {}_j\vec{y}^*_{n+1} - {}_j\vec{y}_{n+1} = \lambda_{j,j-1}\, h_n\, J_n({}_{j-1}\vec{y}^*_{n+1} - {}_{j-1}\vec{y}_{n+1}) + {}_j\vec{\rho}^*_n, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad j = 2,3,\ldots,m, \\ \vec{y}^*_{n+1} - \vec{y}_{n+1} = {}_m\vec{y}^*_{n+1} - {}_m\vec{y}_{n+1}. \end{cases}$$

Door recursie volgt hieruit

$$\begin{aligned} \vec{y}^*_{n+1} - \vec{y}_{n+1} = {} & \prod_{j=2}^{m} \lambda_{j,j-1} [h_n\, J_n]^{m-1}\, {}_1\vec{\rho}^*_n + \\ & + \prod_{j=3}^{m} \lambda_{j,j-1} [h_n\, J_n]^{m-2}\, {}_2\vec{\rho}^*_n + \\ & \qquad \ldots \\ & + \lambda_{m,m-1}\, h_n\, J_n\, {}_{m-1}\vec{\rho}^*_n + {}_m\vec{\rho}^*_n. \end{aligned}$$

We definiëren nu de functies

$$(2.21)\qquad Q_\mu(z) = \prod_{j=m+1-\mu}^{m} \lambda_{j,j-1}\, z^\mu, \qquad \mu = 1,2,\ldots,m-1.$$

De waarden van $Q_\mu(h_n\delta)$, δ eigenwaarde van J_n, bepalen de mate waarin afrondfouten zich ophopen. De factoren $Q_\mu(h_n\,|\delta|_{max})$ zullen we *interne amplificatiefactoren* noemen. Om de *interne stabiliteit* te verzekeren zal men voorwaarden van de vorm

$$(2.22)\qquad \underset{1\le\mu\le m-1}{\text{maximum}} \left|Q_\mu(h_n\,|\delta_{max}|)\right| < \eta$$

moeten stellen; hierin stelt η een tolerantie voor die des te kleiner gekozen zal moeten worden naarmate de machine-precisie geringer is (vergelijk Dekker [1972]).

2.8. Exponentieële aanpassing

We hebben de variatievergelijking voor de Runge-Kutta formules afgeleid; laten we deze vergelijking eens vergelijken met de variatievergelijking van de differentiaalvergelijking. In hoofdstuk 1 werd hiervoor aangegeven

$$\vec{v}' = J(x,\vec{y}(x))\,\vec{v}, \tag{2.23}$$

waarin v(x) het verschil van twee oplossingen $\vec{y}(x)$ en $\vec{y}^*(x)$ is. Uit (2.23) volgt direkt

$$\vec{y}^*(x_{n+1}) - \vec{y}(x_{n+1}) = \exp\Big(\int_{x_n}^{x_n+h_n} J(x;\vec{y}(x))dx\Big)\,(\vec{y}^*(x_n)-\vec{y}(x_n)). \tag{2.24}$$

Dus zoals de amplificatieoperator $P_m(h_n J_n)$ in benadering de numeriek verstoringen in het numerieke integratieproces beschrijft, zo beschrijft de amplificatieoperator $\exp(\int_{x_n}^{x_n+h_n} J\,dx)$ verstoringen in het analytische integratieproces.

Het principe van exponentiële aanpassing is nu dat voor zekere verstoringen $\vec{e}$ de werking van de numerieke en analytische amplificatieoperatoren in de punten $(x_n,\vec{y}_n)$, n = 0,1,..., identiek is, dus

$$P_m(h_n J_n)\,\vec{e} = \exp\int_{x_n}^{x_n+h_n} J(x,\vec{z}(x_n,\vec{y}_n,x))\,dx\,\vec{e}. \tag{2.25}$$

Aangezien de locaal analytische oplossing $\vec{z}$ niet bekend is vervangt men deze voorwaarde in de praktijk door

$$P_m(h_n J_n)\,\vec{e} = \exp(h_n J_n)\,\vec{e}. \tag{2.25'}$$

Dit is realistisch in de gevallen waar de Jacobiaan langzaam verandert. Wanneer $\vec{e}$ een eigenvector van J_n is met eigenwaarde δ dan reduceert (2.25') tot

$$P_m(h_n\delta) = \exp(h_n\delta). \tag{2.25"}$$

Men zegt dat het polynoom $P_m(z)$ *enkelvoudig exponentieël aangepast* is in

het punt $z = h_n\delta$. Men spreekt van *l-voudige aanpassing* wanneer

$$(2.26) \qquad P_m^{(j)}(z) = \exp(z), \qquad j = 0,1,\ldots,l-1.$$

Voorwaarden van bovenstaand type vindt men in een aantal recente publicaties. We noemen Pope [1963], Liniger en Willoughby [1970] en MC Syllabus 15 [1972].

We merken op dat het stabiliteitspolynoom van een p-de orde nauwkeurig Rung-Kutta proces (p+1)-voudig in z = 0 aangepast is (vergelijk (2.15)).

2.9. Stijve differentiaalvergelijkingen

Een voorbeeld van een stijve differentiaalvergelijking is een vergelijking waarvan de Jacobiaan een eigenwaardenspectrum heeft bestaande uit 3 clusters, één dicht bij de oorsprong, de andere twee clusters toegevoegd complex met groot negatief reëel deel (zie figuur 2.2).

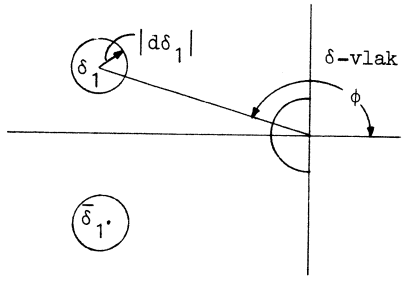


fig. 2.2. Eigenwaarden-spectrum van een stijve differentiaalvergelijking

Om een stabiele integratieformule voor een dergelijke vergelijking te contrueren zal het stabiliteitspolynoom zodanig moeten zijn dat het stabiliteitsgebied S een gebied bij de oorsprong bevat en gebieden in de omgeving van de punten $z_1 = h_n\,\delta_1$ en $\bar{z}_1 = h_n\,\bar{\delta}_1$, waarin δ_1 en $\bar{\delta}_1$ de "middelpunten" van de twee complexe clusters van eigenwaarden zijn. Stel dat het polynoom $R_r(z)$ het gewenste stabiliteitsgebied bij de oorsprong heeft en bovendien consistent met orde van nauwkeurigheid p is, dus

$$(2.27) \qquad R_r(z) = 1 + z + \ldots + \frac{1}{p!}\,z^p + \beta_{p+1}\,z^{p+1} + \ldots + \beta_r\,z^r.$$

We schrijven het stabiliteitspolynoom $P_m(z)$ als

$$(2.28) \qquad P_m(z) = R_r(z) + \beta_{r+1} z^r + \ldots + \beta_m z^m,$$

waarin de $l = m-r$ nog vrije coëfficiënten β_j gebruikt kunnen worden om $P_m(z)$ exponentieel aan te passen in z_1 en $\bar{z}_1$. Wanneer $z_1 = \bar{z}_1$ dan kan een l-voudige aanpassing in z_1 verkregen worden. Wanneer $z_1 \neq \bar{z}_1$ dan kunnen beide punten $l/2$-voudig aangepast worden (in dit geval dient $l = m-r$ even te zijn). Een expliciete constructie van de aldus gedefinieerde stabiliteitspolynomen vindt men in Dekker [1972]. Men kan nu de volgende stelling bewijzen voor het exponentieel aangepaste polynoom (2.28):

Stelling 2.3

Zij $z_1 = |z_1| \exp(i\phi)$. Het stabiliteitsgebied S van (2.28) bevat het stabiliteitsgebied van $R_r(z)$ en de cirkelvormige gebieden

$$(2.29) \quad \begin{cases} |z - z_1| \leq \dfrac{|z_1|^{\frac{l-r}{l}}}{\beta_r^{\frac{1}{l}}} & \text{als } \phi = \pi, \\ |z - z_1| \leq \dfrac{|z_1|^{\frac{l-2r}{l}}}{2\beta_r^{\frac{2}{l}} |\sin \phi|} & \\ |z - \bar{z}_1| \leq \dfrac{|\bar{z}_1|^{\frac{l-2r}{l}}}{2\beta_r^{\frac{2}{l}} |\sin \phi|} & \text{als } \phi = \pi. \end{cases}$$

Bewijs

Zie van der Houwen [1971b]. □

Uit deze stelling volgt dat het stabiliteitsgebied van grootte verandert wanneer de integratiestap h_n verandert (bij gegeven eigenwaarde δ_1). Aangezien men in de praktijk de eigenwaarde δ_1 dikwijls niet exact kent moeten de methoden met $l < r$ voor $\phi = \pi$ en $l < 2r$ voor $\phi \neq \pi$ met de nodige zorg toegepast worden.

2.10. De procedure "exponential fitted runge-kutta"

We zijn nu zover dat een Runge-Kutta formule van de orde p = 1, 2 of 3 geconstrueerd kan worden waarvan het stabiliteitspolynoom exponentieel aangepast is in punten $z_1 = h_n \delta_1$ en $\bar{z}_1 = h_n \bar{\delta}_1$. Een ALGOL 60-versie van deze integratieformule vindt men in Dekker [1972]. Deze algoritme, de procedure "exponential fitted runge kutta", integreert beginwaardeproblemen van het type (2.1). Van de door de gebruiker op te geven integratiestappen wordt gecontroleerd of ze voldoen aan de voorwaarden voor stabiliteit binnen één stap (voorwaarde (2.22)) en de stabiliteitsvoorwaarden voorgeschreven door (2.29). Zonodig wordt de integratiestap hieraan aangepast. De procedrue "exponential fitted runge kutta" is gedocumenteerd in de LR-uitgave van het Mathematisch Centrum (LR 3.3.7). Volledigheidshalve volgt hieronder de gebruiksaanwijzing met referenties naar de notatie en en theorie zoals die in dit hoofdstuk gegeven is:

Declaratie

```
procedure exponential fitted runge kutta (t,te,m0,m,u,sigma,
          phi,diameter,derivative,k,step,r,l,beta,third order,
          tol,output);
integer   m0,m,k,r,l;
real      t,te,sigma,phi,diameter,step,tol;
array     u,beta;
boolean   third order;
procedure derivative,output;
```

Parameters

t : <variable>;
de onafhankelijke variabele x;
bij aanroep moet t de waarde van x_0 hebben;
t kan als Jensen-parameter gebruikt worden;

te : <expression>;
eindwaarde van x;

m0,m : <expression>;
indices van eerste en laatste component van de vector-differentiaalvergelijking;

u : <array identifier>; array u [m0:m];
het array u stelt de vector $\vec{y}_n$ voor;
bij aanroep moet dit array de componenten van $\vec{y}_0$ bevatten;

sigma : <expression>;
de waarde van $|\delta_1|$ (zie figuur 2.2);

phi : <expression>;
de waarde van ϕ in radialen (zie figuur 2.2);

diamater : <expression>;
diameter $2|d\delta_1|$ van de linker clusters (zie figuur 2.2);

derivative: <procedure identifier>;
een procedure die als volgt door de gebruiker gegeven moet worden:

```
procedure derivative(x,y); real x array y;
          <body>;
```

na uitvoering moet het array y de componenten van $\vec{f}(x,\vec{y})$ bevatten;

k : <variable>;
k stelt de index n voor;
k telt dus het aantal integratiestappen;
bij aanroep moet k een startwaarde hebben (b.v. k = 0);
k kan als Jensen-parameter gebruikt worden;

step : <expression>;
stelt de integratiestap h_n voor en moet door de gebruiker opgegeven worden;
wanneer de stabiliteit van het integratieproces in gevaar komt verkleint de procedure de waarde van step automatisch;

r : <expression>;
de graad van het polynoom $R_r(z)$ uit (2.28);

l : <expression>;
orde van de exponentiële aanpassing;
l = m-r, waarin m het aantal te gebruiken punten van de Runge-Kutta formule is;

beta : <array identifier>; array beta [0:l+r];
bij aanroep moet beta[j] = β_j voor j = 0,...,r;
dus het polynoom $R_r(z)$ moet van te voren door de gebruiker opgesteld worden;

third order: <boolean expression>;
indien third order = true dan rekent de procedure 3^e orde nauwkeurig (mits het polynoom $R_r(z)$ geschikt gekozen is), anders hoogstens 2^e orde nauwkeurig;

tol : <expression>;
een maat voor de nauwkeurigheid waarmee binnen één stap gerekend wordt en heeft te maken met de tolerantie n in voorwaarde (2.22);

output : <procedure identifier>;
een procedure waarmee de gebruiker de waarden van

t, u[m0:m], sigma, phi, diameter, k en step

kan opvragen.

2.11. Voorbeeld van aanroep

In de biochemie is het volgende beginwaardeprobleem van belang:

$$(2.30)\quad \begin{cases} y_1' = -y_1 + .99\, y_2 + y_1\, y_2, \\ y_2' = 1000(y_1 - y_2 - y_1 y_2), \\ y_1(0) = 1, \quad y_2(0) = 0. \end{cases}$$

Gevraagd wordt de oplossing in het punt x = 50

Met een 5^e orde Runge-Kutta proces werden in $16_{10}4$ stappen de volgende waarden gevonden:

$$(2.31)\quad \begin{cases} y_1(50) = .7658783202..., \\ y_2(50) = .4337103535..., \end{cases}$$

We zullen de oplossing nu trachten te vinden met de procedure exponential fitted rung kutta. Daartoe beschouwen we eerst de Jacobiaan van het systeem; deze wordt gegeven door

$$J = \begin{bmatrix} y_2 - 1 & y_1 + .99 \\ 1000(1-y_2) & -1000(1+y_1) \end{bmatrix}.$$

De eigenwaarden worden gegeven door

$$\delta = \frac{1}{2}(S \pm \sqrt{S^2 - 4P}),$$

$$S = -1001 - 1000y_1 + y_2,$$

$$P = 10(1-y_2).$$

Een redelijke benadering is

$$(2.32) \qquad \delta_0 \tilde{=} -\frac{1}{100}\frac{1-y_2}{1+y_1}, \quad \delta_1 \tilde{=} -1000(y_1+1).$$

Op grond hiervan moet men verwachten dat het systeem een zeer stijf gedrag zal vertonen, zodat een exponentieel aangepaste Runge-Kutta methode een effeciënte integratietechniek lijkt. Hierbij kan men dan in de eigenwaarde δ_1 exponentieel aanpassen. Geeft men de exacte waarde van $|\delta_1|$ mee aan de procedure (sigma = $|\delta_1|$) dan kan voor de parameter diameter nul gekozen worden, waarmee de stabiliteitsconditie geen beperking aan de integratiestap oplegt. Om echter ook het effect van de stabiliteit te illustreren zullen we voor δ_1 de benaderde waarde (2.32) nemen. Voor diameter nemen we de waarde $2(.99+y_1)$; volgens een stelling van Gerschgorin ligt de exacte waarde van δ_1 namelijk binnen de cirkel bepaald door (2.32) en deze waarde van diameter. Verder moet men nog het polynoom $R_r(z)$ zo kiezen dat δ_0 binnen het bijbehorende stabiliteitsgebied ligt. Aangezien echter δ_0 erg klein is zullen we hiervoor geen bijzonder polynoom hoeven te kiezen, dat wil zeggen we kunnen r = p kiezen, ofwel

$$R_r(z) = 1 + z + \frac{1}{2!}z^2 + \ldots + \frac{1}{p!}z^p.$$

Hieronder volgt de volledige tekst van het gebruikte ALGOL 60-programma en de hiermee verkregen resultaten:

```
1   BEGIN
2
3   PROCEDURE EXPONENTIAL FITTED RUNGE KUTTA(T, TE, M0, M, U, SIGMA, PHI,
4                                           DIAMETER, DERIVATIVE, K,
5                                           STEP, R, L, BETA, THIRDORDER,
6                                           TOL, OUTPUT);
7   INTEGER M0, M, K, R, L;
8   REAL T, TE, SIGMA, PHI, DIAMETER, STEP, TOL;
9   ARRAY U, BETA;
10  BOOLEAN THIRDORDER;
11  PROCEDURE DERIVATIVE, OUTPUT;
12
13  BEGIN INTEGER N;
14      REAL THETA0,THETANM1,TAU,B,B0,PHI0,PHIL,BETAR;
15      BOOLEAN FIRST,COMPLEX,CHANGE;
16      INTEGER ARRAY P[1:L];
17      REAL ARRAY MU,LABDA[0:R+L-1],PT[0:R],FAC,BETAC[0:L-1],RL[M0:M],
18                A[1:L,1:L];
19
20      PROCEDURE FORMCONSTANTS;
21      BEGIN INTEGER II;
22          FIRST:=FALSE;
23          FAC[0]:=1;
24          FOR II:=1 STEP 1 UNTIL L-1 DO FAC[II]:=II*FAC[II-1];
25          PT[R]:=L*FAC[L-1];
26          FOR II:=1 STEP 1 UNTIL R DO PT[R-II]:=PT[R-II+1]*(L+II)/II
27      END FORMCONSTANTS;
28
29      PROCEDURE FORMBETA;
30      BEGIN INTEGER II,JJ; REAL BB,C,D;
31          IF FIRST THEN FORMCONSTANTS;
32          D:=C:=EXP(-B);
33          BETAC[L-1]:=D/FAC[L-1];
34          FOR II:=1 STEP 1 UNTIL L-1 DO
35          BEGIN C:=B*C/II; D:=D+C; BETAC[L-1-II]:=D/FAC[L-1-II] END;
36          BB:=1;
37          FOR II:=R+1 STEP 1 UNTIL R+L DO
38          BEGIN C:=0;
39              FOR JJ:=0 STEP 1 UNTIL R DO
40              C:=(BETA[JJ]-(IF JJ<L THEN BETAC[JJ] ELSE 0))*PT[JJ]/(II-JJ)*C/B;
41              BETA[II]:=C/B/FAC[L+R-II]/FAC[II-R-1]*
42                        (IF II<L THEN BB*BETAC[II] ELSE 0);
43              BB:=BB*B
44          END
45      END FORMBETA;
46
47      PROCEDURE SOLUTIONOFCOMPLEXEQUATIONS;
48      BEGIN INTEGER II,JJ,C1,C3;
49          REAL C2,E,B1,Z11,COSPHI,SINPHI,COS11PHI,SIN11PHI,COSPHIL;
50          REAL ARRAY D[1:L];
51
52          PROCEDURE ELEMENTS OF MATRIX;
53          BEGIN PHIL:=PHI0;
54              COSPHI:=COS(PHIL); SINPHI:=SIN(PHIL);
55              COS11PHI:=1; SIN11PHI:=0;
56              FOR II:=0 STEP 1 UNTIL L-1 DO
```

```
57              BEGIN C1:=R+1+II; C2:=1;
58                  FOR JJ:=L-1 STEP -2 UNTIL 1 DO
59                  BEGIN A[JJ,L-II]:=C2*COS(IPHI;
60                      A[JJ+1,L-II]:=C2*SIN(IPHI;
61                      C2:=C1*C2; C1:=C1-1
62                  END;
63                  COSPHIL:=COS(IPHI*COSPHI-SIN(IPHI*SINPHI;
64                  SIN(IPHI:=COS(IPHI*SINPHI+SIN(IPHI*COSPHI;
65                  COS(IPHI:=COSPHIL
66              END;
67              DET(A,L,P)
68          END EL OF MAT;
69
70          PROCEDURE RIGHT HAND SIDE;
71          BEGIN E:=EXP(B*COSPHI);
72              B1:=B*SINPHI-(R+1)*PHIL;
73              COS(IPHI:=E*COS(B1); SIN(IPHI:=E*SIN(B1);
74              B1:=1/B; Z(I:=B1+R;
75              FOR JJ:=L STEP -2 UNTIL 2 DO
76              BEGIN D[JJ]:=Z(I*SIN(IPHI;
77                  D[JJ-1]:=Z(I*COS(IPHI;
78                  COSPHIL :=COS(IPHI*COSPHI-SIN(IPHI*SINPHI;
79                  SIN(IPHI:=COS(IPHI*SINPHI+SIN(IPHI*COSPHI;
80                  COS(IPHI:=COSPHIL;
81                  Z(I:=Z(I*B
82              END;
83              COS(IPHI:=Z(I:=1; SIN(IPHI:=0;
84              FOR II:=R STEP -1 UNTIL 0 DO
85              BEGIN C1:=II; C2:=BETA[II];
86                  C3:=IF 2*II>L-2 THEN 2 ELSE L-2*II;
87                  COSPHIL :=COS(IPHI*COSPHI-SIN(IPHI*SINPHI;
88                  SIN(IPHI:=COS(IPHI*SINPHI+SIN(IPHI*COSPHI;
89                  COS(IPHI:=COSPHIL;
90                  FOR JJ:=L STEP -2 UNTIL C3 DO
91                  BEGIN D[JJ]:=D[JJ]+Z(I*C2*SIN(IPHI;
92                      D[JJ-1]:=D[JJ-1]-Z(I*C2*COS(IPHI;
93                      C2:=C2*C1; C1:=C1-1
94                  END;
95                  Z(I:=Z(I*B1
96              END
97          END RIGHT HAND SIDE;
98
99          IF PHI0≠PHIL THEN ELEMENTS OF MATRIX;
100         RIGHT HAND SIDE;
101         SOL(A,L,P,D);
102         FOR II:=1 STEP 1 UNTIL L DO BETA[R+II]:=D[L+1-II]*B1
103     END SOLOFCOMEQ;
104
105     PROCEDURE COEFFICIENT;
106     BEGIN INTEGER J,K; REAL C;
107         B0:=B; PHI0:=PHI;
108         IF B≥1 THEN
109         BEGIN IF COMPLEX THEN SOLUTION OF COMPLEX EQUATIONS
110             ELSE FORMBETA
111         END;
112         LABDA[0]:=MU[0]:=0;
113         IF THIRDORDER THEN
114         BEGIN THETA0:=.25; THETANM1:=.75;
115             IF B<1 THEN
116             BEGIN C:=MU[N-1]:=2/3; LABDA[N-1]:=5/12;
```

```
117                 FOR J:=N-2 STEP -1 UNTIL 1 DO
118                 BEGIN C:=MU[J]:=C/(C-.25)/(N-J+1);
119                     LABDA[J]:=C-.25
120                 END
121             END ELSE
122             BEGIN C:=MU[N-1]:=BETA[2]*4/3; LABDA[N-1]:=C-.25;
123                 FOR J:=N-2 STEP -1 UNTIL 1 DO
124                 BEGIN C:=MU[J]:=C/(C-.25)*BETA[N-J+1]/BETA[N-J]/
125                         (IF J<L THEN 8 ELSE 1);
126                     LABDA[J]:=C-.25
127                 END
128             END
129         END ELSE
130         BEGIN THETA0:=0; THETANM1:=1;
131             IF B<1 THEN
132             BEGIN FOR J:=N-1 STEP -1 UNTIL 1 DO MU[J]:=LABDA[J]:=1/(N-J+1)
133             END ELSE
134             BEGIN LABDA[N-1]:=MU[N-1]:=BETA[2];
135                 FOR J:=N-2 STEP -1 UNTIL 1 DO
136                 MU[J]:=LABDA[J]:=BETA[N-J+1]/BETA[N-J]/(IF J<L THEN 8 ELSE 1)
137             END
138         END
139     END COEFFICIENT;
140
141     PROCEDURE STEPSIZE;
142     BEGIN REAL D,TAUSTAB,TAUSTABINT;
143         TAU:=STEP;
144         COMPLEX:= ABS(PHI-3.1416)>.01 ;
145         IF COMPLEX^EVEN(L)=-1 THEN
146         BEGIN NLCR; PRINTTEXT(£ZL COMPLEX EN L NIET EVEN$);
147             GOTO END OF ERR
148         END;
149         D:= IF COMPLEX THEN (SIGMA/DIAMETER/SIN(PHI))↓(.5*L/R)
150             ELSE (2*SIGMA/DIAMETER)↓(L/R);
151         TAUSTAB:=ABS(D/BETAR/SIGMA);
152         D:= IF THIRDORDER THEN (2*₁₀12*TOL/BETA[R])↓(1/(N-1))*4↓((L-1)/(N-1))
153             ELSE (₁₀12*TOL)↓(1/R)/BETAR;
154         TAUSTABINT:= ABS(D/SIGMA);
155         IF TAU>TAUSTAB THEN TAU:=TAUSTAB;
156         IF TAU>TAUSTABINT THEN TAU:=TAUSTABINT;
157         IF T+TAU>TE*(1-₁₀-11) THEN TAU:=TE-T;
158         B:=TAU*SIGMA; D:=DIAMETER*.1*TAU; D:=D*D;
159         IF TAU<T*₁₀-12 THEN GOTO END OF ERR;
160         CHANGE:= B0=0 ∨ ((B-B0)*(B-B0)+B*B0*(PHI-PHI0)*(PHI-PHI0)>D)
161     END STEPSIZE;
162
163     PROCEDURE DIFFERENCE SCHEME;
164     BEGIN INTEGER I,J; REAL MT,LT,THT;
165         I:=-1;
166         NEXT TERM:
167         I:=I+1; MT:=MU[I]*TAU; LT:=LABDA[I]*TAU;
168         FOR J:=M0 STEP 1 UNTIL M DO RL[J]:=U[J]+LT*RL[J];
169         DERIVATIVE(T+MT,RL);
170         IF I=0∨I=N-1 THEN
171         BEGIN THT:=IF I=0 THEN THETA0*TAU ELSE THETANM1*TAU;
172             FOR J:=M0 STEP 1 UNTIL M DO U[J]:=U[J]+THT*RL[J]
173         END;
174         IF I<N-1 THEN GOTO NEXT TERM;
175         T:=T+TAU
176     END DIFFERENCE SCHEME;
```

```
177
178         N:=R+L; FIRST:=TRUE; B0:=PHI0:=0; BETAR:=BETA[R]↓(1/R);
179     NEXT LEVEL:
180         STEPSIZE;
181         IF CHANGE THEN COEFFICIENT;
182         K:=K+1;
183         DIFFERENCE SCHEME;
184         OUTPUT;
185         IF T<TE THEN GOTO NEXT LEVEL;
186     END OF EFRK;
187     END EXPONENTIAL FITTED RUNGE KUTTA;
188
189     REAL T, STEP, LN10;
190     INTEGER I, J, K, L, M, R;
191     ARRAY U, UEX[1:2], B[0:3], BETA[0:6];
192     BOOLEAN OUT;
193
194     PROCEDURE DERIVATIVE(X, V); REAL X; ARRAY V;
195     BEGIN   REAL V1, V2;
196             V1:= V[1]; V2:= V[2];
197             V[1]:= -V1 + .99 * V2 + V1 * V2;
198             V[2]:= -3 * (V1 - V2 - V1 * V2)
199     END DERIVATIVE;
200
201     PROCEDURE OUTPUT;
202     BEGIN   IF M = 1 ^ K = 1 THEN BEGIN CARRIAGE(2); PRSYM(R); SPACE(2); PRSYM(L) END;
203             IF T ≥ 50 THEN
204             BEGIN   PRSYM(127);
205                     FOR J:= 1, 2 DO FIXT(3, 1, - LN(ABS(U[J] - UEX[J])) / LN10);
206                     ABSFIXT(4, 0, K); OUT:= TRUE
207             END
208     END OUTPUT;
209
210     UEX[1]:= .765878 3202487; UEX[2]:= .433710 3535768; LN10:= LN(10);
211     B[0]:= B[1]:= 1; B[2]:= .5; B[3]:= 1/6;
212
213     PRINTTEXT(↓STEP↓); FOR STEP:= 10, 5, 2, 1, .5, .2, .1 DO
214     BEGIN   PRSYM(127); ABSFIXT(10, 1, STEP); SPACE(6) END;
215     SPACE(4); FOR I:= 1 STEP 1 UNTIL 7 DO PRINTTEXT(↓| AANTAL CORRECTE   ↓);
216     SPACE(4); FOR I:= 1 STEP 1 UNTIL 7 DO PRINTTEXT(↓| DECIMALEN IN      ↓);
217     PRINTTEXT(↓R  L↓);
218     FOR I:= 1 STEP 1 UNTIL 7 DO PRINTTEXT(↓|   Y1      Y2      K ↓);
219     FOR I:= 1 STEP 1 UNTIL 144 DO PRSYM(65);
220
221     FOR I:= 1 STEP 1 UNTIL 6 DO
222     BEGIN   R:= IF I < 4 THEN 1 ELSE IF I < 6 THEN 2 ELSE 3;
223             L:= IF I = 1 THEN 1 ELSE IF I = 2 ∨ I = 4 THEN 2 ELSE 3;
224
225             M:= 0; FOR STEP:= 10, 5, 2, 1, .5, .2, .1 DO
226             BEGIN   M:= M + 1; OUT:= FALSE;
227                     FOR J:= 0 STEP 1 UNTIL R DO BETA[J]:= B[J];
228                     T:= 0; K:= 0; U[1]:= 1; U[2]:= 0;
229
230                     EXPONENTIAL FITTED RUNGE KUTTA(T, 50, 1, 2, U, -3 * (U[1] + 1), 3.14,
231                     2 * (.99 + U[1]), DERIVATIVE, K, STEP, R, L, BETA, R + 3, -4, OUTPUT);
232
233                     IF ¬ OUT THEN BEGIN PRINTTEXT(↓| STEP < T*-12↓); ABSFIXT(3, 0, K) END
234             END
235     END
236     END
```

STEP		10.0			5.0			2.0			1.0			.5			.2			.1		
		AANTAL CORRECTE DECIMALEN IN			AANTAL CORRECTE DECIMALEN IN			AANTAL CORRECTE DECIMALEN IN			AANTAL CORRECTE DECIMALEN IN			AANTAL CORRECTE DECIMALEN IN			AANTAL CORRECTE DECIMALEN IN			AANTAL CORRECTE DECIMALEN IN		
R	L	Y1	Y2	K	Y1	Y2	K	Y1	Y2	K	Y1	Y2	K	Y1	Y2	K	Y1	Y2	K	Y1	Y2	K
1	1	+3.8	+4.3	94	+3.8	+4.3	94	+3.8	+4.3	94	+3.8	+4.3	94	+3.8	+4.3	100	+4.2	+4.7	250	+4.5	+5.0	501
1	2	STEP < T_{em}-12		5	STEP < T_{em}-12		7	+3.2	+3.7	25	+3.5	+4.0	50	+3.8	+4.3	100	+4.2	+4.7	250	+4.5	+5.0	501
1	3	+2.5	+3.2	5	+2.8	+3.4	10	+3.2	+3.7	25	+3.5	+4.0	50	+3.8	+4.3	100	+4.2	+4.7	250	+4.5	+5.0	501
2	2	STEP < T_{em}-12		4	STEP < T_{em}-12		4	STEP < T_{em}-12		4	STEP < T_{em}-12		4	STEP < T_{em}-12		4	STEP < T_{em}-12		4	STEP < T_{em}-12		4
2	3	STEP < T_{em}-12		4	STEP < T_{em}-12		4	STEP < T_{em}-12		7	+5.3	+2.3	50	+6.6	+3.6	100	+7.7	+4.9	250	+8.5	+5.9	501
3	3	STEP < T_{em}-12		3	STEP < T_{em}-12		3	STEP < T_{em}-12		3	STEP < T_{em}-12		3	STEP < T_{em}-12		3	STEP < T_{em}-12		3	STEP < T_{em}-12		3

2.12. Opgaven

1. Vind de waarde in x = 1 van de oplossing van het beginwaardeprobleem

$$y''' + 1000y'' + 1001000y' + 10^6 y = 0,$$

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

2. Gegeven het beginwaardeprobleem (Liniger-Willoughby [1970])

$$y_1' = .2(y_2 - y_1),$$

$$y_2' = 10y_1 - (60 + \tfrac{x}{8})\, y_2 + .124x,$$

$$y_1(0) = y_2(0) = 0.$$

Gevraagd $y_1(.4)$, $y_2(.4)$, $y_1(10)$ en $y_2(10)$.

