TEKST NR 231B 1992

Elektrondiffusion i silicium

- en matematisk model
Kildetekster

af

Jesper Voetmann
Karen Birkelund
Mette Olufsen
Ole Møller Nielsen

Roskilde Universitetscenter - juni 1991

TEKSTER fra

IMFUFA ROSKILDE UNIVERSITETSCENTER
INSTITUT FOR STUDIET AF MATEMATIK OG FYSIK SAMT DERES FUNKTIONER I UNDERVISNING, FORSKNING OG ANVENDELSER

IMFUFA, Roskilde Universitetscenter, Postbox 260, 4000 Roskilde

Elektrondiffusion i silicium - en matematisk model.

af: Jesper Voetmann,
Karen Birkelund,
Mette Olufsen,
Ole Møller Nielsen.

Vejledere: Johnny Ottesen,
H.B.Hansen.

IMFUFA tekst nr. 231B/92, RUC. 66 sider. ISSN 0106-6242

Abstract

Teksten rummer kildeteksterne til systemet gennemgået i tekst nr 231A - Elektrondiffusion i silicium.

Indhold

Kapitel 1

Indledning

Denne delrapport (tekst 231B) rummer udskrifter af hele DIFFUSYS^{TM}.

Selve implementeringen findes i de egentlige SIMULA kildetekster (kapitel 2), hvis design er beskrevet i tekst 231A.

Brugergrænsefladen er imidlertid repræsenteret i de kommandofiler (shell scripts) og projektfiler (makefiler), der sørger for at de rigtige moduler bliver oversat, lænket og udført (kapitel 3).

Filer som ikke bør ses af brugeren har _ som første tegn. F.eks hedder programfilen til numerisk løsning, `_numcalc.sim`, mens den korresponderende kommandofil `numcalc` sørger for oversættelse og lænkning af de nødvendige moduler, samt kørsel af programmet.

