

TW

**stichting
mathematisch
centrum**



TW

AFDELING TOEGEPASTE WISKUNDE

TN 63/72

APRIL

R. PLOEGER
EEN ALGOL 60 VERSIE VAN EEN VERSNELLING VAN DE
METHODE VAN RICHARDSON

2e boerhaavestraat 49 amsterdam

BIBLIOTHEEK MATHEMATISCH CENTRUM
AMSTERDAM

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

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

Inhoud

0. Inleiding	1
1. Reductiemethode	2
1.1. Methode van Richardson	2
1.2. Procedure richardson, heading en betekenis van de parameters	2
1.3. Bespreking van de subprocedures	4
2. Eliminatiemethode	6
2.1. Versnelling van de methode van Richardson	6
2.2. Procedure elimination, heading en betekenis van de parameters	6
2.3. Bespreking van de subprocedures	7
3. Opmerkingen	9
4. Experimenten	11
4.1. Dirichletprobleem voor de snaar	11
4.2. Dirichletprobleem voor het membraan	13
4.3. Matrixprobleem uit de rheologie	17
5. ALGOL 60 tekst	22
5.1. procedure richardson (ALGOL 60 tekst)	22
5.2. procedure elimination (ALGOL 60 tekst)	24
Literatuur	25

0. Inleiding

Er bestaan verschillende iteratieve methoden om partiële differentiaalvergelijkingen van het elliptische type op te lossen. Een ervan is de methode van Richardson die geschikt is om matrixvergelijkingen $Lu = g$ op te lossen, waarin L een symmetrische positief definitieve operator is (zie [1] en [2]).

De methode berust op reductie van de componenten van de eigenfuncties van L waaruit de beginbenadering u_0 van $Lu = g$ opgebouwd is.

In hoofdstuk 1 geven we een ALGOL 60 versie van deze methode in de vorm van de procedure richardson. Heading en betekenis van de parameters komen aan de orde, evenals de subprocedures.

De methode van Richardson kan versneld worden; deze versnelling berust op reductie van de componenten van de laatste eigenfuncties van L waaruit de beginbenadering u_0 is opgebouwd, gevolgd door eliminatie van de componenten van de overblijvende eigenfuncties (zie [1] en [2]).

In hoofdstuk 2 wordt een ALGOL 60 versie gegeven van dit eliminatieproces in de vorm van de procedure elimination; voor de reductie kan richardson gebruikt worden.

Hoofdstuk 3 bevat een tweetal opmerkingen over de procedure richardson, de eerste opmerking betreft het aantal iteratieslagen en de tweede betreft eventuele besparing van geheugenruimte.

Hoofdstuk 4 bestaat uit de resultaten van een drietal experimenten om beide procedures te testen, te weten:

- 1°. het Dirichletprobleem voor de snaar op het interval $[0, \pi]$;
- 2°. het Dirichletprobleem voor het membraan op het vierkant $[0, \pi; 0, \pi]$;
- 3°. een probleem uit de rheologie.

Ten slotte wordt in hoofdstuk 5 de ALGOL 60 tekst gegeven van de procedures richardson en elimination.

1. Reductiemethode

1.1. Methode van Richardson

De methode van Richardson wordt gebruikt om iteratief matrixvergelijkingen op te lossen van het type

$$Lu = g,$$

waarin L een symmetrische matrix is met positieve eigenwaarden. In toepassingen van deze methode moeten de kleinste eigenwaarde van L , notatie λ_1 , en de grootste, notatie $\sigma(L)$, voorgeschreven worden (zie [1], [2]). De (asymptotische) convergentiesnelheid is gelijk aan $2\sqrt{\lambda_1}/\sigma(L)$.

De procedure richardson die nu volgt, is een ALGOL 60 versie van de methode van Richardson.

1.2. Procedure richardson

Heading en betekenis van de parameters.

```
procedure richardson (u,lj,uj,ll,ul,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;
```

De actuele parameters die met de formele corresponderen, zijn:

u : twee dimensionaal array $u[lj:uj,ll:ul]$; in u staat na iedere iteratiestap de door de procedure berekende benadering van de oplossing van $Lu = g$.

lj,uj,ll,ul : <expression>;
 onder- en bovengrenzen van het array u .

$inap$: <boolean>;
 $inap := \underline{true}$, als de gebruiker een beginapproximatie op-
 geeft in het array u ;
 $inap := \underline{false}$, als beginapproximatie wordt een vector met
 componenten die alle de waarde 1 hebben,
 aangehouden.

$residual$: procedure die door de gebruiker gedeclareerd moet worden:
procedure $residual(u)$; array u ;
 <body>;
 na aanroep levert $residual$ het residu $Lu - g$ af in array
 u .

a,b : <expression>;
 a en b geven resp. de onder- en bovengrens aan van de
 eigenwaarden van L ; er moet gelden $a > 0$.

n : <expression>;
 n geeft het aantal iteratieslagen aan, dat richardson uit-
 voert.

$discr$: 1-dim. array $discr[1:2]$;
 $richardson$ levert in dit array na iedere iteratieslag de
 volgende waarden af:
 $discr[1]$: hierin $\|Lu-g\|_2$ met $\| \cdot \|_2$: euclidische norm,
 $discr[2]$: hierin $\|Lu-g\|_\infty$ met $\| \cdot \|_\infty$: maximum norm.

k : <variable>;
 Jensenparameter voor de procedure output; verder telt k
 het aantal iteratieslagen dat de procedure richardson uit-
 voert.