3. Gegeven het beginwaardeprobleem (Lapidus-Seinfeld [1971])

$$y' = -200y - e^{-x}(199y+1991) + 2000,$$

$$y(0) = 10.$$

Gevraagd $y(-4)$ en $y(10)$.

4. Gegeven het beginwaardeprobleem (Liniger-Willoughby [1968])

$$y_1' = .01 - [1001(1+y_1) + y_1^2]\,[.01 + y_1 + y_2],$$

$$y_2' = -.01y_2^2 - y_1 - y_2 - y_1y_2^2 - y_2^3,$$

$$y_1(0) = y_2(0) = 0.$$

Gevraagd $y_1(100)$ en $y_2(100)$.

5. Gegeven het beginwaardeprobleem (Robertson [1964])

$$y_1' = .04y_1 + .01y_2y_3,$$

$$y_2' = 400y_1 - 100y_2y_3 - 3000y_2^2,$$

$$y_3' = 30y_2^2,$$

$$y_1(0) = 1, \quad y_2(0) = y_3(0) = 0.$$

Gevraagd $y_1(40)$, $y_2(40)$ en $y_3(40)$.

2.13. Literatuur

BEENTJES, P.A. & DEKKER, K. *Een 5^e orde 6-punts Runge-Kutta formule met optimale stabiliteitsgrens*, rapport NR 27/72, Mathematisch Centrum, Amsterdam. 1972

DEKKER, K. *Een ALGOL 60 versie van exponentieel aangepaste Runge-Kutta methoden*, rapport NR 25/72, Mathematisch Centrum, Amsterdam. 1972

DEKKER, T.J., HEMKER, P.W. & HOUWEN, P.J. VAN DER *Colloquium Stijve Differentiaalvergelijkingen*, MC Syllabus 15, Mathematisch Centrum, Amsterdam. 1972

HOUWEN, P.J. VAN DER *Stabilized Runge-Kutta-methods with limited storage requirements*, report TW 124, Mathematisch Centrum, Amsterdam. 1971

HOUWEN, P.J. VAN DER *A survey of stabilized Runge-Kutta formulae*, MC Tracts 37, Mathematisch Centrum, Amsterdam, 1971.

HOUWEN, P.J. VAN DER *Explicit Runge-Kutta formulas with increased stability boundaries*, Numer. Math., 20, 1972, 149.

KUTTA, W. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*, Zeitschr. Math. und Phys., 46, 1901, 435,

LAPIDUS, L. & Seinfeld, J.H. *Numerical solution of differential equations*, Academic Press, New York. 1971.

LINIGER, W. & WILLOUGHBY, R.A. *Efficient integration methods for stiff systems of ordinary differential equations*, SIAM J. Numer. Anal., I (1970) 47.

POPE, D.A. *An exponential method of numerical integration of ordinary differential equations*, Comm. ACM, 6 (1963), 491.

MATHEMATICAL CENTRE SYLLABUS 20, 1975, III, 49-78.

GESTABILISEERDE RUNGE-KUTTA METHODEN VOOR PARABOLISCHE EN HYPERBOLISCHE DIFFERENTIAALVERGELIJKINGEN

E. SLAGT

3.1. Definities en afspraken

Een vergelijking van de vorm

$$F(x,y,\ldots;u,u_x,u_y,\ldots;u_{xx},u_{xy},\ldots) = 0 \tag{3.1}$$

heet een *partiële* differentiaalvergelijking (p.d.v). Hierin stellen u_x, u_{xy} de partiële afgeleiden $\frac{\partial u}{\partial x}$, resp. $\frac{\partial^2 u}{\partial y \partial x}$ voor.

We zoeken naar een functie $u(x,y,\ldots)$, die aan (3.1) voldoet. Bovendien moet de oplossing u in het algemeen aan verschillende extra voorwaarden voldoen, De graad van de hoogste afgeleide, die in de p.d.v. voorkomt heet de *orde* van de p.d.v..

Voorbeeld: 1^e orde: $u_x - cu_y = 0$.

2^e orde: $u_{xx} - c^2 u_{yy} = 0$,

$$u_x - \frac{\partial}{\partial y} c(y) \frac{\partial u}{\partial y} = 0.$$

Iedere hogere orde p.d.v. kan geschreven worden als een stelsel 1^e orde p.d.v.n..

Voorbeeld: $u_{xx} = u_{yy}$ gaat door $u_x = v$ en $u_y = w$ over in het stelsel 1^e orde p.d.v.n.:

$$\begin{cases} v_x = w_y, \\ w_x = v_y. \end{cases}$$

We kunnen ons dus beperken tot stelsels 1e orde p.d.v.n..

Een dergelijk stelsel wordt gegeven door

$$(3.2) \qquad \sum_{i=1}^{n} A_i \vec{u}_{x_i} = \vec{b},$$

waarin $\vec{u} = (u_1,\ldots,u_m)^T$,

A_i: een n×m matrix.

Dit stelsel heet:

homogeen, als $\vec{b} = 0$,

lineair met constante coëfficiënten, als alle elementen van A_i constant zijn,

lineair, als alle elementen van A_i en b differentieerbare functies van $x_1,\ldots,x_n$ zijn.

quasi-lineair, als er elementen van A_i en $\vec{b}$ zijn, die behalve van $x_1,\ldots,x_n$ ook nog van $\vec{u}$ afhangen.

Veelal werken we met p.d.v.n. in 2 onafhankelijke variabelen x en t.

Voorbeeld: $u_t + uu_x = 1$ (quasi-lineair, 1^e orde, inhomogeen).

3.2. Eerst orde quasi-lineaire partiële differentiaalvergelijkingen

Een dergelijke vergelijking wordt in 2 onafhankelijke variabelen algemeen gegeven door

$$(3.3) \qquad au_x + bu_y = c,$$

waarin a, b en c differentieerbare functies van x,y en u zijn. Voor de oplossing u(x,y) moet gelden, dat ieder punt P met coördinaten (x_p,y_p,u_p) een raakvlak aan het integraaloppervlak u(x,y) heeft, waarvan de richtingscoëffeciënten aan (3.3) voldoen. Anders gezegd

$$(3.4a) \qquad dx : dy : du = a : b : c.$$

Het stelsel gewone differentiaalvergelijkingen (3.4a) definieert de *karak-*

teristieke krommen van de p.d.v.. Invoering van de parameter s leidt tot

$$(3.4b) \qquad \frac{dx}{ds} = a; \quad \frac{dy}{ds} = b, \quad \frac{du}{ds} = c.$$

Het beginwaarde- of Cauchy-probleem

Zij geëist, dat een oplossing u van (3.3) gaat door de beginkromme

$$(3.5) \qquad x = x(t); \quad y = y(t); \quad u = u(t),$$

dan bepalen (3.4b) en (3.5) een schaar karakteristieke krommen

$$(3.6) \qquad x = x(s,t); \quad y = y(s,t); \quad u = u(s,t).$$

Voldoende voorwaarde om de parameters s en t in x en y uit te drukken is

$$\Delta = x_s y_t - y_s x_t = a y_t - b x_t \neq 0.$$

In dat geval volgt de oplossing u(x,y) onmiddellijk uit de nog niet gebruikte vergelijking van (3.6) (Courant-Hilbert [1962]).

Voorbeeld:

$$\begin{cases} u u_x + u_y = 0, & y > 0 \\ u(x,0) = x. \end{cases}$$

De karakteristieke differentiaalvergelijkingen zijn

$$\frac{dx}{ds} = u, \quad \frac{dy}{ds} = 1, \quad \frac{du}{ds} = 0.$$

De oplossing van dit stelsel gewone differentiaalvergelijkingen is

$$\begin{cases} x = x_0 + u_0 s, \\ y = y_0 + s, \\ u = u_0 \end{cases}$$

De beginkromme in parametervorm is

$$y_0 = 0, \qquad x_0 = u_0 = p.$$

Dit geeft

$$\begin{cases} x = p(1+s), \\ y = s, \\ u = p. \end{cases}$$

Door eliminatie van s verkrijgen we de karakteristieken (zie figuur 3.1):

$$y = p^{-1}x - 1.$$

s en p elimineren geeft de oplossing van de p.d.v. met de gegeven beginvoorwaarde

$$u(x,y) = \frac{x}{y+1}.$$

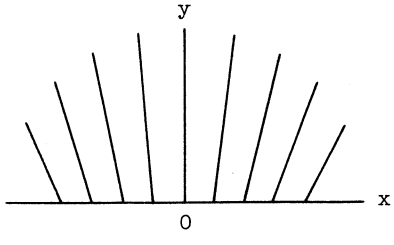


fig. 3.1. Karakteristieken.

3.3. Tweede orde quasi-lineaire partiële differentiaalvergelijkingen

In 2 onafhankelijke variabelen worden deze algemeen gegeven door

$$au_{xx} + 2bu_{xy} + cu_{yy} = f(x,y,u,u_x,u_y). \tag{3.7}$$

Met deze p.d.v. associëren we een *karakteristieke vergelijking*

$$a\lambda^2 - 2b\lambda + c = 0, \tag{3.8}$$

en de karakteristieken volgen uit

$$\frac{dy}{dx} = \lambda. \tag{3.9}$$

Voor deze 2^e orde p.d.v.n. kennen we de volgende type-indeling:
Zij $D = b^2 - ac$, dan heet de p.d.v.

$$\begin{cases} \textit{hyperbolisch} & \text{als } D > 0, \\ \textit{parabolisch} & \text{als } D = 0, \\ \textit{elliptisch} & \text{als } D < 0, \end{cases} \quad \text{(Courant-Hilbert [1962]).}$$

Het laatste type zal in hoofdstuk 4 behandeld worden.

Voorbeeld:

$$u_{xx} - yu_{yy} = u_y$$

dus $a = 1$, $b = 0$, $c = -y$. De karakteristieke vergelijking is

$$\lambda^2 - y = 0 \qquad \text{met oplossing} \quad \lambda = \pm y^{\frac{1}{2}}$$

De karakteristieken volgen uit

$$\frac{dy}{dx} = \pm y^{\frac{1}{2}}$$

en hebben de gedaante

$$x = \pm 2y^{\frac{1}{2}} + c.$$

De karakteristieken zijn dus parabolen met vergelijking (zie figuur 3.2):

$$y = \tfrac{1}{4}(x-c)^2.$$

Het type volgt uit $D = y$.
De p.d.v. is derhalve:

$$\begin{cases} \text{parabolisch} & \text{voor } y = 0, \\ \text{hyperbolisch} & \text{voor } y > 0, \\ \text{elliptisch} & \text{voor } y < 0. \end{cases}$$

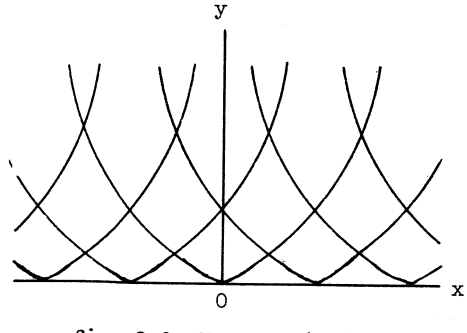


fig. 3.2. Karakteristieken

3.4. De methode der lijnen

In deze paragraaf bespreken we de theorie en een aantal voorbeelden

van de *methode der lijnen* (method of lines).

Zij gegeven een (quasi-) lineaire p.d.v. van de 1^e orde in x en t

$$(3.10) \qquad au_x + bu_t = c \quad \text{op R:} \begin{cases} -L \le x \le L \\ 0 \le t \le T. \end{cases}$$

We schrijven nu de tijdafgeleide expliciet

$$u_t = -\frac{a}{b} u_x + \frac{c}{b} .$$

Vervolgens verdelen we het x-interval [-L,L] door middel van roosterpunten $x_j = j \times h$ (j=-l,...,l), waarin

$$h = \Delta x = \frac{L}{l} .$$

Vervanging van de operator $\frac{\partial}{\partial x}$ in (3.10) door een differentiequotiënt geeft een stelsel gewone differentiaalvergelijkingen

$$(3.11) \qquad \frac{d\vec{u}}{dt} = D\vec{u} + \vec{F}(t,x).$$

De vector $\vec{u}$ wordt gevormd door de componenten $u_{-l+1},\ldots,u_{l-1}$ en de operator D is het discrete analogon van de operator $-a/b\ \partial/\partial x$.

Door deze discretisatie introduceren we een fout van een zekere orde q in h: $O(h^q)$.

Voorbeeld:

$$\frac{\partial}{\partial x} \rightarrow \frac{E_+ - E_-}{2h}$$

waarbij E de translatieoperator is.

$$E_+ u_j = u_{j+1} = u_j + h(u_x)_j + \frac{h^2}{2}(u_{xx})_j + \frac{h^3}{6}(u_{xxx})_j + \ldots$$

$$E_- u_j = u_{j-1} = u_j - h(u_x)_j + \frac{h^2}{2}(u_{xx})_j - \frac{h^3}{6}(u_{xxx})_j + \ldots$$

dus

$$\frac{E_+ - E_-}{2h} u_j = (u_x)_j + \frac{h^2}{6}(u_{xxx})_j + \ldots$$

Dit houdt in, dat

$$\frac{E_+ - E_-}{2h} u = \frac{\partial u}{\partial x} + O(h^2).$$

De beginvoorwaarde $u(x,0) = f(x)$ wordt nu

$$u_0 = \{f(j\times h)\}_j \qquad (j=-l,\ldots,l),$$

terwijl de eventuele randvoorwaarden $u(-L,t) = g(t)$ of $u(L,t) = h(t)$ in de vector $\vec{F}$ opgenomen worden, zie (3.11).

Als we uitgaan van een hogere orde p.d.v., dan moeten we die eerst omzetten in een stelsel 1^e orde p.d.v.n., waarna omzetting in een stelsel gewone differentiaalvergelijkingen volgt (Van der Houwen e.a. [1971]).

Voorbeelden

1.
$$\begin{cases} u_t = \lambda u_x & \text{op } -\infty < x \le L \text{ voor } t > 0 \ (\lambda>0), \\ u(x,0) = f(x) , \\ u(L,t) = g(t). \end{cases}$$

Discretisering van de plaatsafgeleide geeft:

$$\frac{d\vec{u}}{dt} = \lambda \frac{E_+ - E_-}{2h} \vec{u} + \vec{F}(t).$$

Volledig uitgeschreven wordt dit stelsel:

$$\frac{d}{dt}\begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ u_j \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ u_{l-1} \end{bmatrix} = \frac{\lambda}{2h} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \ldots & 0 & -1 & 0 & 1 & 0 & \ldots \ 0 \\ & & & & & & \cdot \\ & & & & & & \cdot \\ & & & & & & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & -1 & 0 \ 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -1 \ 0 \end{bmatrix} \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ u_j \\ \cdot \\ \cdot \\ \cdot \\ u_{l-1} \end{bmatrix} + \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \frac{\lambda}{2h} g(t) \end{bmatrix}$$

Als we aannemen, dat we een integratiemethode hebben, die het volgende tijdsniveau (t_{n+1}) direct uit het vorige berekend (t_n), dan krijgen we het volgende afhankelijkheidsgebied in het x,t-vlak:

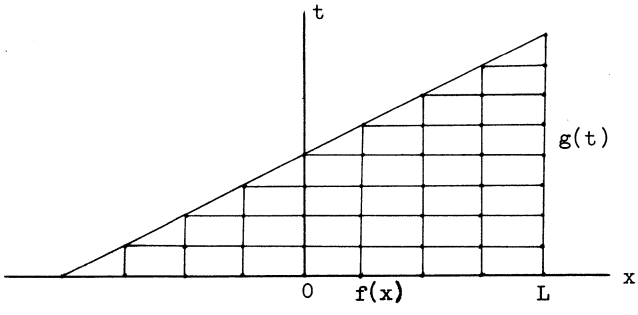


fig. 3.3. Afhankelijkheidsgebied.

Per niveau verliezen we een punt uit het x-interval door de gekozen discretisatie van de operator $\partial/\partial x$. Als de integratie-methode gebruik maakt van m tussenniveau's, dan verliezen we m punten per tijdstap.

Willen we nu de oplossing weten op de rechthoek

$$R : \begin{cases} -L \le x \le L \\ \\ 0 \le t \le T, \end{cases}$$

dan schatten we hoeveel integratiestappen, zeg K, nodig zijn om het probleem tot t = T te integreren en we moeten dan starten met 2l + Km punten op de x-as.

Voor een homogene 1^e orde p.d.v. geldt, dat de oplossing constant is langs een karakteristiek, want daar geldt immers

$$du = u_x dx + u_t dt = (au_x + bu_t)ds = cds = 0$$

In ons voorbeeld zijn de karakteristieken

$$t = -\lambda^{-1}x + c.$$

De richtingscoëfficiënt van deze evenwijdige rechten is negatief en daarom

mag alleen een rechterrand voorgeschreven worden, wil het probleem goed gesteld zijn.

2.
$$\begin{cases} u_t = u_{xx}, \\ u(x,0) = f(x), \\ u(-L,t) = g(t), \\ u(L,t) = h(t). \end{cases}$$

Deze tweede orde p.d.v. is eerste orde in t en kan dus direct in de vorm

$$\frac{d\vec{u}}{dt} = D\vec{u} + \vec{F}(t); \qquad u_0 = \{f(j\times h)\}$$

geschreven worden, waarin

$$D = h^{-2}\begin{bmatrix} 2 & 1 & \cdots & & 0 \\ 1 & -2 & 1 & & \cdot \\ 0 & 1 & -2 & 1 & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & 1 & -2 & 1 \\ 0 & \cdots & & 0 & 1 & -2 \end{bmatrix} \quad \text{en} \quad \vec{F}(t) = h^{-2}\begin{bmatrix} g(t) \\ 0 \\ \cdot \\ \cdot \\ 0 \\ h(t) \end{bmatrix}$$

3.
$$\begin{cases} tu_{tt} + u_t = u_{xx}, & 0 < x < 2\pi, \\ u(x,0) = f(x), \\ u(0,t) = u(2\pi,t) = g(t). \end{cases}$$

Substitutie van $u_t = v$ resulteert in het 2×2-stelsel

$$\begin{cases} \begin{pmatrix} u \\ v \end{pmatrix}_t = \begin{pmatrix} 0 & 1 \\ 1/t\ \partial^2/\partial x^2 & -1/t \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}, \\ u(0,x) = f(x), \\ v(0,x) = f''(x), \\ u(0,t) = u(2\pi,t) = g(t), \\ v(0,t) = v(2\pi,t) = g'(t). \end{cases}$$

Vervangen we de operator $\partial^2/\partial x^2$ door $\dfrac{E_+ - 2 + E_-}{h^2}$, dan verkrijgen we het volgende stelsel:

$$\frac{d\vec{r}}{dt} = \begin{bmatrix} 0 & \cdots & 0 & 1 & & \\ & & & 0 & \ddots & \\ & & & & \ddots & 1 \\ & & & & & 0 \\ -2/h^2t & 1/h^2t & 0 & \cdots & 0 & -1/t \\ 1/h^2t & -2/h^2t & 1/h^2t & & & 0 \\ & \ddots & \ddots & \ddots & & \\ 0 & & 1/h^2t & -2/h^2t & 1/h^2t & 0 \\ & & & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & 1/h^2t & -2/h^2t & -1/t \end{bmatrix} \vec{r} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ g(t)/h^2t \\ 0 \\ \vdots \\ 0 \\ g(t)/h^2t \end{bmatrix}$$

Hierin bestaat de vector r uit de componenten $r_1,\ldots,r_{2N}$ en $(r_1,\ldots,r_N) = (u_1,\ldots,u_N)$, $(r_{N+1},\ldots,r_{2N}) = (v_1,\ldots,v_N)$ $(N=\frac{2\pi}{h} - 1)$.
De beginvoorwaarde wordt:

$$(r_0)_j = \begin{cases} f(j\times h), & 1 \le j \le N, \\ f''((j-N)h), & N+1 \le j \le 2N. \end{cases}$$

We hebben nu gezien, dat p.d.v.n. met begin- en randvoorwaarden terug te brengen zijn tot het Cauchy-probleem voor het stelsel gewone differentiaalvergelijkingen

$$\begin{cases} \dfrac{d\vec{u}}{dt} = D\vec{u} + \vec{F}(t), \\ \vec{u}_0 = f(jh), \qquad (j=0,\ldots,N). \end{cases}$$

3.5. Gestabiliseerde Runge-Kutta methoden

In de vorige paragrafen van dit hoofdstuk hebben we gezien, dat de partiële differentiaalvergelijkingen van het hyperbolische en parabolische type gebracht kunnen worden op een vorm, die numeriek behandeld kan worden

met methoden, waarvan de theorie grotendeels in het begin van hoofdstuk 2 behandeld is. Hierin werden gestabiliseerde Runge-Kutta methoden geïntroduceerd, die gegenereerd werden door een matrix

$$(3.12) \qquad (\lambda_{j,l}) = \begin{bmatrix} \lambda_{10} & 0 \ldots 0 \\ \lambda_{20} & \lambda_{21} \\ & & 0 \\ \lambda_{m0} & \ldots\ldots & \lambda_{mm-1} \end{bmatrix}$$

De parameters $\lambda_{j,l}$ werden bepaald door

a) consistentievoorwaarden (orde van nauwkeurigheid p),

b) stabiliteitsvoorwaarden (voorwaarden voor de stapgrootte τ),

c) voorwaarden voor eenvoud en geheugenbeperking.

Bovendien zagen we, dat bij iedere Runga- Kutta methode een stabiliteits-polynoom

$$(3.13) \qquad P_m(z) = 1 + z + \ldots + \frac{1}{p!} z^p + \beta_{p+1} z^{p+1} + \ldots + \beta_m z^m$$

hoort, met bijbehorend stabiliteitsgebied

$$(3.14) \qquad S: \{z \mid |P_m(z)| < 1\}.$$

Als S enkelvoudig samenhangend is heten de absolute waarden van het snijpunt van de randkromme ∂S van S met de negatieve en imaginaire as de *reële* en *imaginaire stabiliteitsgrens* resp. $\beta_{re}(m)$ en $\beta_{im}(m)$. De straal van de cirkel met middelpunt in 0, die in het linker halfvlak nog juist tot $S \cup \partial S$ behoort zullen we de *absolute stabiliteitsgrens* $\beta_{abs}(m)$ noemen. De nog vrije parameters $\beta_{p+1},\ldots,\beta_m$ worden nu zo gekozen, dat $\beta(m)$ maximaal is. De gebruikte methode is nu stabiel als alle grootheden $\tau\delta$ met δ eigenwaarde van D in S liggen.

Voor enkelvoudig samenhangende stabiliteitsgebieden geldt dus:

$$(3.15) \qquad \tau \leq \frac{\beta_{abs}(m)}{\sigma(D)}$$

met $\sigma(D)$ de spectraalradius van de matrix D. Het is dus van het grootste belang om de ligging van de eigenwaarden van de Jacobiaan D van het te integreren stelsel te kennen.

In het algemeen zijn de eigenwaarden van parabolische vergelijkingen negatief en van hyperbolische vergelijkingen zuiver imaginair. Dit is de reden, dat er voor ieder type een klasse van stabiliteitspolynomen ontwikkeld is(Van der Houwen [1970]).

Enkele belangrijke stabiliteitspolynomen zijn:

δ imaginair:

$P_2(z) = 1 + z + z^2$	$\beta_{im}(2) = 1$	1^e orde
$P_3(z) = 1 + z + 1/2z^2 + 1/4z^3$	$\beta_{im}(3) = 2$	2^e orde
$P_4(z) = 1 + z + 1/2z^2 + 1/6z^3 + 1/24z^4$	$\beta_{im}(4) = 2\sqrt{2}$	4^e orde
$P_5(z) = 1 + z + 1/2z^2 + 3/16z^3 + 1/32z^4 + 1/128z^5$	$\beta_{im}(5) = 4$	2^e orde

δ reëel:

$P_2(x) = 1 + x + 1/8x^2$	$\beta_{re}(2) = 8$	1^e orde
$P_3(x) = 1 + x + 1/2x^2 + 1/4x^3$	$\beta_{re}(3) = 6.27$	2^e orde
$P_4(x) = 1 + x + 1/2x^2 + 1/6x^3 + .0184557x^4$	$\beta_{re}(4) = 6$	3^e orde
of $P_4(x) = 1 + x + 1/2x^2 + .078x^3 + .0036x^4$	$\beta_{re}(4) \approx 12$	2^e orde

In stelling 2.1 (hoofdstuk 2) hebben we gezien, dat er 1^e en 2^e orde Runga-Kutta formules zijn, die gegenereerd worden door matrices met slechts elementen op de hoofddiagonaal.