Kapitel 2

Simula kildetekster

2.1 Class PROBLEM

Class problem findes i filen `problem.sim`.

```
!-------------------------------------------------------------------+
| Det er i denne programdel, at den konkrete diffusionsligning skal  |
| beskrives.                                                         |
|                                                                    |
| Ligningen har formen :                                             |
|                                                                    |
|          du              d(du)                                     |
|          --(x,t) - alpha -----(x,t) + beta u(x,t) = 0              |
|          dt              dxdx                                      |
|                                                                    |
| hvor u(x,t) er den ubekendte funktion,                             |
|      alpha tilhoerer R+ og                                         |
|      beta  tilhoerer R\R-.                                         |
|                                                                    |
| Randbetingelserne har formerne :                                   |
|                                                                    |
| venstre side (x=0) :                                               |
|                                                                    |
|          du                                                        |
|          --(0,t) = f(t)u(0,t) + g(t)  (Neumann) eller              |
|          dx                                                        |
|                                                                    |
|          u(0,t)  = f(t)               (Dirichlet)                  |
|                                                                    |
| hoejre side (x=1) :                                                |
|                                                                    |
|          du                                                        |
|          --(1,t) = f(t)u(1,t) + g(t)  (Neumann) eller              |
|          dx                                                        |
|                                                                    |
|          u(1,t)  = f(t)               (Dirichlet)                  |
|                                                                    |
| hvor f(t) og g(t) er vilkaarlige kontinuerte funktioner.           |
|                                                                    |
```

```
35  | Begyndelsesbetingelsen har formen :                                  |
36  |                                                                      |
37  |          u(x,0)  = h(x)                                              |
38  |                                                                      |
39  | hvor h(x) er en vilkaarlig kontinuert funktion defineret paa         |
40  | intervallet [0,1].                                                   |
41  |                                                                      |
42  |    ------------------------------------------------------------      |
43  |                                                                      |
44  | Programmet loeser ligningen numerisk ved brug af S.O.R. metoden,     |
45  | og skal dertil bruge foelgende stoerrelser :                         |
46  |                                                                      |
47  | Final_time: Tidspunktet hvor udregningen af u(x,t), skal stoppe.     |
48  | Spacesteps: Antal punkter, som x-intervallet opdeles i (f.eks  10).  |
49  | Timesteps : Antal punkter, som t-intervallet opdeles i (f.eks 400).  |
50  |                                                                      |
51  |    ------------------------------------------------------------      |
52  |                                                                      |
53  | Yderligere vejledning kan faas i den medfoelgende brugervejledning.  |
54  |                                                                      |
55  +----------------------------------------------------------------------;
56
57
58  class PROBLEM;
59  begin
60
61    !--------------------------------------------------------------------+
62    | Ligningens koefficienter alpha og beta skal angives efter          |
63    | lighedstegnet, i form af et rationelt tal afsluttet med et         |
64    | semikolon.                                                         |
65    |                                                                    |
66    | Eksempel :                                                         |
67    |                                                                    |
68    | long real alpha = 1.4738;                                          !
69    | long real beta  = 1/3;                                             !
70    |                                                                    |
71    +--------------------------------------------------------------------;
72
73    long real alpha = 1;
74    long real beta  = 1;
75
76
77    !--------------------------------------------------------------------+
78    | S.O.R. parametrene timesteps, spacesteps og final_time skal        |
79    | angives her, hver efterfulgt af et semikolon, idet                 |
80    |                                                                    |
81    |  - spacesteps, timesteps er hele tal.                              |
82    |  - final_time er et rationelt tal.                                 |
83    |                                                                    |
84    | Eksempel :                                                         |
85    |                                                                    |
86    | integer timesteps    = 1000;                                       !
87    | integer spacesteps   =   50;                                       !
88    | long real final_time =    1.5;                                     !
89    |                                                                    |
90    +--------------------------------------------------------------------;
91
92    integer timesteps    =   1000;
93    integer spacesteps   =     35;
94    long real final_time =    100;
95
```

```
96
97    !-------------------------------------------------------------------+
98    | Begyndelsesbetingelsen (t=0) :                                    |
99    |                                                                   |
100   | Det er kun de i eksemplet naevnte linier, der skal skrives.       |
101   |                                                                   |
102   | Eksempel 1 :                                                      |
103   |                                                                   |
104   |   h := 0;                                                         !
105   |                                                                   |
106   | Eksempel 2 :                                                      |
107   |                                                                   |
108   |   h := 2 * x;                                                     !
109   |                                                                   |
110   | Eksempel 3 :                                                      |
111   |                                                                   |
112   |   h := if x < 0.5 then 2 * x                                      |
113   |                   else 2 * (1 - x);                               !
114   |                                                                   |
115   +-------------------------------------------------------------------;
116
117   class Init;
118   begin
119     long real procedure h(x);
120     long real x;
121     begin
122       h := 0;                                  !En funktion af x;
123     end;
124   end ** class Init **;
125
126
127   !-------------------------------------------------------------------+
128   | Randbetingelserne i venstre side (x=0) :                          |
129   |                                                                   |
130   | Det er kun de i eksemplerne naevnte linier, der skal skrives.     |
131   |                                                                   |
132   | Eksempel 1 (Neumann - randbetingelse) :                           |
133   |                                                                   |
134   |   boolean Neumann = TRUE;                                         !
135   |                                                                   |
136   |   f := 0;                                                         !
137   |   g := cos(anglefreq*t);                                          !
138   |                                                                   |
139   | Eksempel 2 (Dirichlet - randbetingelse) :                         |
140   |                                                                   |
141   |   boolean Neumann = FALSE;                                        !
142   |                                                                   |
143   |   f := 1.3 * sin(anglefreq*t);                                    !
144   |                                                                   !
145   |   (g's vaerdi er ligegyldig i Dirichlet tilfaeldet.)              |
146   +-------------------------------------------------------------------;
147
148   class Left;
149   begin
150     boolean Neumann = true;              !Typen;
151
152     long real procedure f(t);
153     long real t;
154     begin
155       f := 0;
156     end;
```

```
157
158     long real procedure g(t);
159     long real t;
160     begin
161       g := E_0 * sin(anglefreq * t);
162     end;
163   end ** class Left **;
164
165
166   !-------------------------------------------------------------------+
167   | Randbetingelserne i hoejre side (x=1) :                           |
168   |                                                                   |
169   | Det er kun de i eksemplerne naevnte linier, der skal skrives.     |
170   |                                                                   |
171   | Eksempel 1 (Neumann - randbetingelse) :                           |
172   |                                                                   |
173   |   boolean Neumann = TRUE;                                         !
174   |                                                                   |
175   |   f := 0;                                                         !
176   |   g := cos(anglefreq * t);                                        !
177   |                                                                   |
178   | Eksempel 2 (Dirichlet - randbetingelse) :                         |
179   |                                                                   |
180   |   boolean Neumann = FALSE;                                        !
181   |                                                                   |
182   |   f := -1.3 * sin(anglefreq * t);                                 !
183   |                                                                   !
184   |   (g's vaerdi er ligegyldig i Dirichlet tilfaeldet.)              |
185   +-------------------------------------------------------------------;
186
187   class Right;
188   begin
189     boolean Neumann = true;              !Typen;
190
191     long real procedure f(t);
192     long real t;
193     begin
194       f := 0;
195     end;
196
197     long real procedure g(t);
198     long real t;
199     begin
200       g := E_0 * sin(anglefreq * t);
201     end;
202
203   end ** class Right **;
204
205
206   !-------------------------------------------------------------------+
207   | Definition af konstanter:                                         |
208   |                                                                   |
209   | De konstanter, som benyttes i ligningen, samt i dens randbetin-   |
210   | gelser beskrives her. Det er konstanter af typen:                 |
211   |                                                                   |
212   |   long real        - reelle tal med stor praecision.              |
213   |   real             - reelle tal.                                  |
214   |   integer          - heltal.                                      |
215   |                                                  -9               |
216   | Tal angivet paa eksponenetform for eksempel  3 10   skrives som:  |
217   |                                                                   |
```

```
218    |   3&-9                                                          |
219    |                                                                 |
220    | Eksempel                                                        |
221    |                                                                 |
222    | I randbetingelserne defineret ovenfor benyttes konstanterne E_0 |
223    | og pi, de defineres som foelger:                                |
224    |                                                                 |
225    |   long real pi  = 3.1415926538979;                              !
226    |   long real E_0 = 22.22;                                        !
227    |                                                                 |
228    +-----------------------------------------------------------------;
229
230    long real pi      = 3.14159265358979;
231    long real E_0     = 22.22;
232
233
234    !-----------------------------------------------------------------+
235    | Definition af variable :                                        |
236    |                                                                 |
237    | De variable, som benyttes i ligningen, samt i dens randbetin-   |
238    | gelser beskrives her. Med variable menes de symboler, som skal  |
239    | indlæses i programmet fra tastatur eller fil.                   |
240    |                                                                 |
241    | Eksempel:                                                       |
242    |                                                                 |
243    | I definitionen af randbetingelserne i ovenstående kommentarer er |
244    | benyttet variablen "anglefreq", som er et tal, som skal indlaeses|
245    | fra tastatur eller fil. Denne defineres som foelger:            |
246    |                                                                 |
247    |   long real anglefreq;                                          !
248    |                                                                 |
249    | Derefter indlaeses den fra tastatur eller fil naar programmet   |
250    | udfoeres idet foelgende er udfoert:                             |
251    |                                                                 |
252    |   outtext("Frekvens (Hz) : ");                                  !
253    |   breakoutimage;                                                !
254    |   anglefreq   := inreal * 2 * pi;                               !
255    |                                                                 |
256    +-----------------------------------------------------------------;
257
258    long real anglefreq;
259
260    outtext("Frekvens (Hz) : ");
261    breakoutimage;
262    anglefreq   := inreal * 2 * pi;
263
264  end * class PROBLEM *;
```