$rateconv$: <variable>;
 hierin staat na iedere iteratie de waarde van de conver-
 gentiesnelheid.

domeigval : <variable>;
na iedere iteratie levert de procedure hierin de waarde af
van de dominante eigenwaarde (mits aanwezig).

output : een procedure die door de gebruiker gedeclareerd moet worden:

```

procedure output (k); value k; integer k;
<body>;
output (k) levert de volgende waarden af:
k = 0 :  $u_1$  in array u;
         $||Lu_0 - g||_{2,\infty}$  in resp. discr [1] en discr [2];
0 < k ≤ n:  $u_{k+1}$  in array u;
         $||Lu_k - g||_{2,\infty}$  in resp. discr [1] en discr [2];
        rateconv en domeigval m.b.t.  $u_k$ ;
k = n+1 :  $u_{n+1}$  in array u;
         $||Lu_{n+1} - g||_{2,\infty}$  in resp. discr [1] en discr [2];
        rateconv en domeigval m.b.t.  $u_{n+1}$ .

```

1.3. Bespreking van de subprocedures

procedure calpar;

deze procedure berekent de coëfficiënten α_k en ω_k van het iteratieschema, door een recursierelatie.

procedure iteration;

iteration berekent in iedere iteratieslag:

1°. u_{k+1} volgens het schema:

$$u_{k+1} = \alpha_k u_k - \omega_k (Lu_k - g) + (1 - \alpha_k) u_{k-1}, \quad k > 0,$$

$$u_1 = \alpha_0 u_0 - \omega_0 (Lu_0 - g),$$

zie [2], formule (1.2), pag. 2.

2°. $||Lu_k - g||_{2,\infty}$;

3°. domeigval volgens:

$$\text{labda} = s(\sqrt{ab}-s)/(.25(\sqrt{a}+\sqrt{b})^2-s)$$

met

$$s = ||\text{Lu}_k - g|| / ||\text{u}_{k+1} - \text{u}_k||,$$

(zie o.a. [2]).

De formule levert in de $|| \cdot ||_2$ -norm labda 2 en in de $|| \cdot ||_\infty$ -norm labda ∞ ; nu wordt domeigval als volgt berekend:

$$\text{domeigval} = .5(\text{labda } 2 + \text{labda } \infty).$$

4°. rateconv volgens:

$$\text{rateconv} = \frac{-.5}{k} \left[\ln\left(\frac{||\text{Lu}_k - g||_2}{||\text{Lu}_0 - g||_2}\right) + \ln\left(\frac{||\text{Lu}_k - g||_\infty}{||\text{Lu}_0 - g||_\infty}\right) \right].$$

2. Eliminatiemethode

2.1. Versnelling van de methode van Richardson

Deze methode berust op de reductie van de laatste eigenfuncties van de operator L die corresponderen met eigenwaarden $\lambda_1 \in [a, b]$ met $a > 0$, $b \geq \sigma(L)$, gevolgd door eliminatie van de overblijvende eigenfuncties (die corresponderen met eigenwaarden $\lambda_j < a$).

Hierbij kiezen we de overblijvende eigenwaarden λ_j als nulpunten van een eliminatiepolynoom.

De (asymptotische) convergentiesnelheid is weer $2\sqrt{a/b}$, maar nu is $a > \lambda_1$: een versnelling t.o.v. Richardson met $\sqrt{a/\lambda_1}$, zie ook [1], [2].

De procedure elimination is een ALGOL 60 versie van dit eliminatieproces; voor de reductie kan b.v. de procedure richardson gebruikt worden.

2.2. Procedure elimination

Heading en betekenis van de parameters, voorzover afwijkend van die van de procedure richardson.

```
procedure elimination (u,lj,uj,ll,ul,residual,a,b,n,discr,k,rateconv,
                        domeigval,output);
```

```
value      lj, uj, ll, ul, b;
```

```
integer    lj, uj, ll, ul, n, k;
```

```
real       a, b, rateconv, domeigval;
```

```
array      u, discr;
```

```
procedure residual, output;
```

```
u           : 2-dim. array u[lj:uj, ll:ul]; hierin geeft de gebruiker
              een beginapproximatie.
```

```
a           : <variable>;
              er geldt:  $a > 0$ , en  $a > \text{domeigval}$ .
```

```
n           : <variable>;
              in n levert elimination de graad af van het eliminatie-
              polynoom.
```

domeigval : <expression>;
 domeigval heeft bij aanroep van elimination de waarde van
 de te elimineren eigenwaarde van L.

2.3. Bespreking van de subprocedures

real procedure arccos(w);

berekent de functiewaarden van arccos voor $w \in [-1, 1]$ met behulp
 van de procedure arctan(w).

real procedure tanh(w);

berekent de functiewaarden van tanh voor $w \in (-\infty, \infty)$ met behulp van
 de procedure exp(w).

real procedure optpol(x);

berekent de functiewaarden van $g = g(x)$, waar g de functie vertegenwoordigt zoals gegeven in [1], pag. 102, formules (3.19) en (3.20); het nulpunt van g bepaalt de graad van het beste eliminatiepolynoom, d.w.z. het polynoom dat de grootste convergentie levert.