De grootheden $\lambda_{j,j-1}$ $(j=1,\ldots,m-1)$ worden gegeven door

$$\lambda_{j,j-1} = \beta_{m+1-j}/\beta_{m-j}. \tag{3.16}$$

De genererende matices behorend bij de eerste twee polynomen van elke zojuist genoemde klasse zijn derhalve van de gedaante

$$(\lambda_{j,1}) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad (\lambda_{j,1}) = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

en

$$(\lambda_{j,1}) = \begin{bmatrix} 1/8 & 0 \\ 0 & 1 \end{bmatrix}, \qquad (\lambda_{j,1}) = \begin{bmatrix} 1/8 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

In de volgende paragraaf zullen we een procedure behandelen, welke op het Mathematisch Centrum ontwikkeld is. Voor deze procedure "modified runge kutta" behoeven door de gebruiker niet de elementen van de matrix $(\lambda_{j,1})$ gegeven te worden, maar de coëfficiënten van het stabiliteitspolynoom en de gewenste orde van nauwkeurigheid p = 1, 2 of 3. De procedure berekent dan zelf deze matrixelementen. We zullen daarom verder geen aandacht besteden aan het opstellen van de matrices voor hogere orden.

3.6. De procedure "modified runge kutta"

Een ALGOL 60-versie van deze procedure vindt men in Beentjes [1972]. Deze algoritme integreert beginwaardeproblemen van het type (3.11). De stapgrootte wordt bepaald op grond van de door de gebruiker gewenste absolute en relatieve locale precisie en de stabiliteitsvoorwaarde (3.15). De procedure "modified runge kutta" is gedocumenteerd in de LR-uitgave van het Mathematisch Centrum (LR 3.3.6). Volledigheidshalve volgt nu de gebruiksaanwijzing met referenties naar de in dit hoofdstuk gegeven theorie.

Declaratie

```
procedure modified runge kutta (t,te,m0,m,u,sigma,i,derivative,k,data,
          alfa,norm,aeta,reta,eta,rho,output);
integer   m0,m,i,k,norm;
real      t,te,sigma,alfa,aeta,reta,eta,rho;
array     u,data;
procedure derivative,output;
```

Parameters

t : <variable>
bij de aanroep van modified runge kutta moet t de beginwaarde

van de variabele waarover geïntegreerd wordt hebben;

te : <expression>;
de eindwaarde van t;

m0,m : <expression>;
indices van de eerste en laatste vergelijking van het stelsel gewone differentiaalvergelijkingen (3.11);

u : <array identifier>; array u[m0,m];
bij de aanroep van modified runge kutta moet dit array de startwaarden $u(t_0)$ bevatten;

sigma : <expression>;
de absolute waarde van de in modulus grootste eigenwaarde van de Jacobiaan D, die niet in het positieve halfvlak ligt;
sigma moet door de gebruiker gegeven worden;

i : <variable>
telt het aantal evaluaties van het rechterlid tijdens een integratiestap en loopt van 0 tot en met n-1;

derivative : <procedure identifier>
in deze procedure geeft de gebruiker het rechterlid van vergelijking (3.11);
de heading van de procedure luidt:
procedure derivative (t,v); real t; array v;
<body>;

k : <variable>;
telt het aantal integratiestappen;

data : <array identifier>; array data [-3:data[-2]];
een 1-dimensonaal array, dat de volgende door de gebruiker te geven informatie moet bevatten:
data[-3]: het aantal rechterlid evaluaties, dat men wenst te gebruiken voor een schatting van de locale fout;
data[-2]: de graad van het stabiliteitspolynoom n;
data[-1]: orde van nauwkeurigheid p;
data[0] : de stabiliteitsparameter $\beta(n)$;
data[1],...,data[data[-2]]: de coëfficiënten β_j van het stabiliteitspolynoom;
alfa : <expression>

de toename van de stapgrootte is hiermee als volgt te regelen:
$\tau_k \leq$ alfa $\times \tau_{k-1}$.

norm : <expression>;
voor norm = 1 wordt gerekend met de maximumnorm,
voor norm = 2 met de Euclidische norm;

aeta,reta : <expression>;
verlangde absolute en relatieve precisie;
zijn aeta en reta beide negatief, dan wordt de stapgrootte alleen bepaald op grond van het stabiliteitscriterium (3.15);

eta : <variable>;
de tolerantie η_k als functie van aeta, reta en $\| u_k \|$;

rho : <variable>;
de discrepantie ρ_k;

output : <procedure identifier>;
de heading van deze procedure, die door de gebruiker gegeven moet worden luidt:
procedure output;
<body>.

3.7. Voorbeelden

1. Beschouw het volgende Cauchy-probleem:

$$(3.17) \quad \begin{cases} U_t = xU_x - U, & -\infty < x < \infty,\ 0 \leq t \leq T, \\ U = x + 1, & -\infty < x < \infty,\ t = 0. \end{cases}$$

Het bijbehorende stelsel gewone differentiaalvergelijkingen is dan van de vorm

$$(3.18) \quad \begin{cases} \frac{d\vec{U}}{dt} = \left(jh \frac{E_+ - E_-}{2h} - 1 \right)\vec{U}, & j = 0, \pm 1, \pm 2, \ldots, \\ \vec{U}_0 = jh + 1. \end{cases}$$

Hierin is U een vector met oneindig veel componenten, die corresponderen met de roosterpunten jh.

De operator D is nu een matrix van oneindige orde:

$$(3.19)\qquad D = \begin{bmatrix} \ddots & \ddots & \ddots & \ddots & \ddots & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ \cdots & 0 & -j/2 & -1 & -j/2 & 0 & \cdots & & \\ & \cdots & 0 & -(j+1)/2 & -1 & (j+1)/2 & 0 & \cdots & \\ & & & 0 & -(j+2)/2 & -1 & (j+2)/2 & 0 & \cdots \\ & & & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}$$

D heeft eigenfuncties E_ω, waarvan de j-de component gegeven wordt door

$$E_\omega^{(j)} = a(t)\ \exp(i\omega jh),$$

met ω een willekeurig reëel getal. De corresponderende eigenwaarde δ van D wordt gegeven door

$$(3.20)\qquad \delta = -1 + ij \sin \omega h.$$

De spectraalradius van D is dus

$$(3.21)\qquad \sigma(D) = \max_j \sqrt{1+j^2}.$$

Omdat de eigenwaarden van D op de lijn Re(z) = -1 liggen zijn de polynomen behorend bij imaginaire eigenwaarden geschikt om dit probleem te integreren. Van deze polynomen kunnen we

$$(3.22)\qquad P_3(z) = 1 + z + 1/2z^3 + 1/4z^3 \qquad\qquad \beta_{im}(3) = 2$$

$$(3.23)\qquad P_4(z) = 1 + z + 1/2z^2 + 1/6z^3 + 1/24z^4 \qquad\qquad \beta_{im}(4) = 2\sqrt{2}$$

gebruiken voor resp. 2^e en 3^e orde exacte resultaten.

De analytische oplossing van (3.17), U(t,x) = exp(-t) + x wordt nu benaderd tot op termen, die resp.

$$O(\tau^3) + O(\tau h^2) \qquad \text{voor (3.22)}$$

en

$$O(\tau^4) + O(\tau h^2) \qquad \text{voor (3.23)}$$

bedragen.

De stabiliteitsconditie eist voor de tijdstap:

$$\tau_{stab} \leq \frac{\beta_{im}(n)}{\sigma(D)} \leq \frac{\beta_{im}(n)}{jmax+1} = O(h).$$

We maken dus op zijn minst een totale benaderingsfout van $O(h^3)$. We worden in dit voorbeeld geconfronteerd met een matrix D en vector U van oneindige orde. In werkelijkheid nemen we uiteraard een eindig aantal componenten en afhankelijk van de graad van het stabiliteitspolynoom n en het aantal punten waarin we de oplossing uiteindelijk willen weten, starten we met een van te voren berekend aantal componenten (zie 3.4.,voorbeeld 1, blz. 56).

Het aantal relevante vergelijkingen neemt gedurende het integratieproces af. Stel dat U geïnitialiseerd is in de punten x = jh, j = $g_0, g_0+1, \ldots, g$. De indices m0 en m worden tijdens het proces aangepast, door middel van een procedure m0, waarin m0 toeneemt en m afneemt.

In de procedure derivative wordt de evaluatie van het rechterlid van (3.18) beschreven.

Als we ons tot doel stellen de oplossing van (3.17) te kennen op de rechthoek

$$R: \begin{cases} -.05 \leq x \leq .05 \ (=J\times h), \\ 0 \leq t \leq .5, \end{cases}$$

en we kiezen h = .005, dan moet op t = .5 de oplossing in minstens 21 roosterpunten bekend zijn.

Voor polynoom (3.22) geldt:

$$\tau \leq \frac{2}{\sigma(D)} < \frac{2}{j_{max}+1} = \frac{2}{m+1} \leq \frac{2}{g+1},$$

en om te weten met hoeveel vergelijkingen we moeten starten lossen we de volgende vergelijking op:

$$2J+1 + 2Kn = 2g + 1, \qquad \text{waarin } K = \frac{T}{\tau},$$

ofwel

$$21 + 2 \times \frac{\cdot 5}{2} \quad (g+1) \times 3 = 2g + 1,$$

waaruit volgt

$$g = 43, \quad K \leq 11, \quad \tau \geq 1/22.$$

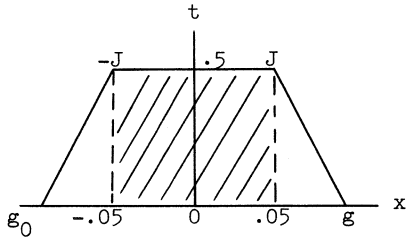


fig.3.4. Afhankelijkheidsgebied voor polynoom (3.22)

De stabiliteitsvoorwaarde schrijft een start met minstens 87 vergelijkingen voor. Nauwkeurigheidscondities kunnen ons echter tot kleinere stappen dwingen, zodat beter wat veiliger grenzen

$$g = -g_0 = 100$$

gekozen kunnen worden. We geven nu het volledige ALGOL 60 programma met de resultaten, zoals dat door de regeldrukker van de EL X8 van het Mathematisch Centrum werd afgedrukt.

Het ALGOL 60-programma

```
    BEGIN   COMMENT THE INITIAL VALUE PROBLEM UT=XUX-U, U0=X+1
 1                  BY PROCEDURE MODIFIED RUNGE KUTTA;
 2
 3
 4      PROCEDURE MODIFIED RUNGE KUTTA(T,TE,M0,M,U,SIGMA,I,DERIVATIVE,K,
 5                  DATA,ALFA,NORM,AETA,RETA,ETA,RHO,OUTPUT);
 6      INTEGER M0,M,I,K,NORM;
 7      REAL T,TE,SIGMA,ALFA,AETA,RETA,ETA,RHO;
 8      ARRAY U,DATA;
 9      PROCEDURE DERIVATIVE,OUTPUT;
10      BEGIN I:=-1;
11          BEGIN INTEGER P,N,N1,Q;
12              OWN REAL EC0,EC1,EC2,TAU0,TAU1,TAU2,TAUS,T2;
13              REAL THETA0,THETANM1,TAU,GAMMA,BETAN;
14              ARRAY MU,LABDA[0:DATA[-2]-1],BETA[1:DATA[-2]],
15                  THETHA[0:DATA[-3]-1],RO,R[M0:M];
16              BOOLEAN START,STEP1;
17
18              REAL PROCEDURE NORMFUNCTION(NRM,W); INTEGER NRM;
19              ARRAY W;
20              BEGIN INTEGER J; REAL S,X;
21                  S:=0;
22                  IF NRM=1 THEN FOR J:=M0 STEP 1 UNTIL M DO
23                  BEGIN X:=ABS(W[J]);
24                      IF X>S THEN S:=X
25                  END;
26                  IF NRM=2 THEN S:=SQRT(VECVEC(M0,M,0,W,W));
27                  NORMFUNCTION:=S
28              END NORMFU;
29
30              PROCEDURE COEFFICIENT;
31              BEGIN INTEGER J,K;REAL S;
32                  N1:=DATA[-3];N:=DATA[-2];P:=DATA[-1];
33                  BETAN:=DATA[0];GAMMA:=.5;
34                  FOR J:=1 STEP 1 UNTIL N DO BETA[J]:=DATA[J];
35                  THETANM1:=IF P=3 THEN .75 ELSE 1;
36                  THETA0:=1-THETANM1;
37                  MU[N-1]:= BETA[2]/THETANM1;
38                  LABDA[N-1]:=MU[N-1]-THETA0;
39                  LABDA[0]:=MU[0]:=0;
40                  S:=THETANM1;
41                  IF N>2 THEN
42                  FOR J:=N-2 STEP -1 UNTIL 1 DO
43                  BEGIN S:=BETA[N-J]-S*THETA0;
44                      MU[J]:=BETA[N+1-J]/S;
45                      LABDA[J]:=MU[J]-THETA0
46                  END;
47                  IF N1=2 THEN
48                  BEGIN THETHA[0]:=IF P=1 THEN (BETA[2]-.5)/LABDA[1]
49                      ELSE -.5/LABDA[1];
50                      THETHA[1]:=-THETHA[0];
51                      Q:=2
52                  END;
53                  IF N1=4 THEN
54                  BEGIN REAL A,B,C,D,E,F;
55                      E:=IF P=3 THEN 0 ELSE BETA[3];
```

```
56                      F:=IF P=3 THEN 0 ELSE BETA[2];
57                      A:=MU[1];B:=MU[2];C:=MU[3];D:=THETA0;
58                      THETHA[3]:=(B*(A-B)*(1/6-E)-A*(B-D)*(A*(.5-BETA[2])
59                        -(1/3-F*F)))/(B*(A-B)*(C-D)*B-A*C*(B-D)*(A-C));
60                      THETHA[2]:=D:=(1/6-E-B*(C-D)*THETHA[3])/(A*(B-D));
61                      THETHA[1]:=C:=(.5-BETA[2]-B*D-C*THETHA[3])/A;
62                      THETHA[0]:=-C-D-THETHA[3];
63                      Q:=IF P=1 THEN 2 ELSE 3
64                  END;
65                  IF N1=8 THEN
66                  BEGIN REAL ARRAY ALPHA[1:6,1:6],AUX[0:3];
67                      FOR J:=1 STEP 1 UNTIL 6 DO
68                      FOR K:=1 STEP 1 UNTIL 6 DO
69                      ALPHA[J,K]:=IF J=1 THEN
70                        BETA[N+1-K]/BETA[N-K] ELSE
71                        IF J=2 THEN MU[K+1]-MU[1] ELSE
72                        IF J=3 THEN (IF K=1 THEN 0 ELSE
73                        BETA[N+2-K]/BETA[N-K]) ELSE
74                        IF J=4 THEN MU[K]*ALPHA[1,K] ELSE
75                        IF J=5 THEN MU[K+1]*ALPHA[1,K] ELSE
76                        MU[K+1]*ALPHA[2,K];
77                      THETHA[1]:=1/6-BETA[3];
78                      THETHA[2]:=1/3-BETA[2]*MU[N-1]-MU[1]*(.5-BETA[2]);
79                      THETHA[3]:=1/24-BETA[4];
80                      THETHA[4]:=1/12-BETA[3]*MU[N-2];
81                      THETHA[5]:=1/8-BETA[3]*MU[N-1];
82                      THETHA[6]:=.25-BETA[2]*MU[N-1]↑2
83                      -MU[1]*(1/3-BETA[2]*MU[N-1]);
84                      AUX[0]:=10↑-12;RNKSOLELM(ALPHA,6,AUX,THETHA);
85                      S:=(.5-BETA[2]-SUM(J,1,6,THETHA[J]))/MU[1];
86                      FOR J:=7 STEP -1 UNTIL 2 DO
87                      THETHA[J]:=THETHA[J-1]/MU[J];
88                      THETHA[1]:=S;THETHA[0]:=-SUM(J,1,7,THETHA[J]);
89                      Q:=P+1
90                  END
91              END COEFFICIENT;
92
93              PROCEDURE LOCAL ERROR BOUND;
94              ETA:=AETA+RETA*NORMFUNCTION(NORM,U);
95
96              PROCEDURE LOCAL ERROR CONSTRUCTION;
97              IF I ≤ N1-1 THEN
98              BEGIN INTEGER J; REAL THT;
99                  IF I=0 THEN
100                 FOR J:=M0 STEP 1 UNTIL M DO RO[J]:=0;
101                 THT:=THETHA[I]*TAU;
102                 FOR J:=M0 STEP 1 UNTIL M DO
103                 RO[J]:=RO[J]+THT*R[J];
104                 IF I=N1-1 THEN
105                 BEGIN I:=N-1;RHO:=NORMFUNCTION(NORM,RO);I:=N1-1;
106                     EC0:=EC1;EC1:=EC2;EC2:=RHO/TAU↑Q
107                 END
108             END L.E.C.;
109
110             PROCEDURE STEPSIZE;
111             BEGIN REAL TAUACC,TAUSTAB,AA,BB,CC,EC;
112                 LOCAL ERROR BOUND;
113                 IF ETA>0 THEN
114                 BEGIN IF START THEN
115                     BEGIN IF K=0 THEN
```

```
116                     BEGIN INTEGER J;
117                         FOR J:=M0 STEP 1 UNTIL M DO
118                         R[J]:=U[J];
119                         I:=0; DERIVATIVE(T,R);
120                         TAUACC:=ETA/NORMFUNCTION(NORM,R);
121                         STEP1:=TRUE
122                     END ELSE
123                     IF STEP1 THEN
124                     BEGIN TAUACC:=(ETA/RHO)↑(1/Q)*TAU2;
125                         IF TAUACC>10*TAU2 THEN
126                         TAUACC:=10*TAU2 ELSE STEP1:=FALSE
127                     END ELSE
128                     BEGIN BB:=(EC2-EC1)/TAU1; CC:=EC2-BB*T2;
129                         EC:=BB*T+CC;
130                         TAUACC:=IF EC<0 THEN TAU2 ELSE
131                         (ETA/EC)↑(1/Q);
132                         START:=FALSE
133                     END
134                 END ELSE
135                 BEGIN AA:=((EC0-EC1)/TAU0+(EC2-EC1)/TAU1)
136                             /(TAU1+TAU0);
137                     BB:=(EC2-EC1)/TAU1-AA*(2*T2-TAU1);
138                     CC:=EC2-T2*(BB+AA*T2); EC:=CC+T*(BB+T*AA);
139                     TAUACC:=IF EC<0 THEN
140                     TAUS ELSE (ETA/EC)↑(1/Q);
141                     IF TAUACC>ALFA*TAUS THEN TAUACC:=ALFA*TAUS;
142                     IF TAUACC<GAMMA*TAUS THEN
143                     TAUACC:=GAMMA*TAUS;
144                     IF TAUACC<10-12 * T THEN TAUACC:= 10-12 * T
145                 END
146             END ELSE TAUACC:=TE-T;
147             TAUSTAB:=BETAN/SIGMA;IF TAUSTAB<10-12*T THEN
148             BEGIN PRINTTEXT(↑STABLE TIME-STEP LESS THAN 10-12*T↑);
149                 GOTO END OF MODIFIED RK
150             END;
151             TAU:=IF TAUACC>TAUSTAB THEN TAUSTAB ELSE TAUACC;
152             TAUS:=TAU; IF TAU>TE-T THEN TAU:=TE-T;
153             TAU0:=TAU1;TAU1:=TAU2;TAU2:=TAU
154         END STEPSIZE;
155
156         PROCEDURE DIFFERENCE SCHEME;
157         BEGIN INTEGER J;
158             REAL MT,LT;
159         NEXT TERM:
160             MT:=MU[I+1]*TAU;LT:=LAMDA[I+1]*TAU;
161             FOR J:=M0 STEP 1 UNTIL M DO R[J]:=U[J]+LT*R[J];
162             I:=I+1;DERIVATIVE(T+MT,R);
163             IF ETA>0 THEN LOCAL ERROR CONSTRUCTION;
164             IF (I=0∧P=3)∨I=N-1 THEN
165             BEGIN REAL THT;
166                 THT:=IF I=0 THEN THETA0*TAU ELSE THETANM1*TAU;
167                 FOR J:=M0 STEP 1 UNTIL M DO
168                 U[J]:=U[J]+THT*R[J]
169             END;
170             IF I<N-1 THEN GOTO NEXT TERM;
171             T2:=T;T:=T+TAU
172         END DIFF.SCH.;
173
174         START:=K=0;
175         COEFFICIENT;
```

```
176           NEXT LEVEL;
177             STEPSIZE[K:=K+1];I:=-1;DIFFERENCE SCHEME;OUTPUT]
178             IF T<TE THEN GOTO NEXT LEVEL
179       END;
180       END OF MODIFIED RK;
181   END MODIFIED RK;
182       INTEGER M, G0, G, J, K, PER, I;
183       REAL T,ETA, RHO, INX, H, EPS, EPSABS,AETA,RETA;
184       ARRAY DATA[-3:4];
185       T:=0; INX:=1; DATA[-3]:=READ; DATA[-2]:=READ;
186       FOR J:=-1 STEP 1 UNTIL DATA[-2] DO DATA[J]:= READ;
187       G0:=READ; G:=READ; PER:=READ;
188       H:=INX/ABS(G-G0);
189       OR←WRITEW($THE INITIAL VALUE PROBLEM WT=XWX-W, W0=X+1
190             BY PROCEDURE MODIFIED RUNGE KUTTA$);
191       NLCR; NLCR;
192       OR←WRITEW($DEGREE=$); ASSP←WV(1,0,DATA[-2]);
193       OR←WRITEW($ORDER=$); ASSP←WV(1,0,DATA[-1]);
194       OR←WRITEW($H=$); SLOV(4,2,H);
195       NLCR; NLCR;
196       OR←WRITEW($   K      T    X=-.015  -.010    -.005     0     .005    .010    .015      EPS    EPSABS$);
197       NLCR;
198       BEGIN ARRAY U[G0:G];
199       PROCEDURE ANSOL(A); ARRAY A;
200       BEGIN   INTEGER J;
201               FOR J:=G0 STEP 1 UNTIL G DO A[J]:=J+H+EXP(-T)
202       END;
203       INTEGER PROCEDURE M0;
204       BEGIN   INTEGER DECREMENT;
205               DECREMENT:=IF K=0 THEN I+1 ELSE (K=1)+DATA[-2]+I+1;
206               M0:=G0+DECREMENT; M:=G-DECREMENT
207       END;
208       PROCEDURE DERIVATIVE(T,V); REAL T; ARRAY V;
209       BEGIN   REAL VJM1, VJ;
210               VJM1:=V[M0-1];
211               FOR J:=M0 STEP 1 UNTIL M DO
212               BEGIN   VJ:=V[J];
213                       V[J]:=J+(V[J+1]-VJM1)/2-VJ;
214                       VJM1:=VJ
215               END
216       END;
217       PROCEDURE OUTPUT;
218       BEGIN   ARRAY B,C[G0:G];
219               REAL PROCEDURE NORM(A); ARRAY A;
220               BEGIN REAL S, AJ; S:=0;
221                       FOR J:=-2+PER STEP 1 UNTIL 2+PER DO
222                       BEGIN AJ:=A[J]; S:=AJ+AJ+S END;
223                       NORM:=SQRT(S)
224               END;
225               NLCR; ASSP←WV(3,0,K); ASSP←WV(1,4,T);
226               FOR J:=-3+PER STEP PER UNTIL 3+PER DO
227               P←WV(1,4,U[J]); ANSOL(B);
228               FOR J:=-2+PER STEP 1 UNTIL 2+PER DO C[J]:=U[J]-B[J];
229               EPSABS:=NORM(C); EPS:=EPSABS/NORM(B);
230               ASSS←WV(1,5,EPS); ASSP←WV(1,5,EPSABS);
231               IF AETA>0∧RETA>0 THEN
232               BEGIN   ASSP←WV(1,4,ETA); ASSS←WV(1,6,RHO) END;
233       END;
234
235               K:=I:=0; ANSOL(U);
```

```
236                     AETA:=READ; RETA:=READ;
237                     MODIFIED RUNGE KUTTA(T, IF K>25 THEN T ELSE .5, M0, M,
238                             U, M+1, I, DERIVATIVE, K,
239                                     DATA, 1.5, 2. AETA, RETA,ETA, RHO, OUTPUT);
240             END
241     END
242
```

THE INITIAL VALUE PROBLEM UT=XUX-U, UO=X+1
BY PROCEDURE MODIFIED RUNGE KUTTA

DEGREE= 3 ORDER= 2 H=+.5000E- 2

K	T	X=-.015	-.010	-.005	0	.005	.010	.015	EPS	EPSABS
1	.0198	+.9054	+.9304	+.9554	+.9804	+1.0054	+1.0304	+1.0554	.00000	.00000
2	.0402	+.8856	+.9106	+.9356	+.9606	+.9856	+1.0106	+1.0356	.00000	.00001
3	.0613	+.8656	+.8906	+.9156	+.9406	+.9656	+.9906	+1.0156	.00000	.00001
4	.0830	+.8453	+.8703	+.8953	+.9203	+.9453	+.9703	+.9953	.00000	.00001
5	.1055	+.8249	+.8499	+.8749	+.8999	+.9249	+.9499	+.9749	.00000	.00002
6	.1287	+.8042	+.8292	+.8542	+.8792	+.9042	+.9292	+.9542	.00001	.00002
7	.1528	+.7833	+.8083	+.8333	+.8583	+.8833	+.9083	+.9333	.00001	.00002
8	.1778	+.7621	+.7871	+.8121	+.8371	+.8621	+.8871	+.9121	.00001	.00003
9	.2038	+.7406	+.7656	+.7906	+.8156	+.8406	+.8656	+.8906	.00001	.00003
10	.2308	+.7189	+.7439	+.7689	+.7939	+.8189	+.8439	+.8689	.00001	.00004
11	.2590	+.6968	+.7218	+.7468	+.7718	+.7968	+.8218	+.8468	.00001	.00005
12	.2884	+.6744	+.6994	+.7244	+.7494	+.7744	+.7994	+.8244	.00002	.00005
13	.3192	+.6517	+.6767	+.7017	+.7267	+.7517	+.7767	+.8017	.00002	.00006
14	.3514	+.6287	+.6537	+.6787	+.7037	+.7287	+.7537	+.7787	.00002	.00007
15	.3853	+.6052	+.6302	+.6552	+.6802	+.7052	+.7302	+.7552	.00002	.00007
16	.4210	+.5813	+.6063	+.6313	+.6563	+.6813	+.7063	+.7313	.00003	.00008
17	.4588	+.5570	+.5820	+.6070	+.6320	+.6570	+.6820	+.7070	.00003	.00009
18	.4988	+.5322	+.5572	+.5822	+.6072	+.6322	+.6572	+.6822	.00004	.00011
19	.5000	+.5315	+.5565	+.5815	+.6065	+.6315	+.6565	+.6815	.00004	.00011

2. Beschouw het begin-randwaarde probleem

$$(3.24)\quad \begin{cases} U_t = U_{xx} - U, & -\pi/2 \le x \le \pi/2,\ 0 \le t \le T, \\ U(x,0) = \cos x, & -\pi/2 \le x \le \pi/2, \\ U(-\pi/2,t) = U(\pi/2,t) = 0, & 0 \le t \le T. \end{cases}$$

Het stelsel gewone differentiaalvergelijkingen wordt nu:

$$(3.25)\quad \begin{cases} \dfrac{d\vec{U}}{dt} = \left(\dfrac{E_+ - 2 + E_-}{h^2}\right)\vec{U} + \vec{F}(t), \\ \vec{U}_0 = \cos(j \times h), \end{cases}$$

met

$$(3.26)\quad D = \begin{bmatrix} -2/h^2-1 & 1/h^2 & 0 & \cdots & 0 \\ 1/h^2 & -2/h^2-1 & 1/h^2 & 0 & \vdots \\ 0 & 1/h^2 & -2/h^2-1 & 1/h^2 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ & & 1/h^2 & -2/h^2-1 & 1/h^2 \\ 0 & \cdots & \cdots & 1/h^2 & -2/h^2-1 \end{bmatrix} \quad \text{en} \quad \vec{F}(t) = \begin{bmatrix} 0 \\ \vdots \\ \\ \\ \\ \\ 0 \end{bmatrix}$$

De eigenwaarden δ van D worden hier

$$(3.27)\qquad -1 - 4/h^2 \sin^2 \frac{\omega h}{2}$$

en dus is de spectraalradius

$$(3.28)\qquad \sigma(D) = 1 + 4/h^2.$$

De eigenwaarden zijn negatief reëel en we kiezen nu één van de polynomen uit de bijbehorende klasse, b.v.

$$(3.29)\qquad P_4(x) = 1 + x + 1/2x^2 + 1/16x^3, \qquad \beta_{re}(3) = 6.27.$$

De analytische oplossing $U(x,t) = e^{-2t} \cos x$ wordt nu benaderd tot op een term, die

$$O(\tau^3) + O(\tau h^2)$$

bedraagt. De stabiliteitsconditie stelt

$$\tau_{stab} \leq \frac{6.27}{1+4/h^2} = \frac{6.27}{4+h^2} h^2 = O(h^2).$$

We maken dus op zijn minst een totale benaderingsfout $O(h^4)$. Dit impliceert, dat voor dit geval ook een eerste orde schema gebruikt kan worden, omdat ook dan de afbreekfout $O(\tau^2) = O(h^4)$ van dezelfde grootte orde is. We kunnen b.v. het polynoom

$$(3.30) \qquad P_4(x) = 1 + x + 5/32x^2 + 1/128x^3 + 1/8192x^4, \quad \beta_{re}(4) = 32,$$

nemen, dat een 5× zo grote stap toelaat.