2.2 Class SOLVER

Class Solver findes i filen `_solver.sim`.

```
external class PROBLEM = "problem";


!-------------------------------------------------------------------+
| Navn      : PROBLEM class SOLVER                                  |
|                                                                   |
| Omgivelser: class PROBLEM.                                        |
|                                                                   |
| Funktion  : Erklaerer foelgende klasser :                         |
|                                                                   |
|             - class Matrix, haandterer loesningen.                |
|             - class Coroutine, praefix for coroutiner.            |
|             - Coroutine class Initial_value, beregner begyndel-   |
|                                          sesvaerdier.             |
|             -          "      Interior, beregner iteration.      |
|             -          "      Neumann_left, beregner randbetin-  |
|                                                 gelser.           |
|             -          "      Neumann_right,       "             |
|             -          "      Dirichlet_left,      "             |
|             -          "      Dirichlet_right,     "             |
|             -          "      Evaluation, varetager Coroutine    |
|                                      forloebet.                   |
|                                                                   |
|             Erklaerer folgende procedurer :                       |
|                                                                   |
|             - procedure calculus, saetter coroutinerne igang.     |
|             - procedure diffu_error, udskriver eventuelle fejl-   |
|               meddelelser og stopper programmet.                  |
|                                                                   |
|             Klassens saetninger beregner de konstante vaerdier,   |
|             som indgaar i SOR iterationen.                        |
|             Endelig testes om parametrene er lovlige.             |
|                                                                   |
| Kaldes af : Programmerne _anacalc eller _numcalc.                 |
|                                                                   |
+-------------------------------------------------------------------;

PROBLEM Class SOLVER;
begin


  !---------------------------------------------------------------+
  | Navn      : class Matrix                                      |
  |                                                               |
  | Omgivelser: integer spacesteps, timesteps  - globale variable i|
  |                                              class SOLVER.    |
  |                                                               |
  | Uddata    : Matrix objekt.                                    |
  |                                                               |
  | Funktion  : Opretter et to-dimensionelt array elm med dimen-  |
  |             sionerne spacesteps og timesteps.                 |
  |                                                               |
  |             Erklaerer procedurerne save, print og minus.      |
  |                                                               |
  | Kaldes af : Hovedprogrammet i class SOLVER.                   |
```

```
56      |                                                                |
57      +----------------------------------------------------------------;
58
59      class Matrix;
60      begin
61        long real array elm(0:spacesteps, 0:timesteps);
62
63
64        !--------------------------------------------------------------+
65        | Navn       : boolean procedure save                          |
66        |                                                              |
67        | Inddata    : text filename.                                  |
68        |                                                              |
69        | Omgivelser: integer spacesteps, timesteps,                   |
70        |             long real final_time  - globale variable i class |
71        |                                   SOLVER.                    |
72        |             This Matrix - den matrix, i hvilken proceduren   |
73        |             findes.                                          |
74        |                                                              |
75        | Uddata     : Logisk vaerdi save, som angiver om this Matrix  |
76        |             er blevet gemt.                                  |
77        |                                                              |
78        | Funktion   : Gemmer vaerdierne spacesteps, timesteps, final_- |
79        |             time samt matricens elemmenter paa fil.          |
80        |                                                              |
81        |             Filen faar navnet filename. Hvis der i forvejen  |
82        |             eksisterer en fil med dette navn, vil denne blive|
83        |             overskrevet med de nye data.                     |
84        |                                                              |
85        | Kaldes af : Programmerne _numcalc og _anacalc.               |
86        |                                                              |
87        +--------------------------------------------------------------;
88
89        boolean procedure save(filename);
90        text filename;
91        begin
92          integer i,j;
93          ref(outfile) file;
94          file :- new outfile(filename);
95          inspect file do
96          begin
97            setaccess("ANYCREATE");
98            if open(blanks(100)) then
99            begin
100             outint(spacesteps, 0);                  ! Gem spacesteps;
101             outimage;
102             outint(timesteps,  0);                  ! Gem timesteps;
103             outimage;
104             outreal(final_time, 16, 0);             ! Gem final_time;
105             outimage;
106             for j:= 0 step 1 until timesteps do
107             begin
108               for i := 0 step 1 until spacesteps do
109               begin
110                 outreal(elm(i,j), 16, 0);           ! Gem Matrix;
111                 outimage;
112               end ******* for i *******;
113             end ****** for j ******;
114             save := TRUE;
115             close;
116           end ***** if open *****;
```

```
117       end **** inspect ****;
118       end *** save ***;
119
120
121       !-------------------------------------------------------------+
122       | Navn      : ref(Matrix) procedure minus                     |
123       |                                                             |
124       | Inddata   : ref(Matrix) subtrahend.                         |
125       |                                                             |
126       | Omgivelser: This Matrix - den matrice, i hvilken proceduren |
127       |             findes.                                         |
128       |             integer spacesteps, timesteps - globale variable |
129       |                                          i class SOLVER.    |
130       |                                                             |
131       | Uddata    : I matricen minus returneres subtraktion mellem  |
132       |             this Matrix og matricen subtrahend.             |
133       |                                                             |
134       | Funktion  : Foerst oprettes matricen temp, hvorefter subtrak-|
135       |             tionen mellem this Matrix's og subtrahenden's   |
136       |             elementer udfoeres raekkevis. Tilsidst kopieres |
137       |             resultatet over i matricen minus.               |
138       |                                                             |
139       | Kaldes af : Programmet  _anacalc.                           |
140       |                                                             |
141       +-------------------------------------------------------------;
142
143       ref(Matrix) procedure minus(subtrahend);
144       ref(Matrix) subtrahend;
145       begin
146         integer i, j;
147         ref(Matrix) temp;
148         temp :- new Matrix;
149         for i := 0 step 1 until spacesteps do
150           for j := 0 step 1 until timesteps do
151             temp.elm(i,j) := elm(i,j) - subtrahend.elm(i,j);
152         minus :- temp;
153       end *** minus ***;
154
155     end ** Matrix **;
156
157
158     !-------------------------------------------------------------+
159     | Navn      : class Coroutine                                 |
160     |                                                             |
161     | Funktion  : Klassen tjener foerst og fremmest som prefix for de|
162     |             objekter, der skal virke som korutiner. Derfor  |
163     |             loesriver den sig fra sin skaber med detach.    |