Het nulpunt wordt bepaald met de procedure zeroin; hierbij dient een onder- en bovengrens opgegeven te worden van het nulpunt. Als bovengrens d is gekozen $d = m\pi \sqrt{(b/domeigval)}$ met $m = 1, 2, \dots$. Als ondergrens c is genomen $c = 1$, waarbij er van uit is gegaan dat het eliminatiepolynoom in ieder geval van de eerste graad is.

We onderscheiden twee gevallen:

1°. $g(c) \geq 0$: voor de graad n van het eliminatiepolynoom wordt dan $n = 1$ gekozen.

2°. $g(c) < 0$: zeroin zoekt een nulpunt in $[1, d]$ met $d = \pi \sqrt{(b/domeigval)}$; als zeroin geen nulpunt vindt, wordt d verdubbeld; zeroin zoekt vervolgens op het nieuwe interval $[1, 2d]$ een nulpunt. Als dit mislukt, dan wordt gezocht op $[1, 3d]$, etc. Dit proces berust op het feit, dat voor voldoende grote d geldt $g(d) > 0$. De graad van het polynoom wordt nu verkregen uit:

$n = \text{entier}(c+.5),$

zie [1], pag. 102.

Voor de werking van de procedures `zero` en `entier` wordt verwezen naar [7] resp. [8].

3. Opmerkingen

Voor de resultaten van de experimenten ter sprake komen, willen we een tweetal opmerkingen maken over de procedure richardson. De eerste opmerking zal gaan over het aantal iteratieslagen, en de tweede betreft een eventuele besparing van geheugenruimte.

Opmerking 1

Het is in het algemeen niet mogelijk het aantal iteraties te schatten dat de procedure richardson moet uitvoeren om een gewenste precisie te bereiken bij de berekening van de oplossing u van $Lu = g$.

De procedure richardson is zo geprogrammeerd dat het mogelijk is om het aantal iteraties n te bepalen in afhankelijkheid van:

- 1°. de convergentiesnelheid (rateconv);
- 2°. de norm van het residu (discr [1] en discr [2]);
- 3°. de dominante eigenwaarde (domeigval).

Voorbeelden

ad 1°. $n := \text{if rateconv} < 10^{-4} \text{ then } k \text{ else } 50;$

ad 2°. $n := \text{if discr [1]} < 10^{-3} \text{ then } k \text{ else } 80;$

ad 3°. dit geval is iets anders; de procedure richardson heeft nl. alleen de beschikking over de berekende waarde van domeigval, en niet over de voorgaande. In de procedure output kan de gebruiker twee variabelen d1 en d2 declareren, waarmee de vorige waarde van domeigval onthouden kan worden en tevens de precisie meegegeven wordt, waarmee domeigval berekend moet worden:

```
procedure output (k); value k; integer k;
begin real d1, d2;
    if k = 0 then d1:= d2:= 0; if k > 0 then
        begin d2:= d1; d1:= domeigval;
            n:= if abs((d1-d2)/d2) < 10-4 then k else 80
        end; S1; ...; SM
    end output;
```

Opmerking 2: vermindering van het aantal array's

De procedure richardson gebruikt drie array's, te weten:

- 1) $u[lj:uj, ll:ul]$ voor het onthouden van u_k .
- 2) $v[lj:uj, ll:ul]$ voor het onthouden van u_{k-1} ; ook fungeert v als doorschuifarray.
- 3) $res[lj:uj, ll:ul]$ voor het onthouden van $Lu_k - g$ (zie ook 1.3, iteration).

De gebruiker kan geheugenruimte besparen door het array $res[lj:uj, ll:ul]$ te laten vervallen.

De volgende veranderingen zijn dan nodig:

- 1) de procedure residual moet als volgt gedeclareerd worden:

```
real procedure residual (u,i,j); value i,j; integer i,j; array u;
<body>;
```

Na aanroep van residual staat in $u[i,j]$ nog steeds dezelfde waarde als voor de aanroep van residual en de waarde van het residu voor de component (i,j) wordt toegekend aan residual zelf!

- 2) in de procedure iteration en in de nulde iteratieslag komen de volgende veranderingen (zie ook programma tekst in 5.1):

de regels 18, 19, 20 en 60, 61, 62 vervallen;

de regels 23 en 68 veranderen als volgt:

```
r. 23   begin auxv:= v[j,1]; auxu:= u[j,1]; auxres:= residual(u,j,1);
r. 68   begin auxres0:= residual (u,j,1);
```

4. Experimenten

4.1. Dirichletprobleem voor de snaar

$$\left\{ \begin{array}{l} -\frac{d^2 u}{dx^2} = 0 \quad \text{in } (0, \pi), \\ u(0) = u(\pi) = 0. \end{array} \right.$$

Discretisatie met een rooster (roosterafstand h) van de operator $-\frac{d^2}{dx^2}$ geeft een matrixvergelijking van het type $Lu = g$ met $g \equiv 0$ en:

$$L = \frac{1}{h^2} \begin{pmatrix} 2 & -1 & & & & & & & \\ -1 & 2 & -1 & & & & & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & \cdot & \cdot & \\ & & & & & \cdot & \cdot & \cdot & \\ & & & & & & \cdot & \cdot & \\ & & & & & & & -1 & \\ & & & & & & -1 & 2 & \end{pmatrix}.$$

De eigenwaarden van L volgen uit

$$\lambda_n = \frac{2}{h^2} (1 - \cos nh),$$

$$n = 1, 2, \dots, N \quad (Nh = \pi).$$

Genomen is $h = \frac{\pi}{11}$, zodat $\lambda_1 \approx .993$ en $\sigma(L) \leq 49$.