We geven nu de procedure derivative en de actuele aanroep van modified runge kutta:

```
procedure derivate (t,v); real t; array v;
begin integer j; real vjm1,vj;
      vjm1:= v[m0];
      v[m0+1] := (v[m0+2]-2*v[m0+1])/(h*h) - v[m0+1];
      for j:= m0+2 step 1 until m-2 do
      begin vj:= v[j];
            v[j]:= (v[j+1]-2*vj+vjm1)/(h*h) - vj;
            vjm1:= vj
      end;
      v[m-1]:= (vjm1-2*v[m-1])/(h*h) - v[m-1]
end;
```

Kies $h = \pi/100$, dan $-m0 = m = 50$ en de actuele aanroep van modified runge kutta wordt:

```
modified runge kutta (t,if k > 20 then t else .1,-50,50,
        u,1+4/h/h,i,derivative,k, data,1.5,2,10^-5,10^-4,eta,rho,output);
```

Tenslotte geven we nog de resultaten van dit voorbeeld:

THE INITIAL BOUNDARY VALUE PROBLEM UT=UXX-U, U0=COS(X),
U(-PI/2,T)=U(PI/2,T)=0

DEGREE= 4 ORDER= 1 H=+.3142E- 1

K	T	X=-PI/2	-PI/4	0	PI/4	PI/2	EPS	EPSABS	ETA	RHO
1	.0000	+.0000	+.7071	+1.0000	+.7071	+.0000	.0000	.0000	.0001	.000000
2	.0001	+.0000	+.7070	+.9999	+.7070	+.0000	.0000	.0000	.0001	.000000
3	.0006	+.0000	+.7062	+.9987	+.7062	+.0000	.0000	.0000	.0001	.000003
4	.0035	+.0000	+.7021	+.9930	+.7021	+.0000	.0000	.0001	.0001	.000081
5	.0064	+.0000	+.6981	+.9873	+.6981	+.0000	.0000	.0002	.0001	.000080
6	.0093	+.0000	+.6941	+.9816	+.6941	+.0000	.0000	.0002	.0001	.000080
7	.0122	+.0000	+.6901	+.9759	+.6901	+.0000	.0000	.0003	.0001	.000079
8	.0150	+.0000	+.6861	+.9703	+.6861	+.0000	.0001	.0004	.0001	.000079
9	.0179	+.0000	+.6821	+.9647	+.6821	+.0000	.0001	.0005	.0001	.000079
10	.0208	+.0000	+.6782	+.9591	+.6782	+.0000	.0001	.0005	.0001	.000078
11	.0237	+.0000	+.6743	+.9536	+.6743	+.0000	.0001	.0006	.0001	.000078
12	.0266	+.0000	+.6704	+.9481	+.6704	+.0000	.0001	.0007	.0001	.000077
13	.0295	+.0000	+.6665	+.9426	+.6665	+.0000	.0001	.0008	.0001	.000077
14	.0324	+.0000	+.6627	+.9372	+.6627	+.0000	.0001	.0008	.0001	.000077
15	.0353	+.0000	+.6588	+.9317	+.6588	+.0000	.0001	.0009	.0001	.000076
16	.0382	+.0000	+.6550	+.9264	+.6550	+.0000	.0001	.0010	.0001	.000076
17	.0411	+.0000	+.6512	+.9210	+.6512	+.0000	.0002	.0010	.0001	.000076
18	.0440	+.0000	+.6475	+.9157	+.6475	+.0000	.0002	.0011	.0001	.000075
19	.0469	+.0000	+.6437	+.9104	+.6437	+.0000	.0002	.0012	.0001	.000075
20	.0498	+.0000	+.6400	+.9051	+.6400	+.0000	.0002	.0012	.0001	.000074
21	.0527	+.0000	+.6363	+.8998	+.6363	+.0000	.0002	.0013	.0001	.000074

3.8. Opgaven

1. Gegeven is het stelsel p.d.v.n.:

$$\begin{cases} p_x - 2xq_y = 0, \quad q_x - 2xp_y = 0, & (x>0,\ y>0), \\ p(0,y) = 0 \quad (y>0), \quad p(x,0) = x, & q(x,0) = 1 \quad (x>0). \end{cases}$$

a. Toon aan, dat de karakteristieken voldoen aan:

$$y \pm x^2 = \text{constant},$$

$p + q$ is constant langs $y + x^2 = c$ en $p - q$ is constant langs $y - x^2 = c$.

b. Bepaal $p(5,24)$, $q(5,24)$ en $q(0,49)$.

2. Geef het type van de p.d.v.:

$$(1-y)U_{xx} + 2xU_{xy} + (1+y)U_{yy} = H(x,y,U_x,U_y).$$

3. Toon aan, dat de karakteristieken van de p.d.v.

$$yU_{xx} - 2xU_{xy} - yU_{yy} = G(x,y)$$

oplossingen zijn van de vergelijking

$$\frac{dx}{dy} = \frac{x}{y} \pm \left(\frac{x^2}{y^2} + 1\right)^{1/2}$$

en vindt deze oplossingen (door x/y als nieuwe variabele te nemen) in de vorm:

$$\sqrt{x^2+y^2} + x = c_1 \text{ en } \sqrt{x^2+y^2} - x = c_2.$$

Als U en U_y voorgeschreven zijn langs de x-as schets dan het afhankelijkheidsgebied van het punt (3,4).

4. Gegeven is de p.d.v.

$$u_x + 2xu_y = x + y$$

met beginvoorwaarde $u(0,y) = 0$.

Bepaal met behulp van de karakteristieken de waarde van U in het punt $(2,1)$.

5. Beschouw het beginwaardeprobleem:

$$U_t = \lambda U_x, \quad \lambda = \text{constant}, \quad t > 0, \quad -\infty < x < \infty,$$

$$U(x,0) = f(x), \quad -\infty < x < \infty.$$

Schrijf dit probleem als een stelsel gewone differentiaalvergelijkingen van de vorm:

$$\frac{d\vec{u}}{dt} = D\vec{u} + \vec{F}(t),$$

zo, dat a) $\lambda \frac{\partial}{\partial x} \equiv D + O(h^2)$,

b) $\lambda \frac{\partial}{\partial x} \equiv D + O(h^3)$.

Bepaal in beide gevallen de spectraalradius van de matrix D en vergelijk voor zekere λ en $f(x)$ de analytische oplossing van dit probleem met de numeriek oplossing, verkregen m.b.v. de in dit hoofdstuk beschreven procedure.

6. Als in opgave 5 voor de p.d.v.:

$$u_t = k_1 u + k_2 u_x + k_3, \quad k_1, k_2, k_3 \text{ constant},$$

$$u(x,0) = f(x).$$

Kies stabiliteitspolynomen voor verschillende waarden van k_1 en k_2.

7. Beschouw het 1-dimensionale golfprobleem:

$$U_{tt} = c^2 U_{xx}, \qquad t > 0, \quad -\infty < x < \infty,$$

$$U(x,0) = f(x), \qquad -\infty < x < \infty,$$

$$U_t(x,0) = g(x).$$

a) Schrijf dit probleem als stelsel 1^e orde p.d.v.n. met behulp van de substitutie: $v = u_t$ en $w = u_x$ en pas de beginvoorwaarden aan.

b) Maak er vervolgens een stelsel gewone differentiaalvergelijkingen van en bepaal de spectraalradius van de matrix D.
Los het verkregen stelsel numeriek op m.b.v. de in dit hoofdstuk beschreven procedure.

c) Is er een substitutie mogelijk, waarbij een stelsel 1^e orde p.d.v.n. ontstaat en U toch direct opgelost wordt? Zo ja, behandel dit geval als a) en b).

3.9. Literatuur

BEENTJES, P.A. *Een ALGOL 60 versie van gestabiliseerde Runga-Kutta methoden*, NR 23, Mathematisch Centrum, Amsterdam. 1972.

COURANT, R. & HILBERT, D. *Methods of mathematecal physics, volume* II, John Wiley-Interscience, New York. 1962.

HOUWEN, P.J. van der, *One step methods for linear initial value problems* I, *polynomial methods*, Rapport TW 119, Mathematisch Centrum, Amsterdam. 1970.

HOUWEN, P.J. van der, BEENTJES, P., DEKKER, K. & SLAGT, E. *One step methods for linear initial value problems* III, Rapport TW 130, Mathematisch Centrum, Amsterdam. 1971.

MATHEMATICAL CENTRE SYLLABUS 20, 1975, IV, 79-128.

METHODE VAN RICHARDSON VOOR ELLIPTISCHE DIFFERENTIAALVERGELIJKINGEN

T.M.T. COOLEN

4.1. Randwaardeproblemen voor elliptische partiële differentiaalvergelijkingen

4.1.1. Definities en afspraken. In paragraaf 3.3 werd reeds de definitie gegeven van een tweede orde elliptische partiële differentiaalvergelijking in twee variabelen. We geven de definitie hier opnieuw. De vergelijking

$$a_{20}(x,y)u_{xx} + a_{11}(x,y)u_{xy} + a_{02}(x,y)u_{yy} = f(x,y,u,u_x,u_y) \tag{4.1}$$

heet *uniform elliptisch* in een enkelvoudig samenhangend open gebied Ω van het platte vlak $\mathbb{R}^2$ als voor alle $x \in \Omega$ geldt

$$a_{11}(x,y)^2 - 4a_{20}(x,y)a_{02}(x,y) \leq \text{const.} < 0. \tag{4.2}$$

We zullen ons in dit hoofdstuk beperken tot vergelijkingen in twee veranderlijken. Van het gebied Ω zullen we steeds aannemen dat zijn rand, die we met $\partial\Omega$ noteren, voldoende glad is, d.w.z. voldoende vaak differentieerbaar. Verder is $\overline{\Omega} = \Omega \cup \partial\Omega$. Een *lineaire* tweede orde elliptische differentiaalvergelijking heeft de gedaante

$$Lu \equiv a_{20}u_{xx} + a_{11}u_{xy} + a_{02}u_{yy} + a_{10}u_x + a_{01}u_y + a_{00}u = f, \tag{4.3}$$

waarin alle coëfficiënten van x en y mogen afhangen. Het deel met de hoogste afgeleiden, de eerste drie termen van (4.3) dus, heet het *hoofddeel* van de elliptische operator L.

Voor de volledigheid geven we nog de definitie van een hogere orde elliptische vergelijking. Als we ons beperken tot reële coëfficiënten van zo'n vergelijking, dan bestaan er slechts elliptische vergelijkingen van even orde. In onze schrijfwijze maken we daarvan gebruik.

Een differentiaalvergelijking van de orde 2m

$$Lu \equiv \sum_{j+l\leq 2m} a_{jl}(x,y) \left(\frac{\partial}{\partial x}\right)^j \left(\frac{\partial}{\partial y}\right)^l u = f \tag{4.4}$$

heet uniform elliptisch in Ω
als geldt dat de uitdrukking

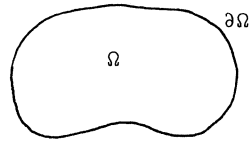


fig. 4.1

$$(4.5) \qquad Q(x,y,\xi,\eta) = \sum_{j+l=2m} a_{jl}(x,y)\xi^j\eta^l$$

voor elke $(x,y) \in \Omega$ *definiet* is.

4.1.2. Voorbeelden. De vergelijking van *Laplace*, ook wel potentiaalvergelijking genaamd,

$$\Delta u \equiv \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 ,$$

is elliptisch in elk gebied van het x - y - vlak. Een oplossing van deze vergelijking heet een *harmonische* functie. Een voorbeeld van een hogere orde elliptische vergelijking is de *biharmonische* vergelijking

$$\Delta\Delta u = \frac{\partial^4 u}{\partial x^4} + 2\frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{\partial^4 u}{\partial y^4} = 0.$$

4.1.3. Stelling. Het hoofddeel van een tweede orde elliptische differentiaalvergelijking in twee variabelen met analytische coëfficiënten is via een geschikte transformatie te schrijven als de operator van Laplace Δ.
Bewijs: zie Coolen, Förch e.a. [1969], p. 5 e.v.
Een consequentie van deze stelling is dat het veelal voldoende is vergelijkingen van de vorm

$$(4.6) \qquad \Delta u + a_{10}u_x + a_{01}u_y + a_{00}u = f$$

te bekijken, als we iets van het karakter van tweede orde elliptische differentiaalvergelijkingen willen leren kennen; (4.6) heet de *canonieke vorm* van elliptische differentiaalvergelijkingen van de tweede orde in twee variabelen.

4.1.4. Zelfgeadjungeerde operator. Een elliptische operator heet *zelfgeadjungeerd* in Ω, als voor voldoend gladde functies v en w de uitdrukking

$$(4.7) \qquad \iint_{\Omega} (wLv-vLw)dxdy$$

alleen afhangt van de waarden van v en w en hun afgeleiden *op de rand* $\partial\Omega$. Een eenvoudige berekening laat zien, dat de operator L gedefinieerd door (4.3) zelfgeadjungeerd is, als

$$a_{10} = \frac{\partial}{\partial x} a_{20} + \frac{1}{2} \frac{\partial}{\partial y} a_{11} \quad \text{en} \quad a_{01} = \frac{\partial}{\partial y} a_{02} + \frac{1}{2} \frac{\partial}{\partial x} a_{11}.$$

Het begrip zelfgeadjungeerd komen we ook tegen in de theorie van de eindige vectorruimten. Elke zelfgeadjungeerde operator in een eindige vectorruimte kan worden voorgesteld door een symmetrische matrix.

4.1.5. Voorbeeld. De operator van Laplace is zelfgeadjungeerd. Dit volgt uit de bekende *formule van Green*

$$(4.8) \qquad \iint_{\Omega} (u\Delta v - v\Delta u)dxdy = -\int_{\partial\Omega} (u \frac{\partial v}{\partial \nu} - v \frac{\partial u}{\partial \nu})ds,$$

waarin $\partial/\partial\nu$ de afgeleide langs de naar binnen gerichte normaal op de rand voorstelt, en ds een lijnelement langs de rand $\partial\Omega$.

4.1.6. Randwaardeproblemen. Er zijn vele oplossingen van een elliptische vergelijking. Maar we zijn altijd slechts geïnteresseerd in een oplossing die voldoet aan bepaalde gegeven voorwaarden op de rand van het gebied Ω. De drie belangrijkste voorwaarden voor vergelijking (4.3) zijn:

$$(4.9) \qquad u = g \qquad \text{op } \partial\Omega,$$

$$(4.10) \qquad b_1 \frac{\partial u}{\partial \nu} + b_2 \frac{\partial u}{\partial s} = g \qquad \text{op } \partial\Omega,$$

$$(4.11) \qquad b_0 + b_1 \frac{\partial u}{\partial \nu} + b_2 \frac{\partial u}{\partial s} = g \qquad \text{op } \partial\Omega,$$

waarin $\partial/\partial s$ de afgeleide langs de raaklijn aan de rand $\partial\Omega$ voorstelt. Men kan laten zien (zie bijvoorbeeld Forsythe & Wasow [1960], p.161), dat voor de vergelijking van Laplace deze drie voorwaarden overgaan in

(4.9') $\qquad u = g \qquad$ op $\partial\Omega$,

(4.10') $\qquad \frac{\partial u}{\partial \nu} = g \qquad$ op $\partial\Omega$,

(4.11') $\qquad \alpha u + \frac{\partial u}{\partial \nu} = g \qquad$ op $\partial\Omega$.

Deze drie randvoorwaarden dragen resp. de namen *Dirichlet voorwaarde*, *Neumann voorwaarde* en *randvoorwaarde van de derde soort*.

Over de existentie en de uniciteit van de oplossing van een elliptisch randwaardeprobleem bestaat een uitgebreide literatuur. Zie bijvoorbeeld de verslagen van het Colloquium Elliptische Differentiaalvergelijkingen van het Mathematisch Centrum, 2 delen, resp. Coolen, Förch, e.a. [1969] en Van den Brink, Coolen, e.a. [1970], waarin uitvoerige literatuurlijsten zijn opgenomen. Als voorbeeld noemen we de volgende stelling.

4.1.7. Stelling. Het Dirichletprobleem

$$\Delta u = f \qquad \text{in } \Omega,$$
$$u = g \qquad \text{op } \partial\Omega,$$
$$u \text{ continu} \qquad \text{in } \overline{\Omega} = \Omega \cup \partial\Omega.$$

en het derde randwaardeprobleem

$$\Delta u = f \qquad \text{in } \Omega,$$
$$\alpha u + \frac{\partial u}{\partial \nu} = g \qquad \text{op } \partial\Omega, \quad \alpha \neq 0,$$
$$u \text{ continu} \qquad \text{in } \overline{\Omega},$$

zijn eenduidig oplosbaar. Het Neumannprobleem

$$\Delta u = f \qquad \text{in } \Omega,$$
$$\frac{\partial u}{\partial \nu} = g \qquad \text{in } \partial\Omega,$$
$$u \text{ continu} \qquad \text{in } \overline{\Omega},$$

is dat op een constante na ook onder de additionele voorwaarde

(4.12) $$\int_{\partial\Omega} g\,ds = \iint_{\Omega} f\,dxdy.$$

Bewijs. Zie bijv. Coolen, Förch e.a.[1969], hoofdstukken 2 en 3. □

4.1.8. Opmerking. Het komt er op neer, dat men voor eenduidige oplosbaarheid van een randwaardeprobleem niet te veel en niet te weinig voorwaarden op de rand moet geven. Voor tweede orde elliptische vergelijkingen moet men kennelijk overal op de rand één voorwaarde geven. Voor vergelijkingen van de orde 2m zijn precies m voorwaarden overal op de rand nodig. Als voorbeeld weer de biharmonische vergelijking

$$\begin{aligned} &\Delta\Delta u = 0 && \text{in } \Omega, \\ &u = g,\ \frac{\partial u}{\partial \nu} = h && \text{op } \partial\Omega, \\ &u \text{ continu} && \text{in } \overline{\Omega}. \end{aligned}$$

Een van de meest bruikbare analytische hulpmiddelen bij het bestuderen van elliptische differentiaalvergelijkingen is het *maximumprincipe*. We zullen dit principe formuleren voor de vergelijking

$$Lu = \Delta u + a_{10}u_x + a_{01}u_y + a_{00}u = 0, \tag{4.13}$$

die de canonieke vorm van elliptische vergelijkingen van de tweede orde is.

4.1.9. Stelling. Als $a_{00} \leq 0$ dan kan een niet-constante oplossing van $Lu = 0$ niet een positief maximum of een negatief minimum aannemen binnen Ω.
Bewijs. Zie Forsythe & Wasow [1960], p.174.

4.1.10. Stelling. Een niet-constante oplossing van $\Delta u = 0$ kan zijn maximum of minimum slechts op de rand van het gebied Ω
Bewijs. In tegenstelling tot wat wij gedaan hebben bij de vorige stellingen, geven we het bewijs hier wel, omdat daarmee nog een belangrijke eigenschap van harmonische functies naar voren komt, de zogenaamde *middelwaardeeigenschap*. Uit de functietheorie is bekend dat elke functie $u(x,y)$ het reële deel is van een analytische complexwaardige functie $w(z)$, $z = x + iy$. Volgens de integraalstelling van Cauchy is dan

$$w(z) = \frac{1}{2\pi i} \int_C \frac{w(\zeta)d\zeta}{\zeta - z}, \tag{4.14}$$

waarin C een cirkel met straal r en middelpunt z is, die geheel binnen Ω

ligt. Daar $\zeta = z + re^{i\theta}$, kunnen we voor (4.14) ook schrijven

$$w(z) = \frac{1}{2\pi}\int_0^{2\pi} w(z+re^{i\theta})d\theta.$$

Door hiervan het reële deel te nemen vinden we

$$u(x,y) = \frac{1}{2\pi}\int_0^{2\pi} u(x+r\cos\theta,\ y+r\sin\theta)d\theta, \tag{4.15}$$

hetgeen betekent dat u(x,y) het gemiddelde is van zijn waarden op elke cirkel rond (x,y). De stelling volgt nu door (4.15) toe te passen in elk punt (x,y) waar u een maximum of minimum aanneemt zonder locaal constant te zijn. □

4.2. Discretisatie van elliptische randwaardeproblemen

De numerieke behandeling van elliptische randwaardeproblemen verschilt nogal van die van hyperbolische en parabolische begin(randwaarde-) problemen. Daar kan men de partiële differentiaalvergelijking met beginvoorwaarde(n) omzetten in een stelsel gewone differentiaalvergelijkingen met beginvoorwaarde(n). De onafhankelijke variabele was de tijd t.
Hier hebben we geen tijdachtige variabelen; de methoden beschreven bij de parabolische en hyperbolische vergelijkingen lukken dan ook niet, te meer omdat hier ook rondom randvoorwaarden worden gesteld.

4.2.1. Keuze van het net. We discretiseren elliptische problemen naar beide variabelen tegelijk. Bij de methoden van het eindige differentietype - andere zullen we niet bekijken - wordt het samenhangende gebied Ω, met onafhankelijke variabele $(x,y) \in \Omega$, waar men de functie u zoekt, vervangen door een eindige verzameling Ω_h van punten.

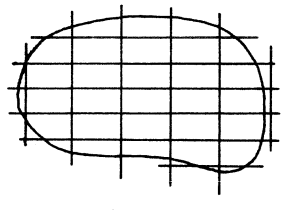


fig. 4.2

Deze punten worden meestal gekozen als de roosterpunten van een rechthoekig net (zie fig. 4.2). Ook wij zullen ons hiertoe beperken.
We gebruiken nu een andere notatie dan die uit de vorige paragraaf. In het vervolg zullen we alle continu gegeven of te bepalen functies aangeven met hoofdletters, en alle functies die slechts waarden hebben in de roosterpunten met kleine letters. Elke afhankelijke functie U(x,y), gedefinieerd

op Ω, wordt vervangen door een roosterfunctie u(x,y), gedefinieerd op Ω_h. Het probleem om uit het elliptische randwaardeprobleem U(x,y) te bepalen wordt op deze wijze vervangen door het probleem een eindig stelsel simultane algebraische vergelijkingen op te lossen, waarin de waarden van u(x,y), $(x,y) \in \Omega_h$ als onbekenden voorkomen. Dit proces heet *discretisatie*. Op welke wijze dat stelsel algebraische vergelijkingen wordt opgelost, is een van de discretisatie onafhankelijk probleem, dat we in paragraaf 4.4 behandelen.

4.2.2. Enige probleem bij discretisatie. (i) Op welke wijze moet de rand worden gerepresenteerd als deze niet bestaat uit zijden van de rechthoeken van het rooster?
(ii) Wat zijn goede benaderingen L_h van de elliptische differentiaaloperator L, en hoe moeten we de normale afgeleide $\partial U/\partial \nu$ op de rand representeren? Het eindige differentie-analogon van het randwaardeprobleem moet met dit randwaardeprobleem *consistent* zijn, d.w.z. als de maaswijdte van het net tot 0 nadert, dan moet het differentieprobleem naderen tot het differentiaalprobleem. Dus $L_h u = f$ moet overgaan in $LU = F$.
(iii) Wat is het verschil tussen u(x,y) en U(x,y) voor punten $(x,y) \in \Omega_h$? De grootheid $\| u(x,y) - U(x,y) \|$ heet de *discretisatiefout*, waarbij $\| \cdot \|$ een of andere norm voorstelt. Deze moet met de maaswijdte tot 0 naderen.

We zullen alleen maar *regelmatige, rechthoekige* netten bekijken. Dit betekend dat de roosterafstand ξ in de x-richting, en de roosterafstand η in de y-richting niet veranderen. Voor onregelmatige en niet-rechthoekige netten, zie bijv. Forsythe & Wasow [1960]. Meestal zullen we de punten nummeren met twee gehele getallen j en l, en zullen we j van links naar rechts laten lopen, en l van onderen naar boven.

4.2.3. Discretisatie van elliptische operatoren. Hoe discretiseren we de operator

$$LU = A_{20}U_{xx} + A_{11}U_{xy} + A_{02}U_{yy} + A_{10}U_x + A_{01}U_y + A_{00}U \quad ? \tag{4.16}$$

Voor een compacte notatie introduceren we de zogenaamde *molecuul*notatie. Laat (x,y) een punt van Ω_h zijn, dan behoren, daar we ons tot regelmatige roosters beperken, de punten $(x \pm \xi, y \pm \eta)$ eveneens tot Ω_h. We schrijven

nu voor

$$
\begin{aligned}
&au(x-\xi,y-\eta) + bu(x,y-\eta)\\
&+ cu(x+\xi,y-\eta) + du(x-\xi,y)\\
&+ eu(x,y) + fu(x+\xi,y)\\
&+ gu(x+\xi,y+\eta) + hu(x,y+\eta)\\
&+ iu(x+\xi,y+\eta)
\end{aligned}
\qquad
\begin{bmatrix} g & h & i \\ d & e & f \\ a & b & c \end{bmatrix} u\,, \tag{4.17}
$$

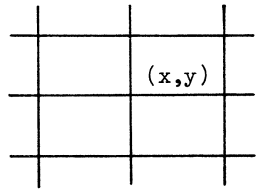


fig. 4.3

waarbij we eventueel een rij of kolom zullen weglaten. Vervang nu

$$
\begin{aligned}
U_{xx} \quad &\text{door} \quad \xi^{-2}\,[1\ -2\ 1]\,u,\\
U_{yy} \quad &\text{door} \quad \rho^2\xi^{-2}\begin{bmatrix}1\\-2\\1\end{bmatrix}u, \text{ waarin } \rho = \xi/\eta,\\
U_{xy} \quad &\text{door} \quad (2\xi)^{-2}\begin{bmatrix}-\rho & 0 & \rho\\ 0 & 0 & 0\\ \rho & 0 & -\rho\end{bmatrix}u,\\
U_x \quad &\text{door} \quad (2\xi)^{-1}\,[1\ 0\ -1]\,u,\\
U_y \quad &\text{door} \quad \rho(2\xi)^{-1}\begin{bmatrix}1\\0\\-1\end{bmatrix}u.
\end{aligned}
\tag{4.18}
$$

Hierbij zien we af van eventuele moeilijkheden in de buurt van de rand. Als we al deze uitdrukkingen combineren, dan wordt de met L corresponderende differentieoperator L_h:

$$
L_h u = \xi^{-2}\begin{bmatrix} -\frac{1}{4}\rho A_{11} & \rho^2 A_{02} + \frac{1}{2}\rho\xi A_{01} & \frac{1}{4}\rho A_{11} \\ A_{20} - \frac{1}{2}\xi A_{10} & -2(A_{20}+\rho^2 A_{02}) + \xi^2 A_{00} & A_{20} + \frac{1}{2}\xi A_{10} \\ \frac{1}{4}\rho A_{11} & \rho^2 A_{02} - \frac{1}{2}\rho\xi A_{01} & -\frac{1}{4}\rho A_{11} \end{bmatrix} u, \tag{4.19}
$$

waarin alle functies A_{jl} in het punt (x,y), het middelste punt van het negental, worden geëvalueerd.