164     |                                                             |
165     | Kaldes af : Saetningerne i class SOLVER.                    |
166     |                                                             |
167     +-------------------------------------------------------------;
168
169     class Coroutine; detach;
170
171
172     !-------------------------------------------------------------+
173     | Navn      : Coroutine class initial_value                   |
174     |                                                             |
175     | Inddata   : ref(Init) initfunc.                             |
176     |                                                             |
177     | Omgivelser: Matrixobjektet u, som oprettes af saetningerne i |
```

```
178     |             class SOLVER.                                          |
179     |             integer spacesteps   - global variabel i SOLVER.       |
180     |             long real delta_x    - global variabel i SOLVER.       |
181     |                                                                    |
182     | Uddata    : Begyndelsesvaerdierne indsaettes i u's element-        |
183     |             array i foerste soejle, dvs for t = 0                  |
184     |                                                                    |
185     | Funktion  : Paa baggrund af inputparameteren Initfunc, som er      |
186     |             det objekt, hvori den konkrete begyndelsesfunktion     |
187     |             er defineret (i class PROBLEM), indsaettes             |
188     |             begyndelsesvaerdien til alle stedskridt (Fra 0 til     |
189     |             spacesteps) i matricen u.elm                           |
190     |                                                                    |
191     | Kalder    : Initfunc's h(x), her har brugeren programmeret be-     |
192     |             gyndelsesbetingelsen til den konkrete diffusions-      |
193     |             ligning.                                               |
194     |                                                                    |
195     | Kaldes af : saetningerne i class SOLVER, men da det er en co-      |
196     |             routine class, vil objektet detachere med det          |
197     |             samme. Selve kroppen i klassen udfoeres, naar          |
198     |             procedure calculus udfoerer saetningen                 |
199     |             resume(initbound)                                      |
200     |                                                                    |
201     +--------------------------------------------------------------------;
202
203     Coroutine class initial_value(Initfunc);
204     ref(Init) Initfunc;
205     begin
206       integer i;      !sted;
207
208       for i := 0 step 1 until spacesteps do
209         u.elm(i,0) := Initfunc.h(i*delta_x);
210     end ** initial_value **;
211
212
213     !--------------------------------------------------------------------+
214     | Navn      : Coroutine class Interior                               |
215     |                                                                    |
216     | Omgivelser: Matricen u, som oprettes i saetningerne i class        |
217     |             SOLVER.                                                |
218     |             integer spacesteps   - global variabel i SOLVER.       |
219     |             long real omega      - global variabel i SOLVER.       |
220     |                                                                    |
221     | Uddata    : I matricen u's elementer indsaettes funktions-         |
222     |             vaerdierne for ethvert x, t til en givet iteration.    |
223     |                                                                    |
224     | Funktion  : Foerst opdateres tiden, ved hjaelp af eval.j, der      |
225     |             fungerer som processens "ur".                          |
226     |                                                                    |
227     |             Naar dette er gjort beregnes konstanten b(i), for      |
228     |             alle stedskridt i.                                     |
229     |             Dette goeres paa baggrund af vaerdier, som findes      |
230     |             i matricen paa pladserne u.elm(i,j-1).                 |
231     |                                                                    |
232     |             Dernaest beregnes ved brug af SOR iterations-          |
233     |             formlen den naeste iteration, for alle stedskridt      |
234     |             i, til det j'te tidsskridt. Resultatet gemmes i        |
235     |             matricens u's element(i,j).                            |
236     |                                                                    |
237     |             Til sidst beregnes den absolutte fejl mellem denne     |
238     |             og den forrige iteration, hvorefter den sammenlig-     |
```

```
239     |            nes med de oevrige fejl, som er fremkommet ved      |
240     |            dette iterationsskridt. Saaledes gemmes den maksi-  |
241     |            male fejl i objektet eval.                          |
242     |                                                                |
243     |                                                                |
244     |            Alle disse beregninger er indsat i en uendelig      |
245     |            while loekke, idet objektet eval soerger for at     |
246     |                  terminere processen.                          |
247     |                                                                |
248     |            - i Interior overgives processen til rightbound,    |
249     |              som enten er af type Neumann_right eller          |
250     |              Dirichlet_right. Her varetages åberegningen af    |
251     |              randvaerdien i det hoejre endepunkt (x=1).        |
252     |                                                                |
253     | Kalder    : resume(rightbound).                                |
254     |                                                                |
255     | Kaldes af : Objektet oprettes af saetningerne i class SOLVER,  |
256     |             men, da det er en corutine class Interior, vil ob- |
257     |             jektet detachere med det samme. Selve beregning-   |
258     |             ne udfoeres naar det kaldes af:                    |
259     |                                                                |
260     |             Coroutine class Neumann_left eller af              |
261     |             Coroutine class Dirichlet_left med saetningen      |
262     |             resume(equation).                                  |
263     |                                                                |
264     +----------------------------------------------------------------;
265
266     Coroutine class Interior;
267     begin
268       long real new_value, err;
269       long real array b(1:spacesteps-1);
270       integer j;                          !lokal tid;
271       integer i;                          !sted;
272       integer endpoint;
273       endpoint := spacesteps - 1;
274
275       while TRUE do
276       begin
277
278         j := eval.j;
279
280         for i := 1 step 1 until endpoint do
281           b(i) := u.elm(i-1,j-1) + 2 * C2 * u.elm(i,j-1) + u.elm(i+1,j-1);
282
283         while j = eval.j do      ! En SOR iteration;
284         begin
285           for i := 1 step 1 until endpoint do
286           begin
287             new_value  := omega/2/C1 * (u.elm(i-1,j)+u.elm(i+1,j)+b(i)) +
288                           u.elm(i,j) - omega * u.elm(i,j);
289             err        := abs(u.elm(i,j) - new_value);
290             u.elm(i,j) := new_value;
291             if err > eval.err then eval.err := err;
292           end ***** for i *****;
293
294           resume(rightbound);
295         end **** while j = eval.j ****;
296       end *** while TRUE do ***;
297     end ** Interior **;
298
299
```

```
300     !-------------------------------------------------------------+
301     | Navn      : Coroutine class Neumann_left                    |
302     |                                                             |
303     | Omgivelser: Matricen u, som oprettes i saetningerne i class |
304     |             SOLVER.                                         |
305     |             long real delta_x, delta_t  - globale variable. |
306     |             long real omega             - global  variabel. |
307     |                                                             |
308     | Uddata    : I matricen u's elementer indsaettes funktions-  |
309     |             vaerdierne for x = 0, og til ethvert t en givet |
310     |             iteration.                                      |
311     |                                                             |
312     | Funktion  : Foerst opdateres tiden, ved hjaelp af eval.j, der |