Als beginbenadering is gekozen $u_0(x) = x(\pi - x)$, $x = jh$ met

$j = 0, 1, 2, \dots, 11$. Richardson is op dit probleem getest met $a = .9$ en $b = 49$.

De aanroep van richardson is dan als volgt:

```
richardson (u,0,11,1,1,true,residual,.9,49,47,discr,k,rateconv,
           domeigval,output);
```

waarin de procedure residual gegeven wordt door:

```

procedure residual (u); array u;
begin real u1, u2;
    u1:= u[lj,1];
    for j:= lj+1 step 1 until uj-1 do
        begin u2:= u[j,1]; u[j,1]:= (2*u2-u1-u[j+1,1])/h2;
            u1:= u2
        end
    end residual;

```

A. richardson met a = .9 en b = 49

k	discr [1]	discr [2]	rateconv	domeigval
0	.633 ₁₀ ⁺¹	.200 ₁₀ ⁺¹	-	-
10	.499 ₁₀ ⁻⁰	.233 ₁₀ ⁻⁰	.23	+.86722 ₁₀ ⁻⁰
20	.185 ₁₀ ⁻¹	.953 ₁₀ ⁻²	.28	-.90196 ₁₀ ⁺¹
30	.305 ₁₀ ⁻²	.152 ₁₀ ⁻²	.25	+.73137 ₁₀ ⁻⁰
40	.207 ₁₀ ⁻³	.931 ₁₀ ⁻⁴	.25	+.89999 ₁₀ ⁻⁰
48	.889 ₁₀ ⁻⁵	.407 ₁₀ ⁻⁵	.27	+.85693 ₁₀ ⁻⁰

De theoretische convergentiesnelheid wordt gegeven door

$$R(k) = 2\sqrt{a/b} - \ln(2)/k,$$

zodat

$$R(48) = .26.$$

We testen nu elimination op hetzelfde probleem;
 de beginbenadering hebben we verkregen door op dezelfde beginoplossing
 als voor de eerste test 44 reductieslagen toe te passen op het eigen-
 waarden interval [4,49] met de procedure richardson, waardoor we tevens
 een schatting verkregen van de dominante eigenwaarde. In het programma
 wordt dan ook eerst richardson aangeroepen en vervolgens elimination:

```
richardson (u,0,11,1,1,true,residual,4,49,43,discr,k,rateconv,
            domeigval.output);

elimination (u,0,11,1,1,residual,4,49,n,discr,k,rateconv,domeigval,
            output);
```

B. elimination met a = 4 en b = 49

k	discr [1]	discr [2]	rateconv	domeigval	opmerkingen
0	.632 ₁₀ ⁺¹	.200 ₁₀ ⁺¹	-	-	reductiefase
10	.274 ₁₀ ⁺¹	.115 ₁₀ ⁺¹	.068	+.99806	
20	.127 ₁₀ ⁺¹	.535 ₁₀ ⁻⁰	.073	+.993221	
30	.586 ₁₀ ⁻⁰	.247 ₁₀ ⁻⁰	.074	+.9932212	
40	.271 ₁₀ ⁻⁰	.114 ₁₀ ⁻⁰	.075	+.9932212	
44	.184 ₁₀ ⁻⁰	.777 ₁₀ ⁻¹	.075	+.9932212	
4	.429 ₁₀ ⁻¹⁰	.293 ₁₀ ⁻¹⁰	5.48	-.164 ₁₀ ⁻²⁰	3 slagen elimination

De convergentie in vergelijking met richardson op het interval [.9,49], t.o.v. de beginapproximatie $u_0 = x(\pi-x)$, wordt berekend uit

$$R(48) = \frac{44}{48} * .075 + \frac{4}{48} * 5.48 = .53.$$

De asymptotische convergentiesnelheid is gelijk aan:

$$R(\infty) = 2\sqrt{a/b} = .57.$$

4.2. Dirichletprobleem voor het membraan

$$\left\{ \begin{array}{l} -\Delta u = 0 \text{ in } G, \quad G = (0, \pi; 0, \pi), \\ u = 0 \text{ op } \partial G, \quad \text{en } \Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}. \end{array} \right.$$

Discretiseren van de operator $-\Delta$ geeft weer een matrix vergelijking $Lu = g$ met $g \equiv 0$, en L gegeven door:

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

waarin h de roosterafstand voorstelt van het rooster waarmee het probleem gediscretiseerd is.

De eigenwaarden van L volgen uit

$$\lambda_n = \frac{4}{h^2} (1 - \cos nh), \quad n = 1, 2, \dots, N, \quad (Nh = \pi).$$

We stellen weer $h = \pi/11$; voor de eigenwaarden volgen dan de grenzen $a = 2$ en $b = 96$.