Deze keuze van differentieoperatoren wordt gemotiveerd voor de bekende Taylorenontwikkeling met restterm. We laten dit zien voor het differentie-analogen van U_{xx}:

tie-analogon van U_{xx}:

$$U(x+\xi,y) = U(x,y) + \xi U_x(x,y) + \tfrac{1}{2}\xi^2 U_{xx}(x,y) +$$
$$+ \frac{1}{6}\xi^3 U_{xxx}(x,y) + \frac{1}{24}\xi^4 U_{xxxx}(x+\theta_1\xi,y),$$

en
$$U(x-\xi,y) = U(x,y) - \xi U_x(x,y) + \tfrac{1}{2}\xi^2 U_{xx}(x,y) +$$
$$- \frac{1}{6}\xi^3 U_{xxx}(x,y) + \frac{1}{24}\xi^4 U_{xxxx}(x+\theta_2\xi,y),$$

waarin $0 < \theta_1,\theta_2 < 1$.

Optellen en herrangschikken van beide gelijkheden levert

$$\xi^{-2}(U(x+\xi,y) - 2U(x,y) + U(x-\xi,y)) =$$
$$= U_{xx}(x,y) + \frac{1}{24}\xi^2(U_{xxxx}(x+\theta_1\xi,y) + U_{xxxx}(x+\theta_2\xi,y)).$$

Onder de aanname dat de vierde partiële afgeleide naar x van U begrensd is op $[x-\xi,x+\xi]$ volgt dan

$$\xi^{-2}[1\ -2\ 1]U(x,y) - U_{xx}(x,y) = O(\xi^2) \text{ als } \xi \to 0.$$

Op eenvoudige wijze is nu de volgende stelling te bewijzen.

4.2.4. Stelling. Wanneer alle vierde partiële afgeleiden van $U(x,y)$ begrensd zijn op Ω, dan geldt op Ω voor $\xi \to 0$ en $\xi/\eta = O(1)$

$$L_h U(x,y) - LU(x,y) = O(\xi^2). \tag{4.20}$$

We bekijken nu nog eens apart de discretisatie van de operator van Laplace.

4.2.5. Stelling. Laat Δ_h gedefinieerd zijn door

$$\Delta_h U = \begin{bmatrix} L_1 & L_3 & L_1 \\ L_2 & L_4 & L_2 \\ L_1 & L_3 & L_1 \end{bmatrix} U, \tag{4.21}$$

waarin

$$(4.22)\quad \begin{cases} L_1 = \tfrac{1}{2}\xi^{-2}(1+\rho^2-\gamma),\ L_2 = \xi^{-2}(\gamma-\rho^2), \\ L_3 = \xi^{-2}(\gamma-1),\ L_4 = -2\gamma\xi^{-2}, \\ \rho = \xi/\eta,\ \gamma \text{ nog een te bepalen reële parameter is.} \end{cases}$$

Dan geldt dat voor $\xi \to 0$ en $\xi/\eta = O(1)$

$$\begin{aligned} &\Delta_h U(x,y) - \Delta U(x,y) = \\ &= \frac{1}{12}\xi^2\left(\frac{\partial^4 U}{\partial x^4} + 12L_1\xi^2\rho^{-2}\frac{\partial^4 U}{\partial x^2\partial y^2} + \rho^{-2}\frac{\partial^4 U}{\partial y^4}\right) + \\ &+ \frac{1}{360}\xi^4\left(\frac{\partial^6 U}{\partial x^6} + 30L_1\xi^2\left(\rho^{-2}\frac{\partial^6 U}{\partial x^4\partial y^2} + \rho^{-4}\frac{\partial^6 U}{\partial x^2\partial y^4}\right) + \rho^{-4}\frac{\partial^6 U}{\partial y^6}\right) \\ &+ \frac{1}{20160}\xi^6\left(\frac{\partial^8 U}{\partial x^8} + 14L_1\xi^2\left(4\rho^{-2}\frac{\partial^8 U}{\partial x^6\partial y^2} + 10\rho^{-4}\frac{\partial^8 U}{\partial x^4\partial y^4} + \right.\right. \\ &\left.\left. + 4\rho^{-6}\frac{\partial^8 U}{\partial x^2\partial y^6}\right) + \rho^{-6}\frac{\partial^8 U}{\partial y^8}\right) + O(\xi^8). \end{aligned} \tag{4.23}$$

Bewijs. Door $\Delta_h U$ uit te schrijven in termen van $U(x \pm \xi, y \pm \eta)$, deze in Taylorreeksen t.o.v. het punt (x,y) te ontwikkelen wordt (4.20) verkregen.□

4.2.6. Hogere orde approximatie van de potentiaalvergelijking. In het algemeen volgt uit de vorige stelling dat voor $(x,y) \in \Omega$

$$\Delta_h U - \Delta U = O(\xi^2),\ \xi \to 0,\ \rho = O(1),$$

als maar de afgeleiden tot en met de vierde orde begrensd zijn op Ω. Wanneer we ons echter beperken tot harmonische functies, dus $\Delta U = 0$, dan kunnen we hogere orde approximaties vinden door gebruik te maken van

$$\partial^2 U/\partial y^2 = -\partial^2 U/\partial y^2,$$

formule (4.23) gaat dan over in

$$\begin{aligned} \Delta_h U - \Delta U &= \frac{1}{12}\xi^2\rho^{-2}(\rho^2-12L_1\xi^2+1)\frac{\partial^4 U}{\partial x^4} + \\ &+ \frac{1}{360}\xi^4\rho^{-2}(\rho^2-1)(\rho^2+1-30L_1\xi^2)\frac{\partial^6 U}{\partial x^6} + \\ &+ \frac{1}{20160}\xi^6\rho^{-6}(\rho^6+1-28L_1\xi^2\rho^4(2\rho^4-5\rho^2+2))\frac{\partial^8 U}{\partial x^8} + O(\xi^8). \end{aligned} \tag{4.24}$$

We zullen nu (4.21) en (4.24) bekijken voor verschillende waarden van γ. Voor $\gamma = 1 + \rho^2$ is $L_1 = 0$, en krijgen we de bekende *vijfpunts + formule*

$$(4.25) \qquad \Delta_h^+ U = \begin{bmatrix} 0 & L_2 & 0 \\ L_3 & L_4 & L_3 \\ 0 & L_2 & 0 \end{bmatrix} U,$$

waarvoor geldt

$$(4.26) \qquad \Delta_h^+ U - \Delta U = O(\xi^2) \text{ voor } \xi \to 0, \ \rho = O(1)$$

Als we bovendien $\rho = 1$ kiezen, krijgen we het overbekende molecuul

$$(4.27) \qquad \Delta_h^+ U = \xi^{-2} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} U$$

Kiezen we $\gamma = \frac{5}{6}(1+\rho^2)$, dan verdwijnt de eerste term in het rechterlid van (4.24), en krijgen we een approximatie voor Δ met een foutterm van $O(\xi^4)$. Als we, additioneel, $\rho = 1$, d.w.z. $\xi = \eta = h$ kiezen, krijgen we zelfs een benadering voor Δ met fout van de orde ξ^6. Uit (4.24) volgt dan, dat

$$(4.28) \qquad \Delta_h U - \Delta U = \frac{1}{3024} \xi^6 \frac{\partial^8 U}{\partial x^8} + O(\xi^8), \ \xi \to 0.$$

Deze keuze van ρ en γ geeft een formule die bekend staat als de *negenpunts Laplace differentieformule*

$$(4.28) \qquad \Delta_h^{(9)} U = \frac{1}{6} \xi^{-2} \begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix} U.$$

4.2.7. Opmerking. Als we de moleculen, die de Laplace vergelijking simuleren, bekijken zien we dat steeds de waarde van de roosterfunctie in het middelste punt van het negental gelijk is aan een gewogen gemiddelde van

de andere acht. Dit correspondeert met de middelwaarde-eigenschap (stelling 4.1.10).

4.2.8. Belangrijke opmerking. In het voorgaande hebben we gesproken over de orde van *consistentie*, d.w.z. over de mate waarin de differentieoperator op de differentiaaloperator lijkt. Dit betekent nog niet zonder meer dat de roosterfuncties u naar de analytische oplossing U *convergeren* voor $\xi \to 0$, $\xi/\eta = O(1)$. Op deze problematiek zullen we hier niet ingaan; men zij verwezen naar Forsythe & Wasow [1960], p.283 e.v.. Steeds zullen we ervan uitgaan dat er convergentie is voor $\xi \to 0$.

4.2.9. Discretisatie van Dirichletrandvoorwaarden. Tot nog toe hebben we alleen gesproken over de discretisatie van de elliptische operator in het inwendige van Ω. Richten wij nu onze aandacht op de rand. Als de rand bestaat uit een aaneenschakeling van verbindingslijnstukken tussen de roosterpunten, dan is discretisatie van de Dirichletrandvoorwaarden eenvoudig: geef u(x,y) de voorgeschreven waarde op de rand.

Wanneer de rand niet zo "mooi" is, dan moet men interpoleren. Als voorbeeld geven we een *lineaire* approximatie (zie figuur 4.4):

$$(4.29) \qquad \begin{aligned} U(x+(1-s)\xi,y) &= sU(x,y) + \\ &+ (1-s)U(x+\xi,y) = \\ &= G(x+(1-s)\xi,y), \end{aligned}$$

waarin G de Dirichletrandvoorwaarde in het continue probleem is. Als differentievergelijking in de buurt van de rand moeten we dan opnemen

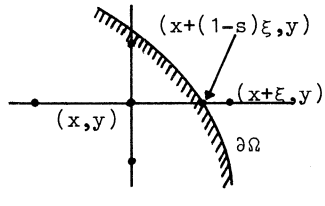


fig. 4.4

$$(4.30) \qquad U(x+\xi,y) = \frac{G(x+(1-s)\xi,y) - sU(x,y)}{1-s} .$$

Hogere orde interpolatie is mogelijk. Zie daarvoor Forsythe & Wasow [1960], p.198 e.v., Hildebrand [1968], p.266 e.v..

4.2.10. Discretisatie van de Neumannrandvoorwaarde. We behandelen slechts de situatie dat de rand is opgebouwd uit lijnstukjes die de roosterpunten verbinden. Taylorontwikkeling geeft (zie figuur 4.5):

$$U(x,y+\eta) = U(x,y) +$$

$$+ \eta U_y(x,y) + \eta^2 U_y(x,y+\theta\eta),$$

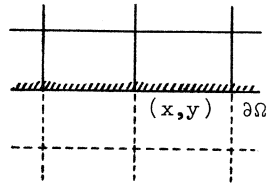


fig. 4.5

waarin $0 < \theta < 1$. Hieruit volgt dan dat de normale afgeleide te benaderen is door

$$(4.31) \qquad U_y(x,y) = \frac{U(x,y+\eta) - U(x,y)}{\eta} + O(\eta).$$

Het kan echter op een eenvoudige manier een orde nauwkeuriger. Breid het rooster met een regel naar onderen uit (zie figuur 4.5, stippellijn). Dan is met behulp van Taylorontwikkelingen t.o.v. (x,y) aan te tonen dat

$$U_y(x,y) = \frac{U(x,y+\eta) - U(x,y-\eta)}{2\eta} + O(\eta^2) .$$

Op deze wijze hebben we echter een aantal extra onbekenden $U(x,y-\eta)$, $(x,y) \in \partial\Omega$, geïntroduceerd. Dit vangen we op door aanvullend ook voor de randpunten de differentievergelijking die in het inwendige van Ω geldt, te introduceren. Voor de Neumannrandvoorwaarde $\frac{\partial U}{\partial \nu} = 0$ op $\partial\Omega$ komt dit overeen met het voortzetten van de analytische oplossing over de rand heen, zodanig dat

$$U(x,y-\eta) = U(x,y+\eta)$$

voor alle η.

Voor minder mooie randen zij verwezen naar de literatuur Forsythe & Wasow [1960] en Hildebrand [1968], resp. p.198 en 266.

4.2.11. Discretisatie van de operator L in de buurt van een willekeurige rand. Men zij verwezen naar bovengenoemde literatuur.

4.3. Het stelsel differentievergelijkingen gezien als matrixvergelijking

4.3.1. Andere schrijfwijze van de differentievergelijkingen. Laten we aannemen dat een elliptische randwaardeprobleem gediscretiseerd is volgens de methoden van de vorige paragraaf. Laten we de verzameling van die punten van de rand $\partial\Omega$, waarin de roosterfunctie u moet worden bepaald, aangeven

met $\partial\Omega_h$, en $\overline{\Omega}_h = \Omega_h \cup \partial\Omega_h$. In plaats van de notatie van de vorige paragraaf te gebruiken, kunnen we voor elk punt $(x,y) = P \in \overline{\Omega}_h$ de bijbehorende differentievergelijking ook schrijven als

$$(4.32) \qquad L_h u(P) \equiv \sum_{Q \in \overline{\Omega}_h} a(P,Q)u(Q) = z(P) ,$$

waarin de a(P,Q) getallen zijn die afhangen van ξ en η, en van de gebruikte discretisatie van de differentiaaloperator of van de randvoorwaarden. Merk op dat in (4.32) de randvoorwaarden mede zijn opgenomen. Laat N het aantal punten $P \in \overline{\Omega}_h$ aangeven; het is vaak handig de waarden van de roosterfunctie u(P) als een vector $\vec{u}$ met N componenten te beschouwen. De lineaire operator L_h wordt dan gepresenteerd door een vierkante matrix A van orde N; i.p.v. $L_h u(P) = z(P)$ schrijven we $A\vec{u} = \vec{z}$. Aan de hand van een voorbeeld zullen we een en ander toelichten.

4.3.2. Voorbeeld. Beschouw een rechthoek met zijden 4h en 3h (zie figuur 4.6), waaroverheen een rooster is gelegd, zoals in de figuur is aangegeven. De roosterafstanden ξ en η worden beide gelijk aan h gekozen. In dit vierkant bekijken we het Dirichletrandwaardeprobleem

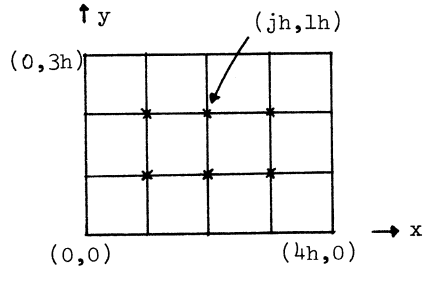


fig. 4.6

$$(4.33) \qquad \Delta U = F \text{ in } \Omega,$$
$$U = G \text{ op } \partial\Omega,$$

dat we discretiseren m.b.v. de vijfpunts + formule (4.27). Het gediscretiseerde probleem ziet er dan uit als

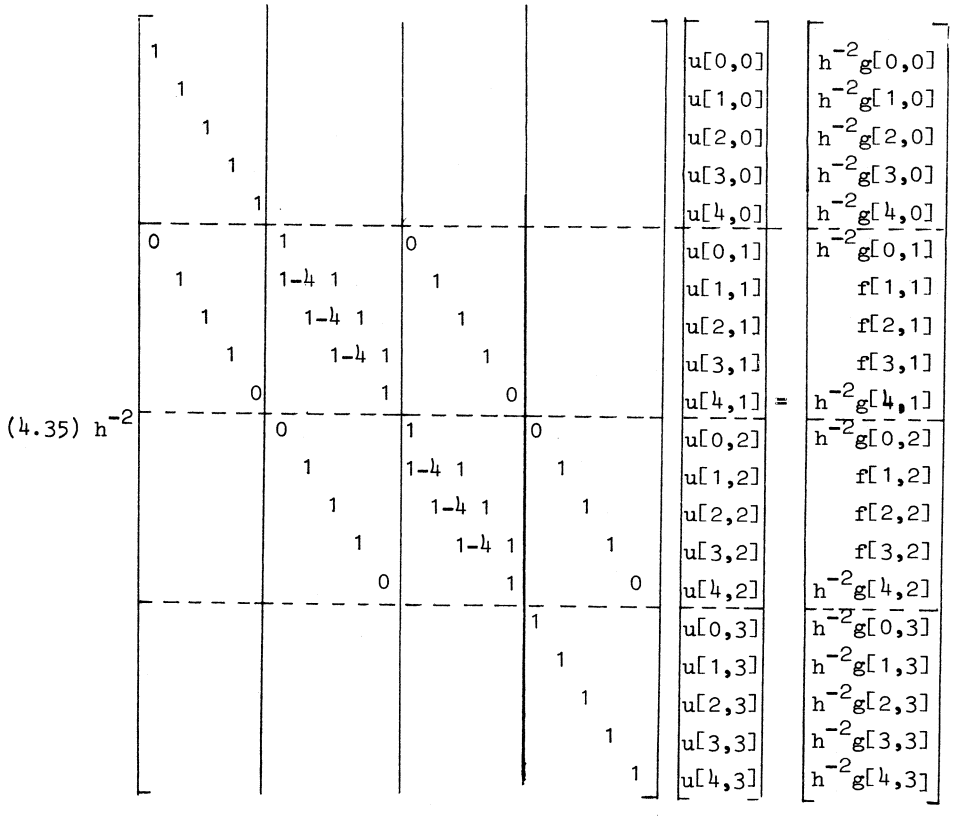
$$(4.34) \qquad \Delta_h^+ u = f \text{ in } \Omega_h,$$
$$u = g \text{ op } \partial\Omega_h,$$

waarin $\Omega_h = \{(jh,lh) \mid 1 \le j \le 3, \quad 1 \le l \le 2\}$,

$\overline{\Omega}_h = \{(jh,lh) \mid 0 \le j \le 4, \quad 0 \le l \le 3\}$,

en $\partial\Omega_h = \bar{\Omega}_h \setminus \Omega_h$.

Schrijven wij nu (4.34) in de vorm van een matrixvergelijking:



(4.35) h^{-2}

1	u[0,0]	$h^{-2}g[0,0]$
1	u[1,0]	$h^{-2}g[1,0]$
1	u[2,0]	$h^{-2}g[2,0]$
1	u[3,0]	$h^{-2}g[3,0]$
1	u[4,0]	$h^{-2}g[4,0]$
0 1 0	u[0,1]	$h^{-2}g[0,1]$
1 1-4 1 1	u[1,1]	f[1,1]
1 1-4 1 1	u[2,1]	f[2,1]
1 1-4 1 1	u[3,1]	f[3,1]
0 1 0	u[4,1]	$h^{-2}g[4,1]$
0 1 0	u[0,2]	$h^{-2}g[0,2]$
1 1-4 1 1	u[1,2]	f[1,2]
1 1-4 1 1	u[2,2]	f[2,2]
1 1-4 1 1	u[3,2]	f[3,2]
0 1 0	u[4,2]	$h^{-2}g[4,2]$
1	u[0,3]	$h^{-2}g[0,3]$
1	u[1,3]	$h^{-2}g[1,3]$
1	u[2,3]	$h^{-2}g[2,3]$
1	u[3,3]	$h^{-2}g[3,3]$
1	u[4,3]	$h^{-2}g[4,3]$

waarin de niet weergegeven matrixelementen alle 0 zijn.

Vooruitlopend op de notatie die in ALGOL 60 voor arrays wordt gebruikt, schrijven we u[j,l] voor de waarde van de roosterfunctie in het punt (jh,lh). Het is echter nogal omslachtig om de u[j,l] met (jh,lh) $\in \partial\Omega$ mee te nemen in de matrixvergelijking, omdat deze reeds bekend zijn. We schrijven (4.33) dan vaak liever in *gereduceerde vorm*

$$(4.36)\quad h^{-2}\left[\begin{array}{ccc|c|ccc} -4 & 1 & 0 & 1 & & & \\ 1 & -4 & 1 & & 1 & & \\ 0 & 1 & -4 & & & 1 & \\ \hline 1 & & & -4 & 1 & 0 & \\ & 1 & & 1 & -4 & 1 & \\ & & 1 & 0 & 1 & -4 & \end{array}\right]\left[\begin{array}{c} u[1,1] \\ u[2,1] \\ u[3,1] \\ \hline u[1,2] \\ u[2,2] \\ u[3,2] \end{array}\right]\left[\begin{array}{c} f[1,1] \\ f[2,1] \\ f[3,1] \\ \hline f[1,2] \\ f[2,2] \\ f[3,2] \end{array}\right]\quad\left[\begin{array}{c} g[1,0]+g[0,1] \\ g[2,0] \\ g[3,0]+g[4,1] \\ \hline g[1,3]+g[0,2] \\ g[2,3] \\ g[3,3]+g[4,2] \end{array}\right].$$

Dit correspondeert met het opleggen van een beperking welke punten we de lopende index Q in (4.32) laten doorlopen, en voor welke P we de vergelijking als vergelijking beschouwen. In (4.36) dus Ω_h i.p.v. $\bar{\Omega}_h$.

We hopen dat aan dit voorbeeld duidelijk is geworden hoe de matrix A behorende bij de differentieoperator L_h, en hoe de vectoren $\vec{u}$ en $\vec{z}$ eruit zien. Merk op dat $\vec{z}$ in zich zowel het rechterlid als de randvoorwaarden van het oorspronkelijk randwaardeprobleem herbergt.

4.3.3. Enkele definities en notaties. Het symbool $\|\vec{u}\|$ zal een niet nader bepaalde vectornorm van $\vec{u}$ aangeven. Het meest zullen we gebruiken de *euclidische* lengte

$$\|\vec{u}\|_2 = \left(\sum_{i=1}^{N} u_i^2\right)^{\frac{1}{2}}$$

en de *maximumnorm*

$$\|\vec{u}\|_\infty = \max_{1\le i\le N} |u_i|.$$

Daarmee corresponderend voeren we twee matrixnormen in, nl.

$$\|A\|_2 = \sup_{\|u\|_2 = 1} \|Au\|_2$$

en

$$\|A\|_\infty = \sup_{\|u\|_\infty = 1} \|Au\|_\infty ,$$

respectievelijk de spectraal en de maximum matrixnorm. Een matrix A kan men spiegelen om de hoofddiagonaal: het resultaat heet de *getransponeerde* A^T van A. Als $A = A^T$, dan heet A *symmetrisch*, als $AA^T = A^TA$, dan heet A

normaal.

De matrixvergelijking

$$A\vec{u} = \vec{z} \tag{4.37}$$

is een bijzonder geval van een klassiek probleem uit de lineaire algebra. Over lineaire stelsels algebraïsche vergelijkingen bestaat een uitgebreide literatuur, en er bestaan allerlei methodes om deze op te lossen. Het ligt voor de hand dat het oplossen van stelsel (4.37) het meest succesvol verloopt, wanneer we de speciale eigenschappen uitbuiten die de matrix A in verband met zijn oorsprong, de differentieoperator L_h, heeft.

4.3.4. Enkele eigenschappen van de uit L_h verkregen matrix A.

(i) L heeft een grote orde N, van enkele tientallen tot enige tienduizendtallen.
(ii) De verhouding tussen het aantal elementen dat van 0 verschilt tot het totaal aantal elementen is zeer klein. Zo'n matrix heet een *ijle* matrix. Een element a(P,Q) (zie (4.32)) verschilt namelijk slechts dan van 0, wanneer de roosterpunten P en Q *gekoppeld* voorkomen in een differentie-uitdrukking die een afgeleide of een randvoorwaarde representeert. In (4.35) zien we dat er hoogstens 5 elementen verschillend van 0 op een rij voorkomen.
(iii) Een belangrijke eigenschap van A is dat de elementen die van 0 verschillen vaak gemakkelijk te genereren zijn wanneer ze nodig zijn, zodat ze geen geheugenruimte in de computer vereisen. Dit is zeker het geval wanneer de coëfficiënten van de elliptische partiële differentiaalvergelijking eenvoudig zijn. De technieken om de elementen a(P,Q) te verkrijgen kunnen dan in een procedure worden opgenomen. In paragraaf 4.7 zal deze residual heten. Een dergelijke matrix heet een *gegenereerde* matrix, in tegenstelling tot een *opgeslagen* matrix.
(iv) Wanneer in het oorspronkelijke continue randwaardeprobleem de operetor zelfgeadjungeerd is, dan is de matrix van het *gereduceerde* stelsel vergelijkingen *symmetrisch*, d.w.z. $A = A^T$. Merk echter op dat, hoewel (4.36) een symmetrische matrix heeft, de matrix in (4.35) niet symmetrisch is. Bij problemen, waarbij het gebied Ω niet zodanig is dat de rand ervan bestaat

uit lijnstukken die roosterpunten verbinden, zullen we er in het algemeen niet in slagen om een symmetrische matrix te krijgen.
(v) Vaak is A een *definitieve* matrix. Dit maakt het oplossen aanmerkelijk prettiger.
(vi) We herinneren eraan dat een *permutatiematrix* Π een vierkante matrix is, waarvan de elementen 0 of 1 zijn, met precies één element 1 in elke kolom en elke rij. Voor Π geldt: $\Pi\Pi^T = I$, zodat $\Pi^{-1} = \Pi^T$.

Schrijf nu $A = (a_{ij})$. Dan heet A *reduceerbaar* als de verzameling gehele getallen $\{1,2,\ldots,N\}$ de vereniging is van twee niet-lege verzamelingen S en T zodanig dat $a_{ij} = 0$ voor alle $i \in S$ en alle $j \in T$. D.w.z. A is reduceerbaar als er een permutatiematrix Π bestaat zodanig dat

$$\Pi A \Pi^T = \begin{bmatrix} A_1 & 0 \\ A_2 & A_3 \end{bmatrix}. \tag{4.38}$$

Als de matrix A reduceerbaar is, dan betekent dit dat een aantal $N_1 < N$ componenten van de vector u in $A\vec{u} = \vec{z}$ eenduidig is bepaald door een N_1-tal componenten van de vector $\vec{z}$. Dit kan optreden wanneer Ω_h uiteenvalt in twee verzamelingen waarvan de punten op geen enkele wijze gekoppeld zijn. Maar dan hebben we met twee randvoorwaardeproblemen te maken. We zullen daarom altijd aannemen dat A niet reduceerbaar is.
(vii) Vaak is de resulterende matrix *diagonaal dominant*, d.w.z. dat $A = (a_{ij})$ voldoet aan

$$|a_{ii}| \geq \sum_{j=1}^{N} |a_{ij}|, \quad i = 1,\ldots,N,$$

met strikte ongelijkheid voor minstens één i.

4.3.5. Directe en iteratieve oplossingsmethoden. Methoden om de matrixvergelijking (4.37) op te lossen, worden gewoonlijk onderscheiden in *directe* en *iteratieve* oplossingsmethoden. Directe methoden, waartoe bijvoorbeeld *Gauss-eliminatie* behoort, zijn die waarin het exacte antwoord in een eindig aantal stappen gevonden wordt, als er geen afrondingsfouter optraden. Dergelijke algoritmen zijn ingewikkeld en niet-herhalend. Iteratieve methoden bestaan uit een herhaald toepassen van een eenvoudig algoritme, maar bereiken, zelfs bij afwezigheid van afrondingsfouten, de exacte oplossing slechts als limiet van een rij.

In deze syllabus zullen we ons *beperken tot iteratieve oplossingsmethoden.* Voor directe zij verwezen naar Dekker [1968] en Dekker & Hoffmann [1968]. Een reden om directe methoden *niet* te gebruiken bij het oplossen van matrixvergelijkingen die afkomstig zijn van elliptische randwaardeproblemen is, dat het opslaan van de matrixelementen vaak niet zal lukken. Zelfs als men de matrix A opslaat als een bandmatrix (zie Dekker [1968]), dan nog heeft men voor een differentieprobleem ment een rooster van 20 ×20 punten zo'n 33600 woorden in de X-8 rekenmachine nodig, en dat is dan nog een bescheiden rooster. Iteratieve methoden buiten de eigenschap uit dat A gegenereerd kan worden, en dat A veel nullen bevat. Verder menen sommige auteurs (bijv. Forsyth & Wasow [1960]) dat iteratieve methoden minder last hebben van afrondigsfouten dan directe methoden.