313     |             fungerer som processens "ur".                   |
314     |                                                             |
315     |             Naar dette er gjort beregnes de konstanter, som |
316     |             skal benyttes i S.O.R. iterationsskemaet for    |
317     |             Neumann randbetingelserne.                      |
318     |                                                             |
319     |             Det er xf0, xf1, xg0, xg1 og b_0                |
320     |                                                             |
321     |             Naar de fire foerste konstanter beregnes benyttes |
322     |             det konkrete problems randvaerdifunktioner, som er |
323     |             defineret i class PROBLEM.                      |
324     |                                                             |
325     |             Dernaest beregnes  den naeste iteration, for x = 0 |
326     |             til det j'te tidsskridt. Resultatet gemmes i    |
327     |             matricens u's element(0,j).                     |
328     |                                                             |
329     |             Til sidst beregnes den absolutte fejl mellem denne |
330     |             og den forrige iteration, hvorefter den sammenlig- |
331     |             nes med de oevrige fejl, som er fremkommet ved  |
332     |             dette iterationsskridt. Saaledes gemmes den maksi- |
333     |             male fejl i objektet eval.                      |
334     |                                                             |
335     |                                                             |
336     |             Alle disse beregninger er indsat i en uendelig  |
337     |             while loekke, idet objektet eval soerger for at |
338     |                   terminere processen.                      |
339     |                                                             |
340     |             - i Neumann_left overgives processen til objek- |
341     |               tet equation, som er af typen Interior, her vare- |
342     |               tages, som beskrevet ovenfor beregningen - itera- |
343     |               tionen i de indre punkter i = 1 ... spacesteps-1. |
344     |                                                             |
345     | Kalder    : resume(equation).                               |
346     |                                                             |
347     | Kaldes af : objektet oprettes af saetningerne i class SOLVER, |
348     |             men da det er en Coroutine class Interior, vil ob- |
349     |             jektet detachere med det samme. Selve beregning- |
350     |             ne udfoeres naar det kaldes af:                 |
351     |                                                             |
352     |             Coroutine class Evaluation.                     |
353     |                                                             |
354     +-------------------------------------------------------------;
355
356     Coroutine class Neumann_left(L);
357     ref(Left) L;
358     begin
359       long real xf0, xf1, xg0, xg1, b_0;    !konstanter;
360       long real new_value, err;
```

```
361       integer j;                              !lokal tid;
362
363       while TRUE do
364       begin
365
366         j   := eval.j;
367
368         xf0 := delta_x * L.f(j*delta_t - delta_t);
369         xf1 := delta_x * L.f(j*delta_t);
370         xg0 := delta_x * L.g(j*delta_t - delta_t);
371         xg1 := delta_x * L.g(j*delta_t);
372
373         b_0 := u.elm(1,j-1) + C2 * u.elm(0,j-1)
374                -xf0 * u.elm(0,j-1) - xg0 - xg1;
375
376         while j = eval.j do
377         begin
378           new_value  := omega /(C1 + xf1) *  (u.elm(1,j) + b_0) +
379                         u.elm(0,j) - omega * u.elm(0,j);
380           err        := abs(u.elm(0,j) - new_value);
381           u.elm(0,j) := new_value;
382
383           eval.err := err;
384           resume(equation);
385         end **** while j = eval.j ****;
386       end *** while TRUE do ***;
387     end ** Neumann_left **;
388
389
390     !-------------------------------------------------------------+
391     | Navn      : Coroutine class Neumann_right                   |
392     |                                                             |
393     | Omgivelser: Matricen u, som oprettes i saetningerne i class |
394     |             SOLVER.                                         |
395     |             long real delta_x, delta_t  - globale variable. |
396     |             long real omega             - global  variabel. |
397     |                                                             |
398     | Uddata    : I matricen u's elementer indsaettes til en givet|
399     |             iteration de paagaeldende funktionsvaerdier for |
400     |             x = 0 og til ethvert t.                         |
401     |                                                             |
402     | Funktion  : Foerst opdateres tiden, ved hjaelp af eval.j, der|
403     |             fungerer som processens "ur".                   |
404     |                                                             |
405     |             Naar dette er gjort beregnes de konstanter, som |
406     |             skal benyttes i S.O.R. iterationsskemaet for    |
407     |             Neumann randbetingelserne.                      |
408     |                                                             |
409     |             Det er xf0, xf1, xg0, xg1 og b_ss               |
410     |                                                             |
411     |             Naar de fire foerste konstanter beregnes benyttes|
412     |             det konkrete problems randvaerdifunktioner, som er|
413     |             defineret i class PROBLEM.                      |
414     |                                                             |
415     |             Dernaest beregnes  den naeste iteration, for x = 1|
416     |             til det j'te tidsskridt. Resultatet gemmes i    |
417     |             matricens u's element(spacesteps,j).            |
418     |                                                             |
419     |             Til sidst beregnes den absolutte fejl mellem denne|
420     |             og den forrige iteration, hvorefter den sammenlig-|
421     |             nes med de oevrige fejl, som er fremkommet ved  |
```

```
422     |             dette iterationsskridt. Saaledes gemmes den maksi- |
423     |             male fejl i objektet eval.                         |
424     |                                                                |
425     |                                                                |
426     |             Alle disse beregninger er indsat i en uendelig     |
427     |             while loekke, idet objektet eval soerger for at    |
428     |             terminere processen.                               |
429     |                                                                |
430     |             - i Neumann_right overgives processen til ob-      |
431     |               jektet eval, som er af typen Evaluation, her va- |
432     |               retager kontrollen med hvornaar iterationen      |
433     |               et givet tidsskridt skal stoppe samt med hvornaar|
434     |               ligningen er loest til alle tidsskridt.          |
435     |                                                                |
436     | Kalder    : resume(eval).                                      |
437     |                                                                |
438     | Kaldes af : objektet oprettes af saetningerne i class SOLVER,  |
439     |             men da det er en Coroutine class Interior, vil ob- |
440     |             jektet detachere med det samme. Selve beregning-   |
441     |             ne udfoeres naar det kaldes af:                    |
442     |                                                                |
443     |             Coroutine class Interior.                          |
444     |                                                                |
445     +----------------------------------------------------------------;
446
447     Coroutine class Neumann_right(R);
448     ref(Right) R;
449     begin
450       long real xf0, xf1, xg0, xg1, b_ss;   !konstanter;
451       long real new_value, err;
452       integer j;                !lokal tid;
453       integer ss;
454
455       ss := spacesteps;
456
457       while TRUE do
458       begin
459
460         j   := eval.j;
461
462         xf0 := delta_x * R.f(j*delta_t - delta_t);
463         xf1 := delta_x * R.f(j*delta_t);
464         xg0 := delta_x * R.g(j*delta_t - delta_t);
465         xg1 := delta_x * R.g(j*delta_t);
466
467         b_ss := u.elm(ss-1,j-1) + C2 * u.elm(ss,j-1)
468                 + xf0 * u.elm(ss,j-1) + xg0 + xg1;
469
470         while j = eval.j do
471         begin