Als beginbenadering hebben we genomen u_0^* met $u_0^* \equiv 1$ in G en $u_0^* \equiv 0$ op ∂G . Eerst is richardson toegepast op dit interval.

aanroep:

```
richardson (u,0,11,0,11,true,residual,2,96,49,discr,k,rateconv,
           domeigval,output);
```

waarin de procedure residual gegeven wordt door:

```

procedure residual (u); array u;
begin array u1[11:u1]; real u2;
  u1[11] := u[1j+1,11];
  for l:= 11+1 step 1 until u1-1 do u1[l] := u[1j,l];
  for j:= 1j+1 step 1 until uj-1 do
    begin for l:= 11+1 step 1 until u1-1 do
      begin u2 := u[j,l];
        u[j,l] := (4*u2-u1[l-1]-u[j,l+1]-u[j+1,l]-u1[l])/h2;
        u1[l] := u2
      end;
      u1[11] := u[j+1,11]
    end
  end residual;

```

De resultaten staan in de eerstvolgende tabel.

Vervolgens hebben we elimination getest op het interval [4,96], met een aangepaste beginoplossing, verkregen door 43 reductieslagen met de procedure richardson uit te voeren op u_0^* (zie vorige pag.), waardoor tevens een schatting verkregen is van de dominante eigenwaarde; richardson en elimination worden dan ook beide in het programma aangeroepen met dezelfde grenzen voor de eigenwaarden:

```

richardson (u,0,11,0,11,true,residual,4,96,43,discr,k,rateconv,
           domeigval,output);
elimination (u,0,11,0,11,residual,4,96,n,discr,k,rateconv,
           domeigval,output);

```

De resultaten staan in de tweede tabel op de volgende pagina.

A. richardson met a = 2, b = 96

k	discr [1]	discr [2]	rateconv	domeigval
0	$.849_{10}^{+2}$	$.245_{10}^{+2}$	-	-
10	$.803_{10}^{+1}$	$.233_{10}^{+1}$	.235	$-.542_{10}^{+2}$
20	$.399_{10}^{-0}$	$.937_{10}^{-1}$	.273	$-.165_{10}^{+2}$
30	$.201_{10}^{-1}$	$.412_{10}^{-2}$	.289	$-.141_{10}^{+1}$
40	$.107_{10}^{-2}$	$.274_{10}^{-3}$	.283	$+.703_{10}^{-0}$
50	$.426_{10}^{-4}$	$.116_{10}^{-4}$	.285	$-.190_{10}^{+1}$

De theoretische convergentiesnelheid wordt gegeven door

$$R(50) = 2\sqrt{a/b} - \frac{\ln(2)}{50} = .28.$$

B. elimination met a = 4, b = 96

k	discr [1]	discr [2]	rateconv	domeigval	opmerkingen
0	$.849_{10}^{+2}$	$.245_{10}^{+2}$	-	-	reductiefase
10	$.569_{10}^{+1}$	$.141_{10}^{+1}$	.277	$+.177_{10}^{+1}$	
20	$.161_{10}^{+1}$	$.291_{10}^{-0}$	.209	$+.19656_{10}^{+1}$	
30	$.489_{10}^{-0}$	$.870_{10}^{-1}$	.179	$+.19876_{10}^{+1}$	
40	$.148_{10}^{-0}$	$.264_{10}^{-1}$	.164	$+.198622_{10}^{+1}$	
44	$.818_{10}^{-1}$	$.146_{10}^{-1}$	.161	$+.198644_{10}^{+1}$	
7	$.485_{10}^{-6}$	$.856_{10}^{-7}$	1.72	$+.36_{10}^{-4}$	6 slagen elimination

De convergentie t.o.v. de beginapproximatie u_0^* wordt gegeven door:

$$R(51) = \frac{44}{51} * .161 + \frac{7}{51} * 1.72 = .37.$$

De theoretisch te verwachten convergentie is gelijk aan:

$$R(\infty) = 2\sqrt{a/b} = .39.$$

4.3. Matrixprobleem uit de rheologie

Het probleem is het berekenen van de verdelingsfunctie van moleculen die in het XYZ coördinatenstelsel een laminaire stroming ondervinden parallel aan het YZ-vlak met een snelheidsgradiënt in de positieve X-richting. Dit leidt dan tot een partiële elliptische differentiaalvergelijking die m.b.v. de methode der kleinste kwadraten aangepakt kan worden. Daarbij wordt een matrix A bepaald, waarvan o.a. de kleinste eigenwaarde berekend moet worden.

A is in ons geval een 16×16 matrix. We geven van A de diagonaalelementen en het grootste (in absolute waarde) niet-diagonaal element:

.27	3.1														
	37.3	12.5													
3.1	37.9	10.2													
		36.8	9.5												
			404	0	29.3										
		12.5	406	-24.2	23.4										
		10.2	404			24.8									
			9.5	403			24.2								
				0	401			20.1							
				-24.2	1772										
			29.3			1776									
				23.4			1772								
					24.8			1771							
						24.2			1770	4.8					
							20.1				1768				
												4.8	1766		

Opm.: de overige elementen van A zijn niet alle 0.

Matrix A heeft de volgende eigenschappen:

- 1°. A is symmetrisch.
- 2°. A is positief definit.
- 3°. A is sterk diagonaal dominant, behalve in de eerste rij.
- 4°. Uit het theorema van Gerschgorin volgt voor de ligging der eigenwaarden van A het volgende:

er zijn drie clusters van eigenwaarden, nl. een cluster bestaande uit 3 eigenwaarden met zwaartepunt ± 37.3 , een volgende cluster van 5 eigenwaarden met zwaartepunt ± 403 , en ten slotte een cluster van 7 eigenwaarden met zwaartepunt ± 1770 ;

er blijft nog één eigenwaarde over, die in een cirkel met straal ± 3 ligt met middelpunt $.27$.