4.4. Stationaire lineaire iteratieve methoden

4.4.1. Algemeen. In iedere iteratieve methode om het niet-singuliere stelsel(4.37) op te lossen (met oplossing $A^{-1}\vec{z}$) wordt een rij $\vec{u}_k$ gedefinieerd in de hoop dat $\vec{u}_k \to A^{-1}\vec{z}$ als $k \to \infty$. Iteratiemethoden van de *eerste orde* hebben de vorm

$$(4.39) \qquad \vec{u}_k = f_k(A, \vec{z}, \vec{u}_{k-1}).$$

Als f_k onafhankelijk is van k, dan is het iteratieproces *stationair*. Als f_k een lineaire functie is van $\vec{u}_{k-1}$, dan heet het proces *lineair*. Een lineair stationair proces kan men schrijven in de vorm

$$(4.40) \qquad \vec{u}_k = H\vec{u}_{k-1} + M\vec{z}, \quad k = 1,2,3,\ldots,$$

waarin $\vec{u}_0$ de *beginapproximatie* van het proces is. Als (4.40) convergeert, dan is dit naar

$$\vec{u} = (1-H)^{-1} M\vec{z}$$

Definieer $\vec{v}_k = \vec{u}_k - \vec{u}$. Deze voldoen aan

$$(4.41) \qquad \vec{v}_k = H\vec{v}_{k-1}, \quad k=1,2,3,\ldots,$$

zodat na k iteraties de fout $\vec{v}_k$ gegeven wordt door

(4.42) $\qquad \vec{v}_k = H^k \vec{v}_0 .$

De bruikbaarheid van het iteratieproces hangt af van $\| H^k \|$, immers

(4.43) $\qquad \| \vec{v}_k \| \leq \| H^k \| \quad \| \vec{v}_0 \|$

Het is duidelijk dat $\| H^k \| < 1$ moet zijn, wil de methode convergeren. De *gemiddelde convergentiesnelheid* na k iteraties R(k) wordt gedefinieerd door

(4.44) $\qquad R(k) = - \frac{\ln \| H^k \|}{k} .$

4.4.2. Stelling. Voor grote waarden van k wordt de gemiddelde convergentiesnelheid gegeven door

(4.45) $\qquad R(k) \sim - (1 - \frac{p-1}{k}) \ln \sigma(H) - \frac{\ln \nu + (p-1) \ln k}{k}$

waarin ν een constante is, $\sigma(H)$ de *spectraalradius* van H (d.w.z. het maximum van de absolute waarden van de eigenwaarden van H), en waarin p de grootste orde is van alle diagonale ondermatrices J_r van de Jordan-normaalvorm J van H met $\sigma(J_r) = \sigma(H)$.

Bewijs. Zie Van der Houwen [1968], p.31. □

Opmerking. Voor *normale matrices* H (d.w.z. $HH^T = H^T H$) geldt $p = \nu = 1$, zodat (4.45) overgaat in

(4.46) $\qquad R(k) = - \ln \sigma(H).$

4.4.3. Modelprobleem. Om verschillende methoden te vergelijken past men deze meestal toe op de matrixvergelijking behorende bij het Dirichletprobleem met verdwijnende randvoorwaarden voor $- \Delta_h^+ u = f$ met $\xi = \eta = h$. We kiezen $- \Delta_h^+$ om ervoor te zorgen dat de eigenwaarden positief zijn. Dit probleem heet het *modelprobleem*, en wordt daarom gekozen, omdat een gedetailleerde analyse ervan mogelijk is. De eigenwaarden $\delta_{m,n}$ van de ope-

rator Δ_h^+ zijn

(4.47) $\qquad \delta_{m,n} = 2(-2 + \cos mh + \cos nh)h^{-2}$

en de eigenfuncties

(4.48) $\qquad e_{m,n} = \sin mjh \sin nlh$

hetgeen door invullen is te bewijzen. Uit (4.47) volgt voor kleine h dat de eigenwaarde $-\delta_{m,n}$ van $-\Delta_h^+$ voldoen aan

$$2 \leq -\delta_{m,n} \leq 8h^{-2} - 2.$$

4.4.4. De methode van Jacobi. Schrijf de matrix $A = (a_{ij})$ in (4.37) in de vorm

(4.49) $\qquad A = E + D + F \ ,$

waarin E de elementen onder de diagonaal bevat, en verder nullen, F de elementen boven de diagonaal, en verder nullen, en D de diagonaalelementen en verder nullen. Neem aan dat

(4.50) $\qquad a_{ii} \neq 0$

De methode van Jacobi, ook wel die der *simultane verplaatsingen* genaamd, wordt nu gedefinieerd door

$$E\vec{u}_{k-1} + D\vec{u}_k + F\vec{u}_{k-1} = \vec{z}$$

ofwel

$$\vec{u}_k = -D^{-1}(E+F)\vec{u}_{k-1} + D^{-1}\vec{z} =$$
$$= (I - D^{-1}A)\vec{u}_{k-1} + D^{-1}\vec{z} \ .$$

Dus is

$$H = I - D^{-1}A \ .$$

We merken op dat D^{-1} direct bepaald kan worden omdat D een diagonaalmatrix is.

4.4.5. Stelling. Als A een niet reduceerbare diagonaal dominante matrix is, dan convergeert de methode van Jacobi.

Bewijs. Zie Forsythe & Wasow [1960], p.221. □

4.4.6. De methode van Jacobi voor het modelvoorbeeld. De gereduceerde matrix corresponderende met $-\Delta_h^+$ is

$$A=h^{-2}\begin{bmatrix} 4 & -1 & & -1 & & & \\ -1 & 4 & -1 & & \ddots & & \\ & -1 & 4 & -1 & & \ddots & \\ -1 & & -1 & 4 & -1 & & -1 \\ & \ddots & & -1 & 4 & -1 & \\ & & \ddots & & -1 & 4 & \\ & & & -1 & & & \end{bmatrix}$$

waaruit E, D en F onmiddelijk zijn af te lezen. Daar A symmetrisch is en de hoofddiagonaalelementen allen gelijk zijn, is $H = I - D^{-1}A$ ook symmetrisch. Een eenvoudige berekening leert dat

$$(4.50) \qquad H = I - \frac{h^2}{4} A$$

en

$$(4.51) \qquad R(k) = -\ln \|H\| = -\ln \sigma(H) \simeq \frac{h^2}{2} \qquad \text{voor kleine } h.$$

4.4.7. Definitie. Het k^e *residu* behorende bij de matrixvergelijking $A\vec{u} = \vec{z}$ wordt gedefinieërd als

$$(4.52) \qquad \vec{r}_k = A\vec{u}_k - \vec{z} \ ,$$

waarin $\vec{u}_k$ de k^e iterand is; $\vec{r}_k$ is een maat die aangeeft in hoeverre $\vec{u}_k$ aan de vergelijking voldoet.

4.5 De methode van Richardson

4.5.1. Niet-stationaire methoden. We zullen een niet-stationair iteratief

proces beschrijven om het stelsel vergelijkingen (4.37) op te lossen. We nemen daarbij aan dat A *reële* eigenwaarden bezit en een *normale* matrix is (d.w.z. $HH^T = H^TH$). Beide zijn bijvoorbeeld het geval als A symmetrisch is. We nemen verder voorlopig aan dat A *positief definiet* is, d.w.z. dat alle eigenwaarden α_j van A positief zijn. We nemen aan dat $a \leq \alpha_j \leq b$, voor alle j.

Voor dergelijke matrices beschouwde Richardson [1910] het volgende niet-stationaire proces

$$(4.54) \qquad \vec{u}_{k+1} = (1-\omega_k A)\vec{u}_k + \omega_k \vec{z} = \vec{u}_k - \omega_k(A\vec{u}_k - \vec{z}) \quad .$$

Hierin stellen de ω_k nog nader te bepalen, zgn. *relaxatieparameters* voor. De fout $\vec{v}_k = \vec{u}_k - \vec{u}$ voldoet aan

$$(4.55) \qquad \vec{v}_{k+1} = \prod_{j=0}^{k-1} (1-\omega_j A)v_0 = P_k(A)v_0 \ ,$$

waarin dus $P_k(A)$ een polynoom van de graad k in A is. Uit (4.55) volgt

$$(4.56) \qquad \|\vec{v}_k\|_\infty \leq \|P_k(A)\|_\infty \ \|\vec{v}_0\|_\infty$$

Volgens stelling 4.4.2. geldt voor normale matrices dat $\sigma(A) = \|A\|_\infty$, zodat we voor (4.56) ook mogen schrijven

$$(4.57) \qquad \|\vec{v}_k\|_\infty \leq \sigma(P(A))\|\vec{v}_0\|_\infty = \max_j |P(\alpha_j)| \ \|\vec{v}_0\|_\infty$$

Het gaat er dus kennelijk om $\max_j|P(\alpha_j)|$ zo klein mogelijk te krijgen, want dan is $\|\vec{v}_k\|$ zo klein mogelijk. Omdat we de eigenwaarden α_j niet kennen, betekend dit dat we $|P_k(\alpha)|$ klein moeten maken over het hele interval $a \leq \alpha \leq b$.

Richardson stelde in 1910 voor de k nulpunten ω_j^{-1} van het polynoom $P_k(\alpha) = \prod (1-\omega_j\alpha)$ gelijkmatig over het interval te verdelen, en inderdaad bereikte hij daarmee een effectief resultaat. Een veel verfijndere gedachte is natuurlijk (Flanders & Shortley [1950], Young [1953]) het polynoom $P_k(\alpha)$

met $P_k(0) = 1$ zodanig te kiezen, dat de waarde van $|P_k(\alpha)|$ over $a \leq \alpha \leq b$ zo klein mogelijk is.

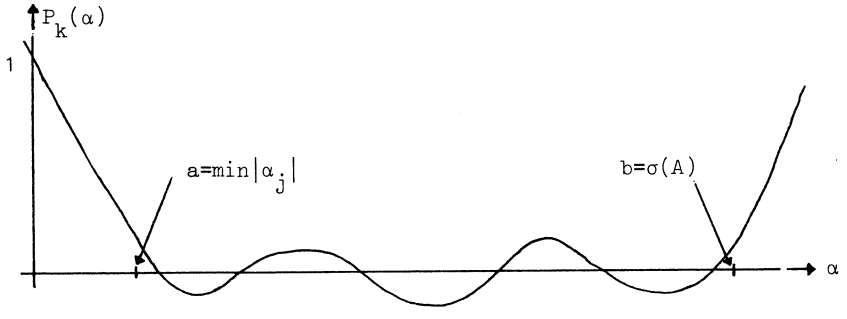


fig. 4.7

4.5.2. Stelling. Het polynoom

$$C_k(a,b,\alpha) = \frac{T_k(\frac{b+a-2\alpha}{b-a})}{T_k(\frac{b+a}{b-a})} \quad , \tag{4.58}$$

waarin $T_k(y)$ het Chebyshev polynoom van graad k is, heeft van alle k^e graads polynomen in α die voldoen aan $P_k(0) = 1$ de kleinste maximumnorm over het interval [a,b].

Bewijs. Volgt onmiddellijk uit de minimaxeigenschap van de Chebyshev polynomen $T_k(y)$, die door Markov in 1886 bewezen is. □

4.5.3. Het eerste orde proces van Richardson. We definiëren Richardsons methode m.b.t. de matrix A door de keuze

$$P_k(A) = C_k(a,b,A) \quad , \qquad a \leq \min_j(\alpha_j) \quad , \quad b \geq \sigma(A) \quad . \tag{4.59}$$

Vaak zullen we de eigenwaarden α_j oplopend in grootte geordend denken, zodat we dan ook hebben $\alpha_1 = \min(\alpha_j) \geq a$.

We moeten nu de relaxatieparameters bepalen. De nulpunten van $P_k(\alpha)$ zijn

$$\alpha = \omega_j^{-1} \quad , \qquad j = 0,1,\ldots,k-1 \quad , \tag{4.60}$$

en de nulpunten van $C_k(a,b,\alpha)$ worden gegeven door

$$(4.61) \qquad \alpha = \tfrac{1}{2}(a+b) + \tfrac{1}{2}(a-b)\cos\frac{2j+1}{2k}\pi \ , \qquad j = 0,1,\ldots,k-1 \ .$$

Gelijkstellen van de rechterleden van (4.60) en (4.61) levert dan de waarden van ω_j.

In overeenstemming met 4.4.1. definiëren we de gemiddelde convergentiesnelheid na k iteraties als

$$(4.62) \qquad P(k) = -\frac{\ln\| P_k(A)\|_\infty}{k} \ .$$

Omdat $P_k(A)$ normaal is, hebben we

$$\| P_k(A)\|_\infty = \sigma(P_k(A)) \simeq \max_{a\le\alpha\le b} |P_k(\alpha)| \ .$$

Uit de definitie van de Chebyshev polynomen volgt dat

$$(4.63) \qquad T_k(\frac{b+a}{b-a}) \simeq \cosh(2k\sqrt{\frac{a}{b}})$$

voor a << b en k >> 1 (zie Forsythe & Wasow [1960], p.228 e.v.),zodat

$$(4.64) \qquad \max_{a\le\alpha\le b} |P_k(\alpha)| \simeq [\cosh(2k\sqrt{\frac{a}{b}})]^{-1}.$$

Hieruit volgt

$$(4.65) \qquad R(k) \simeq \frac{1}{k}\ln[\cosh(2k\sqrt{\frac{a}{b}})] \simeq 2\sqrt{\frac{a}{b}} - \frac{\ln 2}{k}$$

voor k >> 1 en a << b.

4.5.4. <u>Toepassing op het modelvoorbeeld</u>. Om de zojuist besproken methode te vergelijken met die van Jacobi (zie vorige paragraaf), passen we deze toe op het modelvoorbeeld. Daar is $\alpha_1 = 2$ en $\sigma(A) \simeq 8h^{-2}$, zodat (4.65) overgaat in

$$(4.66) \qquad R(k) \simeq h - \frac{\ln 2}{k} \ ,$$

hetgeen aanmerkelijk sneller is dan $h^2/2$ (vgl. (4.51)).

4.5.5. Nadelen van het eerste orde proces van Richardson. Een groot nadeel is dat men *van te voren* de graad van het toe te passen polynoom $C_k(a,b,\alpha)$ moet geven, en dit niet kan laten afhangen van de resultaten tijdens het rekenproces. Een verstandige tactiek is vaak het aantal iteratieslagen te laten bepalen door de opgegeven nauwkeurigheid. Dit is hier dus niet mogelijk. Bovendien blijkt dat de eerste orde methode van Richardson instabiel is voor grote waarden van k en b/a. Dit zullen we niet bewijzen, maar aannemelijk maken m.b.v. figuur 4.8. Het toepassen van het polynoom $C_k(a,b,A)$ op $\vec{v}_0$ komt in feite neer op het k maal toepassen van een lineaire operator $(1-\omega A)$. In de figuur wordt duidelijk wat er gebeurt als ω^{-1} links van het

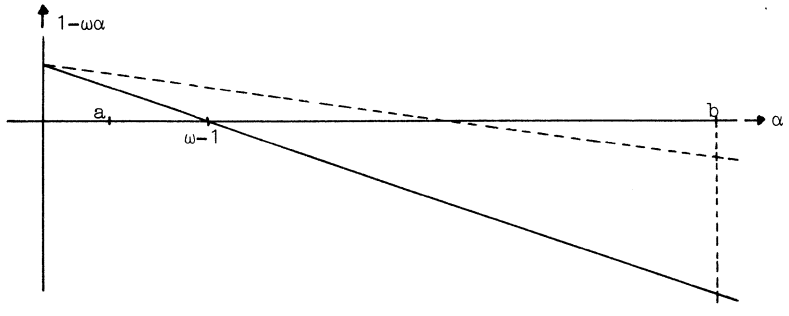


fig. 4.8

midden van [a,b] ligt: als we $\vec{v}_0$ ontbinden naar de eigenvectoren, i.e.

$$(4.67) \qquad \vec{v}_0 = \sum_j c_j \vec{e}_j \ ,$$

dan betekent toepassen van $(1-\omega A)$ dat die componenten van $\vec{v}_0$ langs eigenvectoren behorende bij grote eigewaarden met een factor groter dan 1 worden vermenigvuldigd. Men kan dan ook gemakkelijk begrijpen dat de instabilitiet sterk afhangt van de keuze van de volgorde van de relaxatieparameters ω_j.

4.5.6. Tweede orde Richardson proces. Als men voldoende geheugenruimte heeft, kan men deze nadelen opheffen door het *tweede orde* proces van Richardson te gebruiken. Beschouw het tweede orde proces

$$(4.68) \qquad \vec{u}_{k+1} = (\beta_k - \omega_k A)\vec{u}_k + (1-\beta_k)\vec{u}_{k-1} + \omega_k \vec{z} \ , \qquad k = 1,2,\ldots,$$

waarin nu $\vec{u}_0$ en $\vec{u}_1$ beginapproximaties zijn. De fout $\vec{v}_k$ voldoet aan

$$(4.69)\qquad \vec{v}_{k+1} = (\beta_k-\omega_k A)\vec{v}_k + (1-\beta_k)\vec{v}_{k-1}, \qquad k = 1,2,....$$

Verder nemen we aan dat $\beta_0 = 1$, zodat

$$(4.70)\qquad \vec{v}_1 = (1-\omega_0 A)\vec{v}_0 .$$

Merk op dat keuze $\beta_k = 1$ voor alle k, leidt tot formule (4.55).
We hebben weer

$$\vec{v}_k = P_k(A)\vec{v}_0 , \qquad k = 1,2,... \quad .$$

Substitueer dit in (4.69), dan moeten de polynomen kennelijk voldoen aan de recurrente betrekking

$$(4.71)\qquad P_{k+1}(\alpha) = (\beta_k-\omega_k)P_k(\alpha) + (1-\beta_k)P_{k-1}(\alpha) \quad .$$

Aan de andere kant kunnen we uit de welbekende recurrente betrekking voor Chebyshev polynomen

$$T_{k+1}(y) = 2yT_k(y) - T_{k-1}(y) \quad , \qquad k \geq 1 ,$$

de volgende formule voor $C_k(a,b,\alpha)$ afleiden:

$$(4.72)\qquad C_{k+1}(a,b,\alpha) = (2y_0-\frac{4\alpha}{b-a})\frac{T_k(y_0)}{T_{k+1}(y_0)}\ C_k(a,b,\alpha) +$$

$$-2y_0\ \frac{T_k(y_0) - T_{k+1}(y_0)}{T_{k+1}(y_0)}\ C_{k-1}(a,b,\alpha) \quad ,$$

waarin $y_0 = (b+a)/(b-a)$ en $k \geq 1$. Als we nu voor $k \geq 1$ definiëren

$$(4.73)\qquad \beta_k = 2y_0\ \frac{T_k(y_0)}{T_{k+1}(y_0)} \quad , \qquad \omega_k = \frac{4}{b-a}\frac{T_k(y_0)}{T_{k+1}(y_0)} \quad ,$$

dan zijn de relaties (4.71) en (4.72) identiek. Als we aanvullend definiëren

$$(4.74)\qquad \beta_0 = 1, \qquad \omega_0 = \frac{2}{b+a} ,$$

dan verkrijgen we polynomen $P_k(\alpha)$, die identiek zijn met de polynomen $C_k(a,b,\alpha)$ voor elke k. Door $a \leq \min(\alpha_j)$ te nemen, en $b \geq \sigma(A)$, verkrijgen we de tweede orde methode van Richardson.

Het verschil tussen de eerste en de tweede orde methode is *slechts* het volgende. Stel dat we n iteratieslagen willen uitvoeren. Dan is, als we afzien van afzonderingsfouten, na n iteraties in beide methoden precies hetzelfde resultaat bereikt. Maar voor **iedere** k, $1 \leq k < n$, hebben we bij het tweede orde proces ook een Chebyshev polynoom toegepast, van graad k, zodat de tussenresultaten ook zinvol zijn, in tegenstelling tot het eerste orde proces, dat zinloze tussenresultaten oplevert. We hoeven ons dus ook niet van te voren op een bepaald aantal slagen vast te leggen bij het tweede orde proces.

4.5.7. De procedure "richardson". De methode in deze paragraaf uiteengezet is geheel vervat in de procedure "richardson", die in paragraaf 4.7 beschreven wordt.

4.5.8. Boven- en ondergrens voor de eigenwaarden. Bij de methode van Richardson kunnen we de kennis van een boven- en ondergrens voor de eigenwaarden niet ontberen. In de volgende paragraaf zal blijken dat we a niet zodanig hoeven te kiezen dat alle $\alpha_j > a$, zodat hier enig soelaas optreedt. Een bovengrens voor de eigenwaarden is te vinden m.b.v. de volgende stelling.

4.5.9. Stelling (Gerschgorin). De eigenwaarden van $A = (a_{ij})$ (evt. complexwaardig) liggen in de vereniging van de cirkels

$$(4.75) \qquad |z-a_{ii}| \leq \sum_{\substack{j=1 \\ j\neq 1}}^{n} |a_{ij}| , \qquad i=1,\ldots,n , \quad z \in C .$$

Bewijs. Zie bijv. Varga [1962]. □

Gevolg (belangrijk voor ons omdat we met reële eigenwaarden te maken hebben): Laat α_{max} de eigenwaarde zijn van A met grootste absolute waarde. Dan geldt

$$(4.76) \qquad |\alpha_{max}| \leq \max_i \sum_j |a_{ij}| ,$$

en

$$(4.77) \qquad |\alpha_{max}| \leq \max_j \sum_i |a_{ij}| \quad .$$

4.6. Versnelling van de methode van Richardson

4.6.1. Beschrijving. Van de methode van Richardson wordt een variant behandeld, die een grotere convergentiesnelheid heeft. In essentie komt deze methode erop neer, dat het polynoom $C_k(a,b,\alpha)$ nu zo gekozen wordt dat niet $a \leq \min(\alpha_j)$, maar $a > \min(\alpha_j)$. Toepassing van $C_k(a,b,A)$ op een begin-

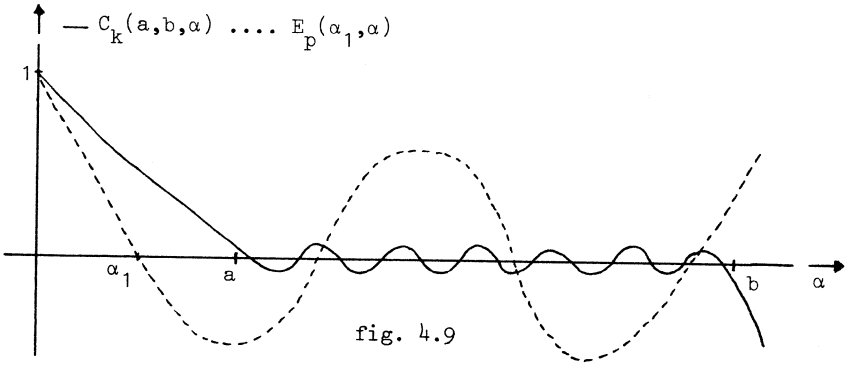


fig. 4.9

approximatie met fout $\vec{v}_0 = \sum_j c_j \vec{e}_j$ heeft dan het volgende effect:

$$C_k(a,b,A)\vec{v}_0 = \sum_j C_k(a,b,\alpha_j) c_j \vec{e}_j \quad ,$$

waarin de factor waarmee $\vec{e}_1$ wordt vermenigvuldigd aanmerkelijk groter is dan die waarmee de overige $\vec{e}_j$ worden vermenigvuldigd. De component van $\vec{v}_0$ in de richting van $\vec{e}_1$ springt er dus uit. Door daarna een tweede polynoom $E_p(A)$ toe te passen dat precies door α_1 gaat, wordt deze component vernietigd. Het toepassen van $C_k(a,b,A)$ noemen we de *reductie*fase, het toepassen van E_p de *eliminatie*fase van het proces.

We behandelen, zeer in het kort, de volgende vier vragen.

(i) Vind een formule, zodanig dat α_1 tijdens het proces wordt bepaald.

(ii) Vind polynomen $E_p(A)$ die elimineren.

(iii) Wat is de convergentiesnelheid van deze versnelde methode ?

(iv) Bepaal de graad p van $E_p(A)$.

4.6.2. De procedure "elimination". Deze variant van de methode van Richard-

son wordt geëffectueerd door eerst de procedure "richardson" toe te passen met een geschatte $a > \alpha_1$, en daarna de procedure "elimination", die in de volgende paragraaf wordt beschreven.

4.6.3. Bepaling van de kleinste eigenwaarde. Er geldt

$$\vec{u}_{k+1} = \vec{u} + \vec{v}_{k+1} = \vec{u} + C_{k+1}(a,b,A)\vec{v}_0$$

en

$$\vec{u}_k = \vec{u} + \vec{v}_k = \vec{u} + C_k(a,b,A)\vec{v}_0 \quad .$$

Aftrekken levert

$$\vec{u}_{k+1} - \vec{u}_k = [C_{k+1}(a,b,A) - C_k(a,b,A)]\vec{v}_0 .$$

Als nu $\alpha_1 < a$ is, en $a \leq \alpha_j \leq b$ voor $j \geq 2$, dan geldt voor voldoend grote k

$$u_{k+1} - u_k \simeq \text{const}[C_{k+1}(a,b,A) - C_k(a,b,A)]\vec{e}_1 = \tag{4.78}$$

$$= \text{const}[C_{k+1}(a,b,\alpha_1) - C_k(a,b,\alpha_1)]\vec{e}_1 \quad .$$

Definieer nu het quotiënt

$$q_{k+1} = \frac{\| u_{k+1} - u_k \|}{\| u_k - u_{k-1} \|} \quad , \tag{4.79}$$

dat tijdens het rekenproces gemakkelijk te bepalen is. Uit (4.78) volgt dat

$$q_{k+1} \simeq \frac{|C_{k+1}(a,b,\alpha_1) - C_k(a,b,\alpha_1)|}{|C_k(a,b,\alpha_1) - C_{k-1}(a,b,\alpha_1)|} \quad . \tag{4.80}$$

Door nu de eigenschappen van de uit de Chebyshev polynomen afgeleide polynomen $C_k(a,b,\alpha_1)$ uit te buiten, kan men een formule voor α_1 vinden van de vorm

$$\alpha_1 = \alpha_1(q_{k+1},a,b) \quad , \tag{4.81}$$

hetgeen tijdens het proces te berekenen is. Voor precieze details zij verwezen naar Van der Houwen [1967]. In de procedure "elinination" is deze bepaling van de kleinste eigenwaarde opgenomen.

4.6.4. Stelling. De keuze van het polynoom E . Schrijven we het polynoom dat de eigenvector behorende bij α_1 moet elimineren, als $E_p(\alpha_1,A)$, dan moet E_p voldoen aan

$$E_p(\alpha_1,0) = 1 , \qquad E_p(\alpha_1,\alpha_1) = 0 . \tag{4.82}$$

Het polynoom

$$E_p(\alpha_1,\alpha) = C_p(a_1^*,b,\alpha)$$

met

$$a_1^* = \frac{2\alpha_1 + b(\cos(\pi/(2p))-1)}{\cos(\pi/(2p))+1}$$

voldoet aan de eis (4.82). Bovendien is dit polynoom onder de polynomen die aan (4.82) voldoen het polynoom dat de kleinste maximumnorm heeft over $[a_1^*,b]$.

Bewijs. Zie Van der Houwen [1967], p.9 e.v.. □

4.6.5. De convergentiesnelheid. Het totaal toegepaste polynoom na k iteraties in de reductiefase en p in de eliminatiefase is $C_p(a_1^*,b,A)C_k(a,b,A)$, zodat de gemiddelde convergentiesnelheid na k+p iteraties gegeven wordt door

$$\begin{aligned} R(k+p) &= -\frac{1}{k+p}(\ln \| C_k(a,b,A)\|_\infty + \ln \| C_p(a_1^*,b,A)\|_\infty) = \\ &= -\frac{1}{k+p}(\ln \sigma(C_k(a,b,A)) + \ln \sigma(C_k(a_1^*,b,A))) . \end{aligned} \tag{4.83}$$

Met gebruikmaking van (4.65) is dit voor grote k, normale A en $a \ll b$ gelijk aan

$$R(k+p) \simeq 2\sqrt{\frac{a}{b}} - \frac{2p\sqrt{\frac{a}{b}} + \ln (C_p(a_1^*,b,A)) + \ln 2}{k+p} . \tag{4.84}$$

Winst t.o.v. de gewone methode van Richardson: voor $k \to \infty$ is de convergentiesnelheid $\sqrt{(a/\lambda_1)}$ maal zo groot.