472           new_value   := omega/(C1 - xf1) *  (u.elm(ss-1,j) + b_ss)
473                          + u.elm(ss,j) - omega * u.elm(ss,j);
474           err         := abs(u.elm(ss,j) - new_value);
475           u.elm(ss,j) := new_value;
476
477           if err > eval.err then eval.err := err;
478
479           resume(eval);
480         end **** while j = eval.j ****;
481       end *** while TRUE do ***;
482     end ** Neumann_right **;
```

```
483
484
485     !-------------------------------------------------------------+
486     | Navn      : Coroutine class Dirichlet_left                  |
487     |                                                             |
488     | Omgivelser: Matricen u, som oprettes i saetningerne i class |
489     |             SOLVER.                                         |
490     |                                                             |
491     | Uddata    : I matricen u's elementer indsaettes funktions-  |
492     |             vaerdien til x = 0 og ethvert t.                |
493     |                                                             |
494     | Funktion  : Foerst opdateres tiden, ved hjaelp af eval.j, der |
495     |             fungerer som processens "ur".                   |
496     |                                                             |
497     |             Dernaest beregnes funktionsvaerdien til det j'te |
498     |             tidsskridt for x = 0. Dette beregnes paa baggrund |
499     |             af de i dette problem definerede funktioner i   |
500     |             class PROBLEM.                                  |
501     |                                                             |
502     |             Da den rigtige funktionsvaerdi kan beregnes direkte|
503     |             uden brug af iteration, beregnes vaerdien kun en |
504     |             gang for hvert tidsskridt.                      |
505     |                                                             |
506     |             For at denne klasse skal kunne samarbejde med de |
507     |             klasser, som har iterationsskemaer, konstrueres en |
508     |             tom while loekke, som soerger for at procces bliver|
509     |             viderebragt til Coroutine class interior.       |
510     |             Som i de oevrige Coroutiner vil objektet eval soer-|
511     |             ge for at processen termineres.                 |
512     |                                                             |
513     |                                                             |
514     |             - i Dirichlet_left overgives processen til objek- |
515     |               tet equation, som er af typen Interior, her vare-|
516     |               tages, som beskrevet ovenfor beregningen - itera-|
517     |               tionen i de indre punkter i = 1 ... spacesteps-1.|
518     |                                                             |
519     | Kalder    : resume(equation).                               |
520     |                                                             |
521     | Kaldes af : objektet oprettes af saetningerne i class SOLVER, |
522     |             men da det er en corutine class Interior, vil ob- |
523     |             jektet detacheres med det samme. Selve beregning- |
524     |             ne udfoeres naar det kaldes af:                 |
525     |                                                             |
526     |             Coroutine class Evaluation.                     |
527     |                                                             |
528     +-------------------------------------------------------------+;
529
530     Coroutine class Dirichlet_left(L);
531     ref(Left) L;
532     begin
533       integer j;                !lokal tid;
534
535       while TRUE do
536       begin
537
538         j := eval.j;
539
540         u.elm(0,j) := L.f(j*delta_t);
541
542         while j = eval.j do
543         begin
```

```
544           resume(equation);
545         end **** while j = eval.j ****;
546       end *** while TRUE do ***;
547     end ** Dirichlet_left **;
548
549
550     !-------------------------------------------------------------+
551     | Navn       : Coroutine class Dirichlet_right                |
552     |                                                             |
553     | Omgivelser: Matricen u, som oprettes i saetningerne i class |
554     |             SOLVER.                                         |
555     |                                                             |
556     | Uddata     : I matricen u's elementer indsaettes funktions- |
557     |             vaerdien til x = 1 og ethvert t.                |
558     |                                                             |
559     | Funktion   : Foerst opdateres tiden, ved hjaelp af eval.j, der |
560     |             fungerer som processens "ur".                   |
561     |                                                             |
562     |             Dernaest beregnes funktionsvaerdien til det j'te |
563     |             tidsskridt for x = 1. Dette beregnes paa baggrund |
564     |             af de i dette problem definerede funktioner i   |
565     |             class PROBLEM.                                  |
566     |                                                             |
567     |             Da den rigtige funktionsvaerdi kan beregnes direkte|
568     |             uden brug af iteration, beregner vi kun vaerdien en|
569     |             gang for hvert tidsskridt.                      |
570     |                                                             |
571     |             For at denne klasse skal kunne samarbejde med de |
572     |             klasser, som har iterationsskemaer, laver vi en |
573     |             tom while loekke, som soerger for at processen bli-|
574     |             ver viderebragt til Coroutine class interior.   |
575     |             Som i de oevrige Coroutiner vil objektet eval soer-|
576     |             ge for at processen termineres.                 |
577     |                                                             |
578     |                                                             |
579     |             - i Dirichlet_right overgives processen til objek- |
580     |               tet eval, som er af typen Evaluation, her vare- |
581     |               retager kontrollen med hvornaar iterationen til |
582     |               et givet tidsskridt skal stoppe samt med hvornaar|
583     |               ligningen er loest til alle tidsskridt.       |
584     |                                                             |
585     | Kalder     : resume(eval).                                  |
586     |                                                             |
587     | Kaldes af  : objektet oprettes af saetningerne i class SOLVER, |
588     |             men da det er en Coroutine class Interior, vil ob- |
589     |             jektet detachere med det samme. Selve beregning- |
590     |             ne udfoeres naar det kaldes af:                 |
591     |                                                             |
592     |             Coroutine class Interior.                       |
593     |                                                             |
594     +-------------------------------------------------------------+;
595
596     Coroutine class Dirichlet_right(R);
597     ref(Right) R;
598     begin
599       integer j;                !lokal tid;
600
601       while TRUE do
602       begin
603
604         j := eval.j;
```

```
605
606         u.elm(spacesteps,j) := R.f(j*delta_t);
607
608         while j = eval.j do
609         begin
610           resume(eval);
611         end **** while j = eval.j do ****;
612       end *** while TRUE do ***;
613     end ** Dirichlet_left **;
614
615
616     !-------------------------------------------------------------+
617     | Navn      : Coroutine class Evaluation                      |
618     |                                                             |
619     | Omgivelser: Matricen u, som oprettes i saetningerne i class |
620     |             SOLVER.                                         |
621     |             integer timesteps, spacesteps   - globale variable.|
622     |             integer cursor_interval         - global  variabel |
623     |             long real tolerance             - global  variabel.|
624     |                                                             |
625     | Uddata    : Saetter et antal prikker paa skaermen, saa bruge- |
626     |             ren kan se hvor langt beregningen er naaet.     |
627     |                                                             |
628     | Funktion  : Eval soerger for at tiden bliver talt op, naar  |
629     |             den maksimale fejl mellem to iterationsskridt   |
630     |             er mindre end tolerance.                        |
631     |                                                             |
632     |             Foer en ny iterationsserie paabegyndes, kopieres |