We beschouwen het probleem $Au = 0$.

Als beginbenadering kiezen we $u_0 = 1$.

Verder is $b = \sigma(A) = 1832$.

We bepalen nu eerst de dominante eigenwaarde van A met de procedure richardson door de benedengrens a van de eigenwaarden van A te variëren.

De resultaten vermelden we in de volgende tabel:

a	domeigval	aantal iteraties
1	1852	80
10	1838	80
20	-	80
40	-	80
60	36.4	80
80	36.7	80
100	36.8	80
120	36.8	80
200	37.0	80

Conclusie: de dominante eigenwaarde is niet de kleinste eigenwaarde, maar ligt in de omgeving van 37.0.

Het ligt nu voor de hand om twee experimenten uit te voeren:

- A. reductie op het interval [36,1832] met richardson;
- B. reductie-eliminatie op het interval [200,1832] met richardson en elimination.

In het geval A is de aanroep van richardson als volgt:

```
richardson (u,1,16,1,1,true,residual,36,1832,160,discr,k,rateconv,
           domeigval,output);
```

met voor de procedure residual:

```
procedure residual (u); array (u);
begin array v[lj:uj];
    for j:= lj step 1 until uj do
        v[j]:= matmat(lj,uj,j,1,g,u);
    for j:= lj step 1 until uj do u[j,1]:= v[j]
end residual;
```

Voor de procedure matmat zij verwezen naar [6].

De resultaten staan in de tabel op pag. 20.

A. richardson met a = 36 en b = 1832

k	discr [1]	discr [2]	rateconv	domeigval
0	.481 10^{+4}	.180 10^{+4}	-	-
10	.500 10^{+3}	.198 10^{+3}	.22	+1839
20	.196 10^{+2}	.908 10^{+1}	.27	+1876
30	.727 10^{-0}	.307 10^{-0}	.29	+2010
40	.632 10^{-1}	.381 10^{-1}	.28	+1904
50	.529 10^{-2}	.257 10^{-2}	.27	+1869
60	.551 10^{-3}	.421 10^{-3}	.26	-43.2
70	.423 10^{-3}	.421 10^{-3}	.23	-.116 10^{-0}
80	.4225 10^{-3}	.421 10^{-3}	.20	-.167 10^{-3}
100	.4225 10^{-3}	.421 10^{-3}	.16	+.54 10^{-3}
120	.4225 10^{-3}	.421 10^{-3}	.13	+.70 10^{-3}
140	.4225 10^{-3}	.421 10^{-3}	.11	+.32 10^{-3}
160	.4225 10^{-3}	.421 10^{-3}	.10	+.37 10^{-3}

De kleinste eigenwaarde is orde 10^{-3} , en heeft blijkbaar nauwelijks invloed op de oplossing van het probleem.

In het geval B passen we dus reductie en eliminatie toe en wel als volgt:

we reduceren totdat een dominante eigenwaarde verkregen is, en elimineren deze dan; vervolgens voeren we weer een aantal reductieslagen uit tot een nieuwe waarde voor domeigval berekend is, en elimineren deze, enz.

De aanroepen van richardson en elimination zijn:

```

richardson (u,1,16,1,1,true,residual,200,1832,30,discr,k,rateconv,
            domeigval,output);
elimination (u,1,16,1,1,residual,200,1832,n,discr,k,rateconv,
            domeigval,output);
richardson (u,1,16,1,1,true,residual,200,1832,20,discr,k,rateconv,
            domeigval,output);
elimination (u,1,16,1,1,residual,200,1832,n,discr,k,rateconv,
            domeigval,output);
richardson (u,1,16,1,1,true,residual,200,1832,40,discr,k,rateconv,
            domeigval,output);

```

De resultaten staan weer in een tabel.

B. elimination met a = 200, b = 1832

k	discr [1]	discr [2]	rateconv	domeigval	opmerkingen
0	.481 ₁₀ ⁺⁴	180 ₁₀ ⁺⁴	-	-	1e reductiefase
10	.349 ₁₀ ⁺²	.212 ₁₀ ⁺²	.47	+.234 ₁₀ ⁺²	
20	.176 ₁₀ ⁺²	.107 ₁₀ ⁺²	.27	+.38209 ₁₀ ⁺²	
30	.856 ₁₀ ⁺¹	.508 ₁₀ ⁺¹	.20	+.382156 ₁₀ ⁺²	
4	.246 ₁₀ ⁻⁰	.165 ₁₀ ⁻⁰	1.16	-.347 ₁₀ ⁺¹	3 slagen elimination
0	.246 ₁₀ ⁻⁰	.165 ₁₀ ⁻⁰	-	-	2e reductiefase
10	.132 ₁₀ ⁻⁰	.888 ₁₀ ⁻¹	.06	+.35650 ₁₀ ⁺²	
20	.665 ₁₀ ⁻¹	.449 ₁₀ ⁻¹	.06	+.35605 ₁₀ ⁺²	
4	.802 ₁₀ ⁻³	.559 ₁₀ ⁻³	1.47	-.300 ₁₀ ⁺¹	3 slagen elimination
0	.802 ₁₀ ⁻³	.559 ₁₀ ⁻³	-	-	3e en laatste reductiefase
10	.551 ₁₀ ⁻³	.409 ₁₀ ⁻³	.034	+.714 ₁₀ ⁺¹	
20	.460 ₁₀ ⁻³	.415 ₁₀ ⁻³	.014	+.300 ₁₀ ⁺¹	
30	.433 ₁₀ ⁻³	.418 ₁₀ ⁻³	.015	+.871 ₁₀ ⁻⁰	
40	.425 ₁₀ ⁻³	.419 ₁₀ ⁻³	.012	+.212 ₁₀ ⁻⁰	