4.6.6. De graad p van $C_p(a_1^*,b,A)$. We willen de uitdrukking (4.84), als func-

tie van p, bij gegeven k, zo groot mogelijk maken. Hieruit volgt dan een vergelijking voor p, die wij hier niet zullen reproduceren. De grootte van $\ln (C_p(a_1^*,b,A))$ is dan eenvoudig te bepalen. Men zij verwezen naar Van der Houwen [1968], p.102. In de procedure "elimination" wordt de graad automaties volgens het in de referentie gegeven procédé bepaald.

4.7. De procedures "richardson" en "elimination"

We geven nu een beschrijving van de ALGOL 60 procedure "richardson" en "elimination", die in de losbladige reeks van het Mathematisch Centrum onder de nummers LR 3.3.10 resp. LR 3.3.11 zijn opgenomen. Beide gaan uit van een matrixvergelijking

$$A\vec{u} = \vec{z}$$

waarin A een matrix met positieve eigenwaarden is. Bij het gebruik van "richardson" moet het interval [a,b] waarover reductie plaatsvindt door de gebruiker gegeven worden. Voor b moet een bovengrens voor de eigenwaarden van A gekozen worden; hoe scherper deze bovengrens, hoe beter. Als men voor a een getal kleiner dan de kleinste eigenwaarde kiest, dan wordt reductie op alle eigenvectoren van A uitgeoefend; kiest men a groter dan de kleinste eigenwaarde, dan zal deze, of een gemiddelde van een aantal van de kleinste eigenwaarden, domineren. De waarde hiervan komt in domeigval terecht. Toepassing van de procedure "elimination" doet in het geval dat domeigval gelijk is aan een eigenwaarde, de component van de fout langs de eigenvector behorende bij deze eigenwaarde sterk verminderen.

4.7.1. De procedure richardson. Declaratie en betekenis van de parameters worden hier gegeven. De tekst van de body vindt men bij het uitgewerkte voorbeeld in de volgende paragraaf.

Declaratie:

```
procedure richardson (u,lj,uj,ll,inap,residual,a,b,n,discr,k,rateconv,
                      domeigval,output);
value                 lj,uj,ll,ul,a,b;
integer               lj,uj,ll,ul,n,k;
real                  a,b,rateconv,domeigval;
```

```
array       u,discr;
boolean     inap;
procedure   residual,output;
```

Parameters:

u : <array identifier>;
in het tweedimensionale array u[lj:uj,ll:ul] staat na iedere iteratieslag de door de procedure berekende benaderde oplossing van $A\vec{u} = \vec{z}$; de gebruiker *kan* in u een beginapproximatie opgeven bij de aanroep van richardson.

lh,uj,ll,ul : <expression>;
onder- en bovengrenzen van het array u.

inap : <boolean>;
als de gebruiker van een beginapproximatie, te geven in het array u, wil uitgaan, dan moet inap:= true gekozen worden; bij de keuze inap:= false wordt door de procedure als beginapproximatie een vector met alle componenten 1 aangehouden.

residual : <procedure identifier>;
deze procedure moet door de gebruiker gedeclareerd worden:
procedure residual (u); array u;
<body>;
de gebruiker moet ervoor zorgen dat bij aanroep residual het meegegeven array u verandert in Au-z en in u substitueert; wanneer het matrixprobleem afkomstig is van een elliptische differentiaalvergelijking, is het handig om bij het schrijven van de body van residual van de molecuulnotatie (paragraaf 4.2) uit te gaan.

a,b : <expression>;
in a en b moet de gebruiker onder- en bovengrens van het reductie-interval aangeven; er moet gelden a > 0.

n : <expression>;

n geeft het aantal iteratieslagen aan, dat richardson zal uitvoeren; n kan via een "Jensen re parameters worden gekoppeld.

discr : <array identifier>; array discr [1:2] ;
in dit array levert richardson na iedere iteratieslag de volgende waarden af:
in discr[1] $\| A\vec{u}-\vec{z}\|_2$, waarin $\| \ \|_2$ euclidische norm;
in discr[2] $\| A\vec{u}-\vec{z}\|_\infty$, waarin $\| \ \|_\infty$ maximumnorm.

k : <variable>;
k telt de iteratieslagen die de procedure richardson uitvoert, en kan o.a. voor output als Jensen-parameter dienen.

rateconv : <variable>;
na iedere iteratie staat hierin de convergentiesnelheid.

domeigval : <variable>;
na iedere iteratie wordt door de procedure aan domeigval de waarde van de dominante eigenwaarde, voorzover aanwezig, afgeleverd volgens formules (4.80) en (4.81)

output : <procedure identifier>;
deze procedure moet door de gebruiker zelf gedeclareerd worden als volgt:
procedure output(k); value k; integer k;
<body>;
via deze procedure kan men beschikken over de volgende grootheden:
voor $0 \leq k \leq n$: in array u; $\| A\vec{u}_k-\vec{z}\|_{2,\infty}$ in resp. discr [1] en discr [2];
voor $0 \leq k \leq n$: bovendien rateconv en domeigval m.b.t. $\vec{u}_k$.

Gebruikte procedures: geen.

4.7.2. De procedure "elimination".

Declaratie:

```
procedure elimination (u,lj,ul,ll,ul,residal,a,b,n,discr,k,rateconv,
                       domeigval, output);
value     lj,uj,ll,ul,a,b;
integer   lj,uj,ll,ul,n,k;
real      a,b,rateconv,domeigval;
array     u,discr;
procedure residual,output;
```

Parameters:

u :<array identifier>;
in het tweedimensionale array u[lj:uj,ll:ul] staat na iedere iteratieslag de door de procedure berekende benaderde oplossing van $A\vec{u}=\vec{z}$; de gebruiker *moet* bij de aanroep van elimination in u een beginapproximatie geven.

lj,uj,ll,ul : <expression>;
onder- en bovengrenzen van het array u;

residual : <procedure identifier>;
als bij richardson.

a,b : <expression>;
b wordt gekozen als bij richardson; de keuze van a doet er niet toe.

n : <variable>;
in n levert elimination de graad van het eliminatiepolynoom af.

discr : <array identifier>;
als bij richardson

k : <variable>;
k telt het aantal iteratieslagen die elimination uitvoert, en kan o.a. voor output als Jensen-parameter dienen.

rateconv : <variable>;

als bij richardson.

domeigval : <variable>;
bij aanroep van elimination moet de gebruiker ervoor zorgen dat domeigval de waarde van de te elimineren eigenwaarde heeft; verder als bij richardson.

output : <procedure identifier>;
als bij richardson.

Gebruikte procedures: zeroin (zie Dekker & Hoffmann [1968]) en richardson.

4.7.3. Het benodigde aantal iteraties. In het algemeen is het niet mogelijk een a priori schatting te geven van het aantal iteraties dat richardson nodig heeft om de oplossing of de dominante eigenwaarde met een bepaalde, gewenste nauwkeurigheid te berekenen. De procedure "richardson" is echter zo geprogrammeerd, dat het mogelijk is het aantal uit te voeren iteraties te laten afhangen van grootheden die bij iedere iteratieslag toch al door "richardson" worden bepaald, met name van
(i) de bij iedere iteratieslag berekende gemiddelde convergentiesnelheid (rateconv);
(ii) de norm van het bij de iteratieslag berekende residu (discr[1] en discr[2]);
(iii) de grootte van het verschil tussen de bij twee opeenvolgende iteraties berekende dominante eigenwaarden (met iets meer moeite).

Van elk van de genoemde mogelijkheden geven we een voorbeeld.
(i) Als men wil dat het iteratieproces afgebroken wordt als de convergentiesnelheid kleiner wordt dan .01, en dat er in ieder geval niet meer dan 50 slagen worden gedaan, dan moet men voor de parameter n de uitdrukking

if rateconv <.01 then k else 50

substitueren.
(ii) Als men het iteratieproces wil laten afbreken als de discrepantie voldoende klein geworden is, dan moet men bijv. voor n substitueren

if discr[1] < $_{10}$-3 then k else 60.

(iii) Dit geval moet men iets anders realiseren, omdat de procedure "richardson" slechts beschikking heeft over de laatst berekende waarde van domeigval en niet over de voorgaande. Door echter bijvoorbeeld twee variabelen d1 en d2 globaal te declareren en de procedure output als volgt te laten beginnen, kan men n laten afhangen van een voldoend nauwkeurig bepaalde domeigval:

```
procedure output (k); value k; integer k;
begin if k = 0 then d1:= d2:= 1 else
     begin d2:= d1; d1:= domeigval;
          n:= if abs ((d1-d2)/d2) < 10↑(-q) then k else 100
     end;
     <vervolg body>
end output;
```

4.8. Uitgewerkt voorbeeld

4.8.1. Beschrijving van werkwijze. We bekijken in een vierkant met zijde π een Dirichletprobleem voor de vergelijking van Poisson, nl.

$$(4.85) \qquad -\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) = F(x,y), \quad 0<x<\pi, \ 0<y<\pi,$$

$$(4.86) \qquad U(x,y) = G(x,y) \qquad \begin{cases} x=0,\pi; & 0<y<\pi, \\ y=0,\pi; & 0<x<\pi, \end{cases}$$

waarin $G(x,y)$ een functie is die op de rand van het vierkant gegeven is.

We leggen een vierkant rooster over het vierkant met roosterafstanden $\xi=\eta=h=\pi/N$, en kiezen als discretisatie van de operator van Laplace Δ de vijfpunts + formule (4.27)

$$\Delta_h^+ = h^{-2} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Het bij (4.85) en (4.86) behorende gediscretiseerde probleem kan geschreven worden als

$$(4.87) \qquad -\Delta_h^+ u[j,l] = f[j,l], \qquad j,l=0,\ldots,N-1 ,$$

$$(4.88) \qquad \begin{cases} u[j,0] = g[j,0], & j=0,\ldots,N, \\ u[0,1] = g[0,1], & l=0,\ldots,N, \\ u[j,N] = g[j,N], & j=0,\ldots,N, \\ u[N,1] = g[N,1], & l=0,\ldots,N. \end{cases}$$

Hier is, zoals in paragraaf 4.2 is afgesproken, u[j,l] = U(jh,lh), f[j,l]= = F(jh,lh), g[j,l] = G(jh,lh). Voor het gemak houden we de niet-gereduceerde vorm (zie paragraaf 4.3) voor het gediscretiseerde probleem aan. Vermenigvuldigen we (4.87) met h^2, dan ziet de uitdrukking $A\vec{u}-\vec{z}$, die we in de procedure "residual" moeten programmeren, er uit als

$$(4.89) \qquad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} u[j,l] - h^2 f[j,l]$$

in de roosterpunten die in het inwendige van het vierkant liggen, en

$$(4.90) \qquad \begin{cases} u[j,0] - g[j,0] , \\ u[0,l] - g[0,l] , \\ u[j,N] - g[j,N] , \\ u[N,l] - g[N,l] \end{cases}$$

in de roosterpunten op de rand. Wanneer we nu voor $\vec{u}_0$ een beginapproximatie kiezen die aan de randvoorwaarden g voldoet, zien we dat het residu $A\vec{u}-\vec{z}$ voor die componenten van $\vec{u}$ die corresponderen met een u[j,l] op de rand steeds de waarde 0 oplevert. In de procedure "residual" die in de nog te volgen programmatekst is opgenomen, is een en ander van bovenstaande beschouwingen in ALGOL 60 omgezet.

4.8.2. De keuze van F en G. We zullen twee gevallen numeriek behandelen, nl. het homogene Dirichletprobleem, waarvoor

$$(4.91) \qquad \begin{cases} F(x,y) \equiv 0 \text{ in het inwendige van het vierkant, en} \\ G(x,y) \equiv 0 \text{ op de rand,} \end{cases}$$

met bijbehorende analytische oplossing $U(x,y) \equiv 0$, en

$$(4.92)\qquad \begin{cases} F(x,y) = -2(x^2+y^2) & \\ G(x,0) = 0\ , & 0<x<\pi\ , \\ G(0,y) = 0\ , & 0<y<\pi\ , \\ G(x,\pi) = \pi^2x^2\ , & 0<x<\pi\ , \\ G(\pi,y) = \pi^2y^2\ , & 0<y<\pi\ , \end{cases}$$

met analytische oplossing $U(x,y) = x^2y^2$. In het nog te volgen programma wordt het rechterlid F geintroduceerd door de procedure "f", de randvoorwaarden door gebruik van "analsol".

4.8.3. De keuze van de overige parameters. Voor ons voorbeeld kozen we een betrekkelijk klein aantal roosterpunten, nl. 10 × 10 inwendige roosterpunten, corresponderend met N = 11, d.w.z. lj = ll = 0, uj = ul = 11. De eigenwaarden van de operator $-\Delta_h^+$ kennen we reeds; de grootste is $8h^{-2}$ -2, de kleinste is 2. De eigenwaarden van $-h^2\Delta_h^+$ liggen dus tussen $2h^2$ en $8-2h^2$. Een eenvoudige berekening m.b.v. (4.47) laat zien dat de op één na kleinste eigenwaarde ongeveer gelijk is aan $4.9h^2$.

4.8.4. Numerieke resultaten. We geven een overzicht van een aantal resultaten, verkregen voor verschillende waarden van a, met en zonder "elimination". In onderstaande tabel staat in de eerste kolom een programmanummer, en in de tweede de, in dit geval bekende, analytische oplossing. Een sterretje (*) bij het getal in de kolom n (aantal slagen "richardson") betekent dat het aantal iteraties automatisch is bepaald afhankelijk van het aantal in kolom q genoemde cijfers dat de waarde van domeigval in twee opeenvolgende iteraties niet mocht veranderen. De wijze waarop dit programmatechnisch is geëffectueerd, is beschreven aan het eind van de vorige paragraaf. In de kolom p staat het aantal uitgevoerde eliminatieslagen; nn=n+p, een maat voor de hoeveelheid werk. $R^*(nn)$ is de numeriek verkregen convergentiesnelheid, terwijl R(nn) de theoretische convergentiesnelheid is, berekend m.b.v. formule (4.84), waarbij voor het laatste gebruik gemaakt is van een tabel in Van der Houwen [1968] p.104/105. $R(\infty)$ is de asymptotische convergentiesnelheid. In de laatste kolom staat de maximumnorm van het verschil tussen de berekende oplossing $u_{nn}^{\rightarrow}$ en de analytische oplossing in de roosterpunten. Deze extreem kleine getallen hangen samen met het feit, dat in dit speciale voorbeeld

de oplossingen van de differentiaal- en de differentievergelijking gelijk zijn (zie verder vraagstuk 4.9.1).

	anal. opl.	a	n	p	nn	q	$R^*(m)$	R(nn)	R(∞)	$\|\vec{u}_{nn}-U\|_\infty$
403	0	$2h^2$	50	0	50	-	.28	.27	.29	$.29_{10}-5$
404	0	$4h^2$	44	7	51	-	.34	.34	.41	$.22_{10}-6$
405	x^2y^2	$2h^2$	50	0	50	-	.29	.28	.29	niet berekend
406	x^2y^2	$4h^2$	44	7	51	-	.36	.34	.41	niet berekend
407	x^2y^2	$4.9h^2$	44	6	50	-	.41	.39	.45	$.24_{10}-6$
408	x^2y^2	$4h^2$	45*	7	52	4	.34	.34	.41	$.38_{10}-5$
409	x^2y^2	$4.9h^2$	35*	6	41	4	.40	.38	.45	$.11_{10}-4$
410	x^2y^2	$4.9h^2$	32*	6	38	3	.37	.38	.45	$.27_{10}-3$
411	x^2y^2	$4.9h^2$	37*	6	43	5	.41	.38	.45	$.35_{10}-5$

Tabel 4.1. Overzicht van numerieke resultaten

4.8.5. Het ALGOL 60 programma. Hieronder volgt de volledige tekst en output van het programma dat wij hebben gebruikt om bovenstaande resultaten te verkrijgen, zoals dit door de regeldrukker van de EL-X8 is afgedrukt. De procedures "richardson" en "elimination" vindt men op de regels 3 t/m 124. De procedure "residual" die wij als gebruiker van "richardson" en "elimination" zelf moesten schrijven, staat op de regels 126 t/m 142 afgedrukt. Op de regels 155 t/m 174 treft men twee procedures output aan, waarvan de tweede de afhankelijkheid van het aantal iteratieslagen van de bereikte nauwkeurigheid in de bepaling van de dominante eigenwaarde realiseert. De overige procedures zijn voor de lezer niet van belang, omdat ze maar een van de vele mogelijkheden vormen om een volledig ALGOL programma te realiseren.

```
1
2
3   BEGIN COMMENT A DIRICHLET PROBLEM FOR LAPLACE S EQUATION;
4
5     PROCEDURE RICHARDSON(U,LJ,UJ,LL,UL,INAP,RESIDUAL,A,B,N,DISCR,K,
6     RATECONV,DOMEIGVAL,OUTPUT); VALUE LJ,UJ,LL,UL,A,B;
7     INTEGER N,K,LJ,UJ,LL,UL; REAL A,B,RATECONV,DOMEIGVAL; BOOLEAN INAP;
8     ARRAY U,DISCR; PROCEDURE RESIDUAL,OUTPUT;
9     BEGIN INTEGER J,L; REAL X,Y,Z,Y0,C,D,ALFA,OMEGA,OMEGA0,
10      EIGMAX,EIGEUCL,EUCLRES,MAXRES,RCMAX,RCEUCL,MAXRES0,EUCLRES0;
11      ARRAY V,RES[LJ:UJ,LL:UL];
12      PROCEDURE CALPAR;
13      COMMENT CALPAR CALCULATES THE PARAMETERS ALFA AND OMEGA
14      OF EACH ITERATION;
15      BEGIN ALFA:= Z/(Z - ALFA);
16        OMEGA:= 1/(X - OMEGA * Y)
17      END CALPAR;
18      PROCEDURE ITERATION;
19      COMMENT FIRST THE ITERATION FORMULA IS CONSTRUCTED;
20      BEGIN REAL AUXV,AUXU,AUXRES,EUCLUV,MAXUV;
21        EUCLUV:= EUCLRES:= MAXUV:= MAXRES:= 0;
22        FOR J:= LJ STEP 1 UNTIL UJ DO
23        FOR L:= LL STEP 1 UNTIL UL DO RES[J,L]:= V[J,L];
24        RESIDUAL(RES);
25        FOR J:= LJ STEP 1 UNTIL UJ DO
26        FOR L:= LL STEP 1 UNTIL UL DO
27        BEGIN AUXV:= U[J,L]; AUXU:= V[J,L]; AUXRES:= RES[J,L];
28          AUXV:= ALFA * AUXU - OMEGA * AUXRES + (1 - ALFA) * AUXV;
29          V[J,L]:= AUXV; U[J,L]:= AUXU;
30          COMMENT THE NORMS OF THE K-TH RESIDUAL AND THE DIFFERENCE
31          BETWEEN THE (K+1)-TH AND K-TH ITERAND ARE CALCULATED;
32          AUXU:= ABS(AUXU - AUXV); AUXRES:= ABS(AUXRES);
33          MAXUV:= IF MAXUV < AUXU THEN AUXU ELSE MAXUV;
34          MAXRES:= IF MAXRES < AUXRES THEN AUXRES ELSE MAXRES;
35          EUCLUV:= EUCLUV + AUXU * AUXU;
36          EUCLRES:= EUCLRES + AUXRES * AUXRES;
37        END;
38        EUCLUV:= SQRT(EUCLUV); EUCLRES:= SQRT(EUCLRES);
39        DISCR[1]:= EUCLRES; DISCR[2]:= MAXRES;
40        COMMENT DOMEIGVAL IS EVALUATED;
41        MAXUV:= MAXRES/MAXUV; EUCLUV:= EUCLRES/EUCLUV;
42        EIGMAX:= MAXUV * (C - MAXUV)/(.25 * D - MAXUV);
43        EIGEUCL:= EUCLUV * (C - EUCLUV)/(.25 * D - EUCLUV);
44        DOMEIGVAL:= .5 * (EIGMAX + EIGEUCL);
45        COMMENT FINALLY THE RATE OF CONVERGENCE IS CALCULATED;
46        RCEUCL:= -LN(EUCLRES/EUCLRES0)/K;
47        RCMAX:= -LN(MAXRES/MAXRES0)/K;
48        RATECONV:= .5 * (RCEUCL + RCMAX)
49      END ITERATION;
50      COMMENT THE CONSTANTS FOR STARTING CALPAR ARE CALCULATED;
51      ALFA:= 2; OMEGA:= 4/(B + A); Y0:= (B + A)/(B - A);
52      X:= .5 * (B + A); Y:= (B - A) * (B - A)/16; Z:= 4 * Y0 * Y0;
53      COMMENT THE CONSTANTS NEEDED FOR DOMEIGVAL ARE CALCULATED;
54      C:= A * B; C:= SQRT(C); D:= SQRT(A) + SQRT(B); D:= D * D;
55      COMMENT THE INITIAL APPROXIMATION IS PUT INTO ARRAY U;
56      IF ¬INAP THEN
```

```
      BEGIN FOR J:= LJ STEP 1 UNTIL UJ DO
        FOR L:= LL STEP 1 UNTIL UL DO U[J,L]:= 1
      END;
      COMMENT THE ZEROTH ITERATION IS NOW PERFORMED;
      K:= 0;
      FOR J:= LJ STEP 1 UNTIL UJ DO
      FOR L:= LL STEP 1 UNTIL UL DO RES[J,L]:= U[J,L];
      RESIDUAL(RES);
      OMEGA0:= 2/(B+A);
      BEGIN REAL AUXRES0;
        MAXRES0:= EUCLRES0:= 0;
        FOR J:= LJ STEP 1 UNTIL UJ DO
        FOR L:= LL STEP 1 UNTIL UL DO
        BEGIN AUXRES0:= RES[J,L];
          V[J,L]:= U[J,L] - OMEGA0 * AUXRES0;
          AUXRES0:= ABS(AUXRES0);
          MAXRES0:= IF MAXRES0 < AUXRES0 THEN AUXRES0 ELSE MAXRES0;
          EUCLRES0:= EUCLRES0 + AUXRES0 * AUXRES0
        END;
      EUCLRES0:= SQRT(EUCLRES0)
      END;
      DISCR[1]:= EUCLRES0; DISCR[2]:= MAXRES0;
     OUTPUT(K);
      IF K ≥ N THEN GOTO FINALLY;
    NEXT STEP:
      K:= K + 1; CALPAR; ITERATION; OUTPUT(K);
      IF K < N THEN GOTO NEXT STEP;
      FINALLY:
    END RICHARDSON;

    PROCEDURE ELIMINATION(U,LJ,UJ,LL,UL,RESIDUAL,A,B,N,DISCR,K,RATECONV,
    DOMEIGVAL,OUTPUT); VALUE LJ,UJ,LL,UL,A,B; INTEGER LJ,UJ,LL,UL,N,K;
    REAL A,B,RATECONV,DOMEIGVAL; ARRAY U,DISCR;
    PROCEDURE RESIDUAL,OUTPUT;
    BEGIN REAL PI,AUXCOS,C,D;
      REAL PROCEDURE ARCCOS(X); VALUE X; REAL X;
      ARCCOS:= IF X ≥ 0 THEN ARCTAN(SQRT(1-X*X)/X) ELSE
               PI + ARCTAN(SQRT(1-X*X)/X);
      REAL PROCEDURE TANH(Z); VALUE Z; REAL Z;
      BEGIN REAL U,V; U:= EXP(Z); V:= EXP(-Z);
        TANH:= (U -V) / (U + V)
      END TANH;
      REAL PROCEDURE OPTPOL(X); VALUE X; REAL X;
      BEGIN REAL W,Y;
    L:  W:= (B + COS(.5*PI/X) + DOMEIGVAL) / (B - DOMEIGVAL);
        IF ABS(W) < 1 THEN
        BEGIN Y:= ARCCOS(W);
          OPTPOL:= 2 * SQRT(A/B) + SIN(X*Y) / COS(X*Y) *
          (Y - B*PI*SIN(.5*PI/X)*.5 / (X * (B-DOMEIGVAL) * SQRT(1-W*W)))
        END ELSE
        BEGIN IF ABS(W) > 1 THEN
          BEGIN Y:= LN(W + SQRT(W*W-1));
            OPTPOL:= 2 * SQRT(A/B) - TANH(X*Y) * (Y + B*PI*SIN(.5*PI/X)*
            .5/(X*(B-DOMEIGVAL)*SQRT(W*W-1)))
          END ELSE
          BEGIN X:= X + .2; GOTO L END
        END;
      END OPTPOL;
      PI:= 4 * ARCTAN(1);
      C:= 1;
```

```
117       IF OPTPOL(C) < 0 THEN
118      BEGIN D:= ,5 * PI * SQRT(B/DOMEIGVAL);
119       M: D:= D + D;
120        IF ZEROIN(C,D,OPTPOL(C),C*10-3) THEN N:= ENTIER(C+,5)
121         ELSE GOTO M;
122       END ELSE N:= 1;
123       AUXCOS:= COS(,5*PI/N);
124       RICHARDSON(U,LJ,UJ,LL,UL,IDUE,RESIDUAL,(2*DOMEIGVAL + B*(AUXCOS-1))/(AUXCOS+1),
125        B,N,DISCR,K,RATECONV,     DOMEIGVAL,OUTPUT)
126     END ELIMINATION;
127
128     PROCEDURE RESIDUAL(U); ARRAY U;
129     BEGIN INTEGER UJMIN1,ULMIN1,LJPLUS1;
130     REAL U2; REAL ARRAY U1[LJ:UJ];
131       UJMIN1:= UJ - 1; ULMIN1 := UL - 1; LJPLUS1:= LJ + 1;
132       FOR J:= LJ STEP 1 UNTIL UJ DO
133       BEGIN U1[J]:= U[J,LL]; U[J,LL]:= 0; END;
134       FOR L:= LL + 1 STEP 1 UNTIL ULMIN1 DO
135       BEGIN  U1[LJ]:= U[LJ,L]; U[LJ,L]:= 0;
136         FOR  J:= LJPLUS1STEP 1 UNTIL UJMIN1 DO
137         BEGIN U2:= U[J,L];
138           U[J,L]:=(4 * U2 - U1[J-1]  - U1[J] - U[J+1,L] - U[J,L+1])- F(J*H,L*H)*H2;
139           U1[J]:= U2
140         END;
141         U[UJ,L]:= 0;
142       END;
143       FOR J:= LJ STEP 1 UNTIL UJ DO U[J,UL]:= 0
144     END RESIDUAL;
145
146     REAL PROCEDURE F(X,Y); VALUE X,Y; REAL X,Y;
147     F:= -2*(X*X + Y*Y);
148
149     REAL PROCEDURE ANALSOL(X,Y); VALUE X,Y; REAL X,Y;
150     ANALSOL:= X*X*Y*Y;
151
152     PROCEDURE INITAPPR(U,J,L,G); INTEGER J,L; ARRAY U; REAL G;
153     FOR J:= LJ STEP 1 UNTIL UJ DO
154       FOR L:= LL STEP 1 UNTIL UL DO
155       U[J,L]:= IF J=LJ OR J=UJ OR L=LL OR L=UL THENG ELSE 1;
156
157     PROCEDURE OUTPUT1(K); VALUE K; INTEGER K;
158     BEGIN INTEGER I;
159       IF K = 0 THEN
160       BEGIN CARRIAGE(3);
161         PRINTTEXT(¢ K     DISCR[1]       DISCR[2]       RATECONV       DOMEIGVAL$);
162         NLCR
163       END;
164       NLCR; FIXT(3,0,K); FOR I:= 1,2 DO FLOT(6,2,DISCR[I]);
165       IF K > 0 THEN
166       BEGIN  FLOT(7,2,RATECONV); FLOT(7,2,DOMEIGVAL) END;
167     END OUTPUT1;
168
169     PROCEDURE OUTPUT2(K); VALUE K; INTEGER K;
170     BEGIN
171       IF K = 0 THEN D1:= D2:= 0 ELSE
172       BEGIN D2:= D1; D1:= DOMEIGVAL;
173         N:= IFABS((D1 - D2)/D2) < 10↓(-Q) THEN K ELSE NN
174       END;
175       OUTPUT1(K)
176     END OUTPUT2;
```

```
177
178     EPROCEDURE OPRI(X,S); SIBLING S;
179     BEGIN NLCR; PRINT(X); PRINTTEXT(S) END;
180
181     EPROCEDURE DATINOUT(X,S); SIBLING S;
182     BEGIN X:= READ; OPRI(X,S) END;
183
184     EPROCEDURE OPAR22(NF,J,L,LJ,UJ,LL,UL,TEXT); VALUE LJ,UJ,LL,UL;
185     INTEGER LJ,UJ,LL,UL,J,L  ; REAL  NF; SIBLING TEXT;
186     BEGIN INTEGER NUMJ,NUML;
187       EPROCEDURE HORVER(HI,VI,LHI,UHI,LVI,UVI,SH,SV);
188       VALUE LHI,UHI,LVI,UVI; INTEGER HI,VI,LHI,UHI,LVI,UVI;
189       SIBLING SH,SV;
190       BEGIN INTEGER HNUM,S,DEC,R,P;
191         S:= 0; HNUM:= UHI - LHI + 1; NLCR;
192       COARSER: S:= S + 1;
193         DEC:= ENTIER(139/HNUM * S - 7); IF DEC ≤ 3 THEN GOTOCOARSER;
194         IF DEC > 13 THEN DEC:= 13;
195         SPACE(3); PRINTTEXT(SH); PRINTTEXT(¢|¢); FIXT(2,0,LHI);
196         FOR HI:= LHI + S STEP S UNTIL UHI DO
197         BEGIN SPACE(DEC+3); FIXT(2,0,HI) END;
198         NLCR; PRSYM(126); PRINTTEXT(SV); PRINTTEXT(¢___1¢);
199         FOR HI:= 1 STEP 1 UNTIL 139 DO PRINTTEXT(¢_¢);
200         SPACE(4); PRINTTEXT(¢|¢);
201         FOR VI:= LVI STEP S UNTIL UVI DO
202         BEGIN NLCR; FIXT(2,0,VI); PRINTTEXT(¢|¢);
203           FOR HI:= LHI STEP S UNTIL UHI DOPLOT(DEC,2,NF);
204         END
205       END HORVER;
206       NUMJ:= UJ - LJ + 1; NUML:= UL - LL + 1;
207       IF LINE NUMBER > 50 - NUMJ ^ LINENUMBER > 50 - NUML
208       THEN NEWPAGE ELSE CARRIAGE(4);
209       PRINTTEXT(TEXT);
210       IF NUMJ > NUML THEN HORVER(L,J,LL,UL,LJ,UJ,¢L¢,¢J¢)
211       ELSE HORVER(J,L,LJ,UJ,LL,UL,¢J¢,¢L¢)
212     END OPAR22;
213
214     INTEGER J,L,LJ,UJ,LL,UL,NN,N,P,K,Q;
215     REAL M,PI,D1,D2,H2,RATECONVR,RATECONVE,DOMEIGVAL,RATECONV,A,B;
216     REAL ARRAY DISCR[1:2];
217     DATINOUT(LJ,¢LJ¢); DATINOUT(UJ,¢UJ¢); DATINOUT(LL,¢LL¢); DATINOUT(UL,¢UL¢);
218     DATINOUT(N ,¢N ¢); DATINOUT(Q,¢Q¢); DATINOUT(A,¢A¢); DATINOUT(B,¢B¢);
219
220     BEGIN REAL ARRAY U[LJ:UJ,LL:UL];
221       PI:=3.1415 92653 58979; M:= PI/(UJ - LJ     ); H2:= M * M; P:= 0;
222       INITAPPR(U,J,L,ANALSOL(J*M,L*M));
223       NN:= N;
224       RICHARDSON(U,LJ,UJ,LL,UL,TRUE,RESIDUAL,A,B,N,DISCR,K,
225       RATECONV ,DOMEIGVAL,OUTPUT2); RATECONVR:= RATECONV;
226       ELIMINATION(U,LJ,UJ,LL,UL,RESIDUAL,A ,B,P,DISCR,K,
227       RATECONV ,DOMEIGVAL,OUTPUT1); RATECONVE:= RATECONV;
228       NLCR; OPRI(-4 *(-1 + COS(M)),¢SMALLEST EIGENVALUE, FROM ANALYTIC FORMULA¢);
229       NN:= N + P;
230       NLCR; OPRI(NN, ¢TOTAL NUMBER OF ITERATIONS¢);
231       OPRI((N  * RATECONVR + P * RATECONVE)/NN, ¢RATE OF CONVERGENCE WITH RESPECT TO THE ZEROTH ITERAND OF RICHARDSON¢);
232       OPRI(2*SQRT(A/B)*(1-P/NN)- LN(2)/NN, ¢ THEOR RATE  CONV  (+ LN(SIGMA(ELIMIN OF / NN)))¢);
233       OPAR22(U[J,L] - ANALSOL(J*M,L*M),J,L,LJ,UJ,LL,UL,
234       ¢DIFFERENCE BETWEEN CALCULATED AND ANALYTIC SOLUTION¢);
235     END
236   END
```

```
          +0        LJ
         +11        UJ
          +0        LL
         +11        UL
         +50        N
          +4        Q
+.32600000000000₁₀-  0 A
+.78300000000002₁₀+  1 B
```