633     |             vaerdierne fra foregaaende tidsskridt til dette |
634     |             tidsskridt. Herved starter iterationen et fornuf- |
635     |             tigt sted.                                      |
636     |                                                             |
637     |             For hver iteration i serien, nulstilles fejlen og |
638     |             venstre randbetingelse aktiveres igen.          |
639     |             Naar alle processer har itereret en gang, afgoeres|
640     |             det om fejlen er lille nok til at tiden kan taelles|
641     |             op.                                             |
642     |                                                             |
643     |             Undervejs saettes "noof_waitpoints" punkter paa |
644     |             skaermen.                                       |
645     |                                                             |
646     |             Naar sluttiden er naaet, terminerer eval, idet den |
647     |             oenskede loesning er indsat i matricen u.       |
648     |                                                             |
649     | Kalder    : resume(leftbound).                              |
650     |                                                             |
651     | Kaldes af : objektet oprettes af saetningerne i class SOLVER, |
652     |             men da det er en Coroutine class Interior, vil ob- |
653     |             jektet med det samme sige detach. Selve loekken |
654     |             udfoeres foerste gang naar objektet kaldes fra: |
655     |                                                             |
656     |             Procedure calculus                              |
657     |                                                             |
658     |             Herefter aktiveres eval af:                     |
659     |                                                             |
660     |             Coroutine class Rightbound                      |
661     |                                                             |
662     +-------------------------------------------------------------;
663
664     Coroutine class Evaluation;
665     begin
```

```
666       integer i,j;      !Rigtig tid;
667       long real err;
668
669       while j < timesteps do
670       begin
671         j   := j + 1;
672         for i := 0 step 1 until spacesteps do
673           u.elm(i,j) := u.elm(i,j-1);
674
675         if mod(j, cursor_interval) = 0 then
676         begin
677           outchar('.');
678           breakoutimage;
679         end **** if mod() = 0 ****;
680
681         err := 100;                          ! Loekken skal startes;
682
683         while err > tolerance do
684         begin
685           err := 0;
686           resume(leftbound);
687         end **** while err > tolerance ****;
688       end *** while j < timesteps do ***;
689     end ** Evaluation **;
690
691     external procedure diffu_error = "_error";
692
693     !-------------------------------------------------------------+
694     | Navn       : procedure calculus                             |
695     |                                                             |
696     | Omgivelser: ref(Coroutine) Initbound, eval.                 |
697     |                                                             |
698     | Funktion   : Beregningen af begyndelsesvaerdierne igangsaettes, |
699     |              med saetningen resume(initbound). Naar disse er be-|
700     |              regnet aktiveres selve iterationen af de oevrige   |
701     |              punkter med saetningen resume(eval).           |
702     |                                                             |
703     | Kalder     : resume(initbound).                             |
704     |              resume(eval).                                  |
705     |                                                             |
706     | Kaldes af  : Programmerne _numcalc og _anacalc.             |
707     |                                                             |
708     +-------------------------------------------------------------;
709
710
711     procedure calculus;
712     begin
713       resume(initbound);
714       resume(eval);
715     end ** calculus **;
716
717
718     !-------------------------------------------------------------+
719     | Navn       : Klassekrop i class SOLVER                      |
720     |                                                             |
721     | Omgivelser: class PROBLEM.                                  |
722     |                                                             |
723     | Funktion   : Erklaerer og opretter de objekter, som deltager i |
724     |              iterationsprocessen.                           |
725     |                                                             |
726     |              Derudover erklaerer og definerer klassen alle de   |
```

```
727     |            "konstanter", der anvendes i SOR iterationen.      |
728     |                                                                |
729     |            Det er :                                            |
730     |                                                                |
731     |               - delta_x, stoerrelsen af et stedskridt.         |
732     |               - delta_t, stoerrelsen af et tidsskridt.         |
733     |               - pi (benyttes bla ved beregning af omega).      |
734     |               - omega, accelerationsparameter til S.O.R.       |
735     |               - tolerance, hvor lille skal fejlen vaere mellem|
736     |                           to iterationsskridt.                 |
737     |               - C1, C2, konstanter i S.O.R. iterationsskema.   |
738     |               - noof_waitpoints, antal prikker paa skaermen, i|
739     |                                  loebet af hele beregningen.   |
740     |               - cursor_interval, hvor mange tidsskridt mellem  |
741     |                                  hver prik.                    |
742     |               - filename, det navn som loesningen skal gemmes  |
743     |                           under. Det saettes pr default til    |
744     |                           "diffusion".                         |
745     |                                                                |
746     |            Det testes om parametrene alpha, beta, timesteps,   |
747     |            spacessteps og final_time har tilladte vaerdier.    |
748     |            Har de ikke det termineres programmet med en pas-   |
749     |            sende meddelelse i external procedure diffu_error.  |
750     |                                                                |
751     +----------------------------------------------------------------;
752
753     ref(Init)   Initfunc;
754     ref(Left)   L;
755     ref(Right)  R;
756
757     ref(Matrix) u;
758
759     ref(Coroutine) initbound, leftbound, rightbound;
760     ref(Evaluation) eval;
761     ref(Interior) equation;
762
763     long real delta_x, delta_t, C1, C2, omega,
764               pi = 3.14159265358979, tolerance =  0.00000001;
765     text filename;
766
767     integer noof_waitpoints;   ! Til udskrift mens der regnes;
768     integer cursor_interval;
769
770
771     if alpha < tolerance**2 then
772        diffu_error("Alpha skal vaere > 0", 2);
773     if beta <  0 then
774        diffu_error("Beta skal vaere >= 0", 2);
775     if final_time < tolerance**2 then
776        diffu_error("Sluttid skal vaere > 0", 2);
777     if timesteps <= 0 then
778        diffu_error("Antal tidsskridt skal vaere > 0", 2);
779     if spacesteps <= 0 then
780        diffu_error("Antal stedskridt skal vaere > 0", 2);
781
782     noof_waitpoints := if timesteps < 10 then timesteps else 10;
783     cursor_interval := timesteps // noof_waitpoints;
784
785
786     Initfunc      :- new Init;
787     L             :- new Left;
```

```
788     R               :- new Right;
789     u               :- new Matrix;
790
791     initbound       :- new Initial_value(Initfunc);
792     leftbound       :- if L.Neumann then new Neumann_left(L)
793                                     else new Dirichlet_left(L);
794     rightbound      :- if R.Neumann then new Neumann_right(R)
795                                     else new Dirichlet_right(R);
796
797     eval            :- new Evaluation;
798     equation        :- new Interior;
799
800     delta_x         := 1 / spacesteps;
801     delta_t         := final_time / timesteps;
802
803     C1              := delta_x**2 / delta_t / alpha +
804                        delta_x**2 * beta / alpha / 2 + 1;
805
806     C2              := delta_x**2 / delta_t / alpha -
807                        delta_x**2 * beta / alpha / 2 - 1;
808
809     omega           := 2/(1 + sqrt(1 -  (cos(pi/spacesteps)/C1)**2));
810
811
812     filename        :- copy("diffusion");
813   end * SOLVER *;
814
```