5. ALGOL 60 tekst

We geven nu de ALGOL 60 tekst van beide procedures.

5.1. procedure richardson (ALGOL 60 tekst)

```

1. 'procedure' richardson(u,lj,uj,ll,ul,inap,residual,a,b,n,descr,k,
2. rateconv,domeigval,output); 'value' lj,uj,ll,ul,a,b;
3. 'integer' n,k,lj,uj,ll,ul; 'real' a,b,rateconv,domeigval;
4. 'boolean' inap; 'array' u,descr; 'procedure' residual,output;

5. 'begin' 'integer' j,l; 'real' x,y,z,y0,c,d,alfa,omega,omega0,
6. eigmax,eigeucl,euclres,maxres,rcmax,rceuc1,maxres0,euclres0;
7. 'array' v,res[lj:uj,ll:ul];

8. 'procedure' calpar;
9. 'comment' calpar calculates the parameters alfa and omega
10. of each iteration;
11. 'begin' alfa:= 1/(1 - alfa/z);
12. omega:= 1/(x - omega * y)
13. 'end' calpar;

14. 'procedure' iteration;
15. 'comment' first the iteration formula is constructed;
16. 'begin' 'real' auxv,auxu,auxres,eucluv,maxuv;
17. eucluv:= euclres:= maxuv:= maxres:= 0;
18. 'for' j:= lj 'step' 1 'until' uj 'do'
19. 'for' l:= ll 'step' 1 'until' ul 'do' res[j,l]:= u[j,l];
20. residual(res);
21. 'for' j:= lj 'step' 1 'until' uj 'do'
22. 'for' l:= ll 'step' 1 'until' ul 'do'
23. 'begin' auxv:= v[j,l]; auxu:= u[j,l]; auxres:= res[j,l];
24. auxv:= alfa * auxu - omega * auxres + (1 - alfa) * auxv;
25. u[j,l]:= auxv; v[j,l]:= auxu;

26. 'comment' the norms of the k-th residual and the difference
27. between the (k+1)-th and k-th iterand are calculated;
28. auxu:= abs(auxu - auxv); auxres:= abs(auxres);
29. maxuv:= 'if' maxuv < auxu 'then' auxu 'else' maxuv;
30. maxres:= 'if' maxres < auxres 'then' auxres 'else' maxres;
31. eucluv:= eucluv + auxu * auxu;
32. euclres:= euclres + auxres * auxres;
33. 'end';
34. eucluv:= sqrt(eucluv); euclres:= sqrt(euclres);
35. descr[1]:= euclres; descr[2]:= maxres;

36. 'comment' domeigval is evaluated;
37. maxuv:= maxres/maxuv; eucluv:= euclres/eucluv;
38. eigmax:= maxuv * (c - maxuv)/(.25 * d - maxuv);
39. eigeucl:= eucluv * (c - eucluv)/(.25 * d - eucluv);
40. domeigval:= .5 * (eigmax + eigeucl);

```

```

41.      'comment' finally the rate of convergence is calculated;
42.      rceuc1:= -ln(euclres/ euclres0)/k;
43.      rcmax:= -ln(maxres/maxres0)/k;
44.      rateconv:= .5 * (rceuc1 + rcmax)
45.      'end' iteration;

46.      'comment' the constants for starting calpar are calculated;
47.      alfa:= 2; omega:= 4/(b + a); y0:= (b + a)/(b - a);
48.      x:= .5 * (b + a); y:= (b - a) * (b - a)/16; z:= 4 * y0 * y0;

49.      'comment' the constants needed for domeigval are calculated;
50.      c:= a * b; c:= sqrt(c); d:= sqrt(a) + sqrt(b); d:= d * d;

51.      'comment' the initial approximation is put into array v;
52.      'if' inap 'then'
53.      'begin' 'for' j:= 1j 'step' 1 'until' uj 'do'
54.      'for' l:= 1l 'step' 1 'until' ul 'do' v[j,l]:= u[j,l]
55.      'end' 'else'
56.      'begin' 'for' j:= 1j 'step' 1 'until' uj 'do'
57.      'for' l:= 1l 'step' 1 'until' ul 'do' v[j,l]:= 1
58.      'end';

59.      'comment' the zeroth iteration is now performed;
60.      'for' j:= 1j 'step' 1 'until' uj 'do'
61.      'for' l:= 1l 'step' 1 'until' ul 'do' res[j,l]:= v[j,l];
62.      residual(res);
63.      k:= 0; omega0:= 2/(b+a);
64.      'begin' 'real' auxres0;
65.      maxres0:= euclres0:= 0;
66.      'for' j:= 1j 'step' 1 'until' uj 'do'
67.      'for' l:= 1l 'step' 1 'until' ul 'do'
68.      'begin' auxres0:= res[j,l];
69.      u[j,l]:= v[j,l] - omega0 * auxres0;
70.      auxres0:= abs(auxres0);
71.      maxres0:= 'if' maxres0 < auxres0 'then' auxres0 'else' maxres0;
72.      euclres0:= euclres0 + auxres0 * auxres0
73.      'end';
74.      euclres0:= sqrt(euclres0)
75.      'end';
76.      discr[1]:= euclres0; discr[2]:= maxres0;
77.      output(k);
78.      'if' k 'ge' n 'then' 'goto' finally;

79.  next step:
80.      k:= k + 1; calpar; iteration; output(k);
81.      'if' k < n 'then' 'goto' next step;

82.      'comment' now an empty iteration is performed in order to
83.      calculate the norm of the residual, rateconv and domeigval with
84.      respect to the last approximation;
85.      finally:
86.      k:= k + 1; calpar; iteration;
87.      'for' j:= 1j 'step' 1 'until' uj 'do'
88.      'for' l:= 1l 'step' 1 'until' ul 'do' u[j,l]:= v[j,l];
89.      output(k);

90.      'end' richardson;