K	DISCR[1]	DISCR[2]	RATECONV	DOMEIGVAL
+0	+.204406₁₀+ 3	+.156345₁₀+ 3		
+1	+.100476₁₀+ 3	+.508460₁₀+ 2	+.9167294₁₀- 0	+.2621073₁₀- 0
+2	+.103373₁₀+ 3	+.703953₁₀+ 2	+.3699275₁₀- 0	-.4244396₁₀+ 1
+3	+.786671₁₀+ 2	+.360373₁₀+ 2	+.4037331₁₀- 0	-.2716363₁₀+ 1
+4	+.534238₁₀+ 2	+.355258₁₀+ 2	+.3529579₁₀- 0	-.2147710₁₀+ 2
+5	+.366002₁₀+ 2	+.171509₁₀+ 2	+.3930074₁₀- 0	-.4078745₁₀+ 1
+6	+.243742₁₀+ 2	+.148305₁₀+ 2	+.3734976₁₀- 0	-.2183809₁₀+ 2
+7	+.164317₁₀+ 2	+.681899₁₀+ 1	+.4038038₁₀- 0	-.2123047₁₀+ 1
+8	+.112547₁₀+ 2	+.539277₁₀+ 1	+.3916459₁₀- 0	-.3887923₁₀+ 1
+9	+.784873₁₀+ 1	+.237861₁₀+ 1	+.4136284₁₀- 0	-.5860083₁₀- 0
+10	+.554476₁₀+ 1	+.195974₁₀+ 1	+.3993257₁₀- 0	-.6862497₁₀- 0
+11	+.414666₁₀+ 1	+.109338₁₀+ 1	+.4027547₁₀- 0	-.1812009₁₀- 0
+12	+.324981₁₀+ 1	+.106386₁₀+ 1	+.3804867₁₀- 0	-.1209331₁₀+ 1
+13	+.264812₁₀+ 1	+.643147₁₀- 0	+.3784504₁₀- 0	-.2312218₁₀- 0
+14	+.221553₁₀+ 1	+.575793₁₀- 0	+.3617391₁₀- 0	-.8111544₁₀- 0
+15	+.188292₁₀+ 1	+.401828₁₀- 0	+.3550363₁₀- 0	+.2781859₁₀- 1
+16	+.163814₁₀+ 1	+.347354₁₀- 0	+.3417508₁₀- 0	-.6912841₁₀- 1
+17	+.143762₁₀+ 1	+.265942₁₀- 0	+.3333432₁₀- 0	+.1286752₁₀- 0
+18	+.126357₁₀+ 1	+.254913₁₀- 0	+.3195854₁₀- 0	+.9006131₁₀- 1
+19	+.111717₁₀+ 1	+.218179₁₀- 0	+.3101005₁₀- 0	+.1436005₁₀- 0
+20	+.990569₁₀- 0	+.184138₁₀- 0	+.3018431₁₀- 0	+.1595131₁₀- 0
+21	+.877204₁₀- 0	+.162472₁₀- 0	+.2933439₁₀- 0	+.1537639₁₀- 0
+22	+.778258₁₀- 0	+.140459₁₀- 0	+.2860389₁₀- 0	+.1641562₁₀- 0
+23	+.690415₁₀- 0	+.121933₁₀- 0	+.2792809₁₀- 0	+.1686954₁₀- 0
+24	+.612635₁₀- 0	+.108160₁₀- 0	+.2726314₁₀- 0	+.1651442₁₀- 0
+25	+.543668₁₀- 0	+.963246₁₀- 1	+.2664325₁₀- 0	+.1618803₁₀- 0
+26	+.482489₁₀- 0	+.867391₁₀- 1	+.2604967₁₀- 0	+.1557614₁₀- 0
+27	+.428202₁₀- 0	+.768444₁₀- 1	+.2553021₁₀- 0	+.1595197₁₀- 0
+28	+.380029₁₀- 0	+.680682₁₀- 1	+.2504809₁₀- 0	+.1594547₁₀- 0
+29	+.337279₁₀- 0	+.602594₁₀- 1	+.2460020₁₀- 0	+.1614941₁₀- 0
+30	+.299341₁₀- 0	+.533693₁₀- 1	+.2418145₁₀- 0	+.1612161₁₀- 0
+31	+.265671₁₀- 0	+.474211₁₀- 1	+.2378446₁₀- 0	+.1610763₁₀- 0
+32	+.235788₁₀- 0	+.421076₁₀- 1	+.2341333₁₀- 0	+.1611642₁₀- 0
+33	+.209266₁₀- 0	+.373803₁₀- 1	+.2306506₁₀- 0	+.1614001₁₀- 0
+34	+.185728₁₀- 0	+.330999₁₀- 1	+.2274100₁₀- 0	+.1623334₁₀- 0
+35	+.164837₁₀- 0	+.293728₁₀- 1	+.2243238₁₀- 0	+.1620304₁₀- 0
+36	+.146296₁₀- 0	+.260597₁₀- 1	+.2214120₁₀- 0	+.1621299₁₀- 0
+37	+.129841₁₀- 0	+.231261₁₀- 1	+.2186543₁₀- 0	+.1620965₁₀- 0
+38	+.115236₁₀- 0	+.205239₁₀- 1	+.2160410₁₀- 0	+.1620758₁₀- 0
+39	+.102274₁₀- 0	+.182174₁₀- 1	+.2135597₁₀- 0	+.1619961₁₀- 0
+40	+.907706₁₀- 1	+.161704₁₀- 1	+.2112022₁₀- 0	+.1619721₁₀- 0
+41	+.805608₁₀- 1	+.143518₁₀- 1	+.2089610₁₀- 0	+.1619948₁₀- 0
+42	+.714993₁₀- 1	+.127386₁₀- 1	+.2068258₁₀- 0	+.1619534₁₀- 0
+43	+.634571₁₀- 1	+.113054₁₀- 1	+.2047912₁₀- 0	+.1620004₁₀- 0
+44	+.563194₁₀- 1	+.100326₁₀- 1	+.2028501₁₀- 0	+.1620480₁₀- 0
+45	+.499846₁₀- 1	+.890386₁₀- 2	+.2009943₁₀- 0	+.1620445₁₀- 0

K	DISCR[1]	DISCR[2]	RATECONV	DOMEIGVAL
+0	+.499846₁₀- 1	+.890386₁₀- 2		
+1	+.479329₁₀- 1	+.853856₁₀- 2	+.4190317₁₀- 1	-.8930285₁₀- 0
+2	+.421992₁₀- 1	+.751746₁₀- 2	+.8464260₁₀- 1	-.6043001₁₀- 1
+3	+.338859₁₀- 1	+.603657₁₀- 2	+.1295611₁₀- 0	+.4408381₁₀- 1
+4	+.243905₁₀- 1	+.434488₁₀- 2	+.1793761₁₀- 0	+.6412276₁₀- 1
+5	+.150115₁₀- 1	+.267393₁₀- 2	+.2405832₁₀- 0	+.6022018₁₀- 1
+6	+.670250₁₀- 2	+.119379₁₀- 2	+.3348824₁₀- 0	+.4110752₁₀- 1
+7	+.356356₁₀- 5	+.671599₁₀- 6	+.1360075₁₀+ 1	+.3697520₁₀- 4

+.1620281055439₁₀- 0 SMALLEST EIGENVALUE, FROM ANALYTIC FORMULA

+52 TOTAL NUMBER OF ITERATIONS

+.3570243574441₁₀- 0 RATE OF CONVERGENCE WITH RESPECT TO THE ZEROTH ITERAND OF RICHARDSON

+.3398266495724₁₀- 0 THEOR RATE CONV (+ LN(SIGMA(ELIMIN OP / NN)))

DIFFERENCE BETWEEN CALCULATED AND ANALYTIC SOLUTION

J	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+11
+0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0	-.0000₁₀- 0
+1	-.0000₁₀- 0	-.2549₁₀- 6	-.5037₁₀- 6	-.7300₁₀- 6	-.8781₁₀- 6	-.9910₁₀- 6	-.1020₁₀- 5	-.9519₁₀- 6	-.8346₁₀- 6	-.5947₁₀- 6	-.2711₁₀- 6	-.1728₁₀-10
+2	-.0000₁₀- 0	-.5037₁₀- 6	-.1014₁₀- 5	-.1463₁₀- 5	-.1774₁₀- 5	-.1971₁₀- 5	-.2019₁₀- 5	-.1860₁₀- 5	-.1615₁₀- 5	-.1172₁₀- 5	-.5468₁₀- 6	-.6912₁₀-10
+3	-.0000₁₀- 0	-.7300₁₀- 6	-.1463₁₀- 5	-.2102₁₀- 5	-.2546₁₀- 5	-.2814₁₀- 5	-.2869₁₀- 5	-.2661₁₀- 5	-.2324₁₀- 5	-.1698₁₀- 5	-.8015₁₀- 6	-.1673₁₀- 9
+4	-.0000₁₀- 0	-.8781₁₀- 6	-.1774₁₀- 5	-.2546₁₀- 5	-.3069₁₀- 5	-.3362₁₀- 5	-.3411₁₀- 5	-.3191₁₀- 5	-.2799₁₀- 5	-.2060₁₀- 5	-.1020₁₀- 5	-.2765₁₀- 9
+5	-.0000₁₀- 0	-.9910₁₀- 6	-.1971₁₀- 5	-.2814₁₀- 5	-.3362₁₀- 5	-.3647₁₀- 5	-.3689₁₀- 5	-.3472₁₀- 5	-.3006₁₀- 5	-.2212₁₀- 5	-.1131₁₀- 5	-.6985₁₀- 9
+6	-.0000₁₀- 0	-.1020₁₀- 5	-.2019₁₀- 5	-.2869₁₀- 5	-.3411₁₀- 5	-.3689₁₀- 5	-.3758₁₀- 5	-.3491₁₀- 5	-.2960₁₀- 5	-.2169₁₀- 5	-.1124₁₀- 5	-.6694₁₀- 9
+7	-.0000₁₀- 0	-.9519₁₀- 6	-.1860₁₀- 5	-.2661₁₀- 5	-.3191₁₀- 5	-.3472₁₀- 5	-.3491₁₀- 5	-.3202₁₀- 5	-.2686₁₀- 5	-.1997₁₀- 5	-.1065₁₀- 5	-.9895₁₀- 9
+8	-.0000₁₀- 0	-.8346₁₀- 6	-.1615₁₀- 5	-.2324₁₀- 5	-.2799₁₀- 5	-.3006₁₀- 5	-.2960₁₀- 5	-.2686₁₀- 5	-.2256₁₀- 5	-.1659₁₀- 5	-.8799₁₀- 6	-.1106₁₀- 8
+9	-.0000₁₀- 0	-.5947₁₀- 6	-.1172₁₀- 5	-.1698₁₀- 5	-.2060₁₀- 5	-.2212₁₀- 5	-.2168₁₀- 5	-.1997₁₀- 5	-.1659₁₀- 5	-.1152₁₀- 5	-.5740₁₀- 6	-.1513₁₀- 8
+10	-.0000₁₀- 0	-.2711₁₀- 6	-.5467₁₀- 6	-.8014₁₀- 6	-.1019₁₀- 5	-.1131₁₀- 5	-.1124₁₀- 5	-.1064₁₀- 5	-.8796₁₀- 6	-.5739₁₀- 6	-.2873₁₀- 6	-.2794₁₀- 8
+11	-.0000₁₀- 0	-.6366₁₀-11	-.2547₁₀-10	-.1164₁₀- 9	-.1019₁₀- 9	-.2619₁₀- 9	-.4657₁₀- 9	-.9895₁₀- 9	-.4075₁₀- 9	-.1513₁₀- 8	-.1048₁₀- 8	-.1164₁₀- 8

4.9. Opgaven

4.9.1. (Hildebrand [1968]). Neem aan dat Δ_h^+ en $\Delta_h^{(9)}$ gediscretiseerd zijn t.o.v. een vierkant rooster met maaswijdte h. Laat zien dat dan de volgende beweringen waar zijn.

(a) Als $U = e^{-x}\sin y$, dan is $\Delta U = 0$ en

$$\Delta_h^+ U - \Delta U = \frac{h^2}{6} U + O(h^4) ,$$
$$\Delta_h^{(9)} U - \Delta U = \frac{h^6}{1008} U + O(h^8) .$$

(b) Als $U = \sin x \sin y$, dan is $\Delta U = -2U$ en

$$\Delta_h^+ U - \Delta U = \frac{h^2}{6} U + O(h^4) ,$$
$$\Delta_h^{(9)} U - \Delta U = \frac{h^2}{3} U + O(h^4) .$$

(c) Als $U = x^2y^2$, dan is $\Delta U = 2(x^2+y^2)$ en

$$\Delta_h^+ U - \Delta U = 0$$
$$\Delta_h^{(9)} U - \Delta U = \frac{2}{3} h^2 .$$

Aanwijzing. Maak gebruik van stelling 4.2.5. Merk op dat het gebruik van de negenpuntsformule alleen zin heeft voor de vergelijking $\Delta U = 0$, en dat hij voor niet homogene vergelijkingen zelfs slechter kan zijn dan de vijfpunts + formule.

4.9.2. In het uitgewerkte voorbeeld in de vorige paragraaf hadden wij juist als analytische oplossing van het Dirichletprobleem in een vierkant met niethomogene randvoorwaarden $U = x^2y^2$. Stel twee Dirichletproblemen in een vierkant op die precies de analytische oplossingen (a) en (b) uit vraagstuk 4.9.1 hebben, en behandel deze numeriek op dezelfde manier als in de vorige paragraaf reeds met (c) is gebeurd. Vergelijk de numerieke uitkomsten en verklaar het verschil.

4.9.3. (Hildebrand [1968]). (a) Toon aan dat als $\Delta U = 0$ en U een polynoom is van graad drie of lager in x en y, U dan ook exact voldoet aan de vergelijking $\Delta_h^+ U = 0$.

(b) Illustreer het resultaat van (a) numeriek door de oplossing van het probleem

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0 \quad , \qquad 0<x<1 \ , \quad 0<y<1 \ ,$$

$$U(0,y) = 0 \ , \quad U(1,y) = 1-3y^2 \ , \quad 0<y<1 \ ,$$
$$U(x,0) = x^3, \quad U(x,1) = x^3-3x \ , \quad 0<x<1 \ ,$$

te benaderen met behulp van de differentie-operator Δ_h^+ met h = .1, en deze te vergelijken met de analytische oplossing $U = x^3 - 3xy^2$ in de roosterpunten.

4.9.4. Gegeven is het randwaardeprobleem

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} - 10U = 0 \ , \quad -\pi<x,y<\pi$$

$$U(x,\pi) = 1 \ , \qquad -\pi<x<\pi \ ,$$
$$U(x,-\pi) = 0 \ , \qquad -\pi<x<\pi \ ,$$
$$\frac{\partial U}{\partial x}(\pi,y) = \sin y \ , \quad -\pi<y<\pi \ ,$$
$$\frac{\partial U}{\partial x}(-\pi,y) = \cos y \ , \quad -\pi<y<\pi \ .$$

Zoek hiervan de oplossing m.b.v. de in dit hoofdstuk gegeven procedures. Kies bijvoobeeld $\xi=\eta=\pi/20$. (De discretisatie van de normale afgeleiden is behandeld in paragraaf 4.2)

4.9.5. Beschouw het volgende randwaardeprobleem in het gebied $\Omega = \{(x,y) \mid 1 < x^2 + y^2 < 4\}$:

$$(4.93) \qquad \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0, \qquad 1<x^2+y^2<4 \ ,$$

$$(4.94) \qquad U = U_1 \ , \quad x^2+y^2=1 \ , \qquad y > 0 \ ,$$
$$U = U_2 \ , \quad x^2+y^2=1 \ , \qquad y < 0 \ ,$$

$$(4.95) \qquad \frac{\partial U}{\partial \nu} = H(U-U_0) \ , \quad x^2+y^2=4 \ ,$$

met H, U_0, U_1, U_2 gegeven constanten. De randvoorwaarde (4.95) is een voorbeeld van de *stralingsvoorwaarde van Newton* (hier is $\partial/\partial\nu$ de afgeleide langs de naar binnen gerichte normaal).

Men kan (4.93) t/m (4.95) opvatten als een mathematisch model voor de stationaire temperatuurverdeling in een oneindig lange ringvormige pijp waarvan het ringvormige gebied uit figuur 4.10 de loodrechte doorsnede is, die aan de binnenkant boven resp. onder op constante temperaturen U_1 resp. U_2 wordt gehouden, en die aan de buitenrand warmte in de omgeving, die een temperatuur U_0 heeft, uitstraalt; H is hier een stralingsconstante.

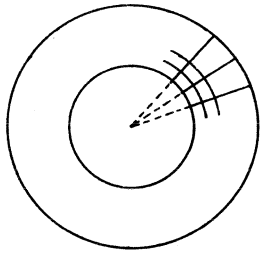


fig. 4.10

Het ligt voor de hand om (4.93) t/m (4.95) te schrijven in poolcoördinaten r,θ ($x=r\cos\theta$, $y=r\sin\theta$):

$$(4.93') \qquad \frac{\partial^2 U}{\partial r^2} + \frac{1}{r}\frac{\partial U}{\partial r} + \frac{1}{r^2}\frac{\partial^2 U}{\partial \theta^2} = 0 , \qquad 1<r<2,\ -\pi\le\theta\le\pi ,$$

$$(4.94') \qquad U(1,\theta) = U_1 , \qquad 0<\theta<\pi ,$$
$$U(1,\theta) = U_2 , \qquad -\pi<\theta<0 ,$$

$$(4.95') \qquad \frac{\partial U}{\partial r}(2,\theta) = H(U(2,\theta)-U_0) , \qquad -\pi\le\theta\le\pi .$$

Leg nu over het gebied Ω een rooster waarvan de roosterpunten gegeven worden door de doorsnijdingen van cirkels $r = j\rho + 1$ en rechte lijnen $\theta = l\alpha$ (ρ en α constantes, en α een "nette" fractie van 2π). Discretiseer de operator van Laplace gegeven in poolcoördinaten t.o.v. dit rooster zodanig dat de fout $O(\rho^2)$ is. Benader de oplossing van het zo te verkrijgen discrete analogon van (4.93') t/m (4.95') m.b.v. de in dit hoofdstuk beschreven procedures. Kies daarbij bijvoorbeeld $U_0 = 0$, $U_1 = 1$, $U_2 = 2$ en $H = .05$.

4.9.6. (a) Laat zien dat de differentie-operator

$$(\Delta_h^+)^2 = h^{-4}\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & -8 & 2 & 0 \\ 1 & -8 & 20 & -8 & 1 \\ 0 & 2 & -8 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

t.o.v. een vierkant rooster de biharmonische operator Δ^2 benadert op een term van de orde h^2 na, dus

$$(\Delta_h^+)^2 U - \Delta^2 U = O(h^2) \quad .$$

(b) Een overal even dunne vierkante elastische plaat wordt begrensd door $x = 0$, $x = L$, $y = 0$ en $y = L$. Als er geen uitwendige kracht op wordt uitgeoefend, dan voldoet een kleine uitwijking U uit de evenwichtstoestand aan $\Delta^2 U = 0$. Veronderstel dat de plaat zonder vervorming ingeklemd is langs de drie randen $x = 0$, $x = L$ en $y = L$, zodat $U = 0$ en $\partial U/\partial \nu = 0$ geldt langs deze randen, en dat de plaat langs $y = 0$ in een parabolische vorm is ingeklemd, zodanig dat daar geldt

$$U = \frac{x}{L}\left(1-\frac{x}{L}\right) ,$$

$$\frac{\partial U}{\partial y} = 0 \quad .$$

Leg over het gebied $0<x, y<L$ een vierkant rooster met maaswijdte h, en gebruik de in (a) gegeven differentie-operator als benadering van Δ^2 om numeriek de waarden van U in de roosterpunten te benaderen.

4.9.7. Een membraan is gespannen in een L-vormig raam, zodanig dat het het gebied getekend in figuur 4.11 overspant. De eigenfrequenties λ van dit membraan worden gevonden uit

$$-\Delta U = \lambda^2 U$$

onder de randvoorwaarde

$$U = 0 \text{ op de rand,}$$

waarin U dan de bij de eigenwaarde λ^2 behorende eigenfunctie is. Deze eigenfrequenties zijn voor deze geometrie niet op eenvoudige analytische manier te vinden. Probeer een benadering van de laagste te vinden met behulp van de procedure "richardson".

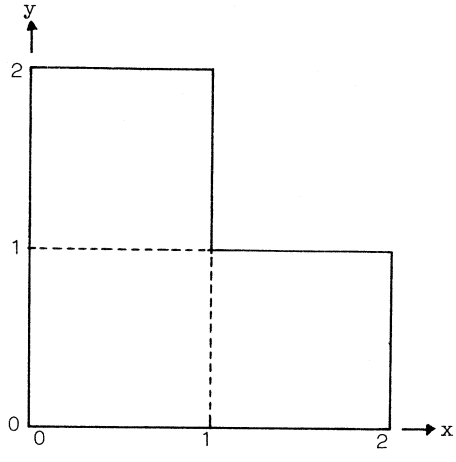


fig. 4.11

Literatuur

BRINK, W.P. van den, COOLEN, T.M.T., e.a. [1970], *Colloquium elliptische differentiaalvergelijkingen*, deel 2, MC Syllabus 9.2, Mathematisch Centrum, Amsterdam.

COOLEN, T.M.T., FÖRCH, G.J.R. e.a. [1969], *Colloquium elliptische differentiaalvergelijkingen*, deel 1, MC Syllabus 9.1, Mathematisch Centrum. Amsterdam.

DEKKER, T.J. [1968], *ALGOL 60 procedures in numerical algebra, 1*, MC Tracts 22, Mathematisch Centrum, Amsterdam.

DEKKER, T.J. & HOFFMANN, W. [1968], *ALGOL 60 procedures in numerical algebra, 2*, MC Tracts 23, Mathematisch Centrum, Amsterdam.

FLANDERS, D.A. & SHORTLEY, G. [1950], *Numerical determination of fundamental modes*, J. Appl. Phys., 21 (1950) 1326-1332.

FORSYTHE, G.E. & WASOW, W.R. [1960], *Finite difference methods for partial differential equations*, John Wiley & Sons Inc., New York.

HILDEBRAND, F.B. [1968], *Finite-difference equations and simulations*, Prentice Hall Inc., Englewood Cliffs, N.J.

HOUWEN, P.J. van der [1967], *On the acceleration of Richardson's method I, theoretical part*, Rapport TW 104/67, Mathematisch Centrum, Amsterdam.

HOUWEN, P.J. van der [1968], *Finite difference methods for solving partial differential equations*, MC Tracts 20, Mathematisch Centrum, Amsterdam.

RICHARDSON, L.F. [1910], *The approximate arithmetical solution by finite differences of physical problems involving differential equations*, Philos. Trans. Roy. Soc. London A, 210 (1910) 307-357.

VARGA, R.S. [1962], *Matrix iterative analysis*, Prentice-Hall Inc., Englewood Cliffs, N.J.

YOUNG, D.M. [1953], *On Richardson's method for solving linear systems with positive definite matrices*, J. Math. Phys. 32 (1953) 243-255.

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 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*, 1974. ISBN 90 6196 103 3.

* MCS 22 R. HELMERS, F.H. RUYMGAART, M.C.A. VAN ZUYLEN & J. OOSTERHOFF, *Asymptotische methoden in de statistiek*, 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*, 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.

De met een * gemerkte uitgaven moeten nog verschijnen.