```

5.2. procedure elimination (ALGOL 60 tekst)

```

1. 'procedure' elimination(u,lj,uj,ll,ul,residual,a,b,n,discr,k,
2. rateconv,domeigval,output); 'value' lj,uj,ll,ul,b;
3. 'integer' lj,uj,ll,ul,n,k; 'real' a,b,rateconv,domeigval;
4. 'array' u,discr; 'procedure' residual,output;

5. 'begin' 'real' pi,auxcos,c,d;

6.   'real' 'procedure' arccos(x); 'value' x; 'real' x;
7.   arccos:= 'if' x 'ge' 0 'then' arctan(sqrt(1-x*x)/x) 'else'
8.           pi + arctan(sqrt(1-x*x)/x);

9.   'real' 'procedure' tanh(z); 'value' z; 'real' z;
10.  'begin' 'real' u; u:= exp(-2*z);
11.    tanh:= (1 - u) / (1 + u)
12.  'end' tanh;

13.  'real' 'procedure' optpol(x); 'value' x; 'real' x;
14.  'begin' 'real' w,y;
15.  1: w:= (b * cos(.5*pi/x) + domeigval) / (b - domeigval);
16.    'if' abs(w) < 1 'then'
17.      'begin' y:= arccos(w);
18.        optpol:= 2 * sqrt(a/b) + sin(x*y) / cos(x*y) *
19.        (y - b*pi*sin(.5*pi/x)*.5 / (x * (b-domeigval) * sqrt(1-w*w)))
20.      'end' 'else'
21.      'begin' 'if' abs(w) > 1 'then'
22.        'begin' y:= ln(w + sqrt(w*w-1));
23.          optpol:= 2 * sqrt(a/b) - tanh(x*y) * (y + b*pi*sin(.5*pi/x)*
24.          .5/x/(b-domeigval)/sqrt(w*w-1))
25.        'end' 'else'
26.        'begin' x:= x + 10-2; 'goto' 1 'end'
27.      'end';
28.  'end' optpol;

29.  pi:= 4 * arctan(1);
30.  c:= 1;
31.  'if' optpol(c) < 0 'then'
32.    'begin' d:= .5 * pi * sqrt(b/domeigval);
33.    m: d:= d + d;
34.    'if' zeroin(c,d,optpol(c),c*10-3) 'then' n:= entier(c+.5)
35.    'else' 'goto' m;
36.  'end' 'else' n:= 1;

37.  auxcos:= cos(.5*pi/n);
38.  a:= (2*domeigval + b*(auxcos-1))/(auxcos+1);
39.  richardson(u,lj,uj,ll,ul,'true',residual,a,b,n-1,discr,k,rateconv,
40.  domeigval,output)

41. 'end' elimination;

```

Literatuur:

- [1] Houwen, P.J. van der, Finite difference methods for solving partial differential equations. MC tract 20, Mathematisch Centrum, Amsterdam (1968).
- [2] Houwen, P.J. van der, On the acceleration of Richardson's method I, theoretical part. TW report 104, Mathematisch Centrum, Amsterdam (1967).
- [3] Houwen, P.J. van der, On the acceleration of Richardson's method II, numerical aspects. TW report 107, Mathematisch Centrum, Amsterdam (1967).
- [4] Houwen, P.J. van der, On the acceleration of Richardson's method III, applications. TW report 108, Mathematisch Centrum, Amsterdam (1967).
- [5] Coolen, T.M.T. en Houwen, P.J. van der, On the acceleration of Richardson's method IV, a non-symmetrical case. TW report 109, Mathematisch Centrum, Amsterdam (1968).
- [6] Dekker, T.J., ALGOL 60 procedures in numerical algebra, part 1. MC tract 22, Mathematisch Centrum, Amsterdam (1968).
- [7] Dekker, T.J. en Hoffmann, W., ALGOL 60 procedures in numerical algebra, part 2. MC tract 23, Mathematisch Centrum, Amsterdam (1968).
- [8] Grune, D. Handleiding milli-systeem voor de EL X8. LR1.1, Mathematisch Centrum, Amsterdam (1971).