2.3 Numcalc

Programmet numcalc findes i filen **_numcalc.sim**

```
 1
 2  !-------------------------------------------------------------------+
 3  | Navn      : _numcalc                                              |
 4  |                                                                   |
 5  | Omgivelser: class SOLVER                                          |
 6  |                                                                   |
 7  | Uddata    : Programmet gemmer resultatet af den numeriske loes-   |
 8  |             ning paa fil, under et filnavn, som er specificeret   |
 9  |             af brugeren efterfulgt af efternavnet "num".          |
10  |                                                                   |
11  | Funktion  : Foerst kaldes den eksterne procedure get_filename,    |
12  |             som soerger for at brugeren specificerer et filnavn   |
13  |             under hvilket resultatet skal gemmes paa fil.         |
14  |                                                                   |
15  |             Mens ligningen loeses, ved at kalde procedure cal-    |
16  |             culus i class SOLVER, udskrives der lidt tekst samt   |
17  |             noof_waitpoints punkter paa skaermen.                 |
18  |                                                                   |
19  |             Naar loesningen er regnet faerdig, bliver den, hvis   |
20  |             det er muligt gemt paa fil, under navnet:             |
21  |                                                                   |
22  |             "filename".num.                                       |
23  |                                                                   |
24  |             Da dette tager en del tid, faar brugeren besked her-  |
25  |             om.                                                   |
26  |                                                                   |
27  |             Til sidst modtager brugeren en besked om, at der nu   |
28  |             kan genereres grafer, ved at koere programmet graph-  |
29  |             make.                                                 |
30  |                                                                   |
31  | Kalder    : -procedure calculus i class SOLVER.                   |
32  |             -text procedure get_filename, indlaeser et filnavn fra|
33  |                       tastaturet.                                 |
34  |             -procedure diffu_error, udskriver fejlmeddeleser.     |
35  |                                                                   |
36  | Kaldes af : Styrefilen numcalc.                                   |
37  |                                                                   |
38  +-------------------------------------------------------------------+;
39
40  begin
41    external class SOLVER = "_solver";
42
43
44
45    SOLVER begin
46
47      external text procedure get_filename = "_get_filename";
48      external procedure diffu_error = "_error";
49
50
51      outtext("!12!");  ! Clear screen ;
52      outimage;
53
54      filename :- get_filename(filename);
55
```

```
 56      outimage;
 57      outtext("Alpha er ");
 58      outreal(alpha,16,0);
 59      outtext("    Delta x er ");
 60      outreal(delta_x,16,0);
 61      outimage;
 62      outtext("Beta er  ");
 63      outreal(beta,16,0);
 64      outtext("    Delta t er ");
 65      outreal(delta_t,16,0);
 66      outimage;
 67      outimage;
 68      outtext("Antal stedskridt: ");
 69      outint(spacesteps,7);
 70      outimage;
 71      outtext("Antal tidsskridt: ");
 72      outint(timesteps,7);
 73      outimage;
 74      outtext("Sluttid:          ");
 75      outreal(final_time,16,0);
 76      outimage;
 77      outimage;
 78
 79      if timesteps < 10 then
           diffu_error("Det anbefales at tidskridt >= 10,1");
 80      !  Dette er kun en warning  ;
 81
 82      outtext("Der saettes et punktum for hvert ");
 83      outint(cursor_interval,0);
 84      outtext(" tidsskridt.");
 85      outimage;
 86      outimage;
 87      outtext("Regner ");
 88      breakoutimage;
 89
 90      calculus;
 91
 92      outtext(" Faerdig !!!");
 93      outimage;
 94      outimage;
 95      outtext("Resultater gemmes - vent et oejeblik ");
 96      outimage;
 97
 98      if not u.save(filename & ".num") then
 99        diffu_error("Filen " & filename & ".num kan ikke gemmes.", 2) else

100      begin
101        outimage;
102        outimage;
103        outtext("Resultaterne af den numeriske beregning er i filen ");
104        outtext(filename & ".num");
105        outimage;
106        outimage;
107        outtext("Udfoer graphmake "& filename &" for evaluering af
             data.");
108        outimage;
109        outimage;
110      end *** resultatet gemmes paa fil ***;
111    end ** SOLVER **;
112  end * numcalc *;
```

2.4 Anacalc1

Programmet anacalc1 findes i filen **_anacal1.sim**

```
!-------------------------------------------------------------------+
| Navn        : _anacalc                                            |
|                                                                   |
| Omgivelser : class SOLVER                                         |
|                                                                   |
| Uddata      : Tre filer med data for de beregnede loesninger, en for |
|               den numeriske - <filnavn>.num, en for den analytiske - |
|               <filnavn>.ana, og en for differencen mellem disse loes-|
|               loesninger - <filnavn>.dif.                         |
|                                                                   |
| Funktion    : Loesning af ligningen :                             |
|                                                                   |
|               du              d(du)                               |
|               --(x,t) -  alpha-----(x,t) = 0                      |
|               dt              d(dx)                               |
|                                                                   |
|               Med randbetingelserne :                             |
|                                                                   |
|               du                                                  |
|               --(0, t) =  u(0,t)                                  |
|               dx                                                  |
|                                                                   |
|               du                                                  |
|               --(1, t) =  -u(0,t)                                 |
|               dx                                                  |
|                                                                   |
|               Og begyndelsesbetingelsen :                         |
|                                                                   |
|               u(x,0) = 1                                          |
|                                                                   |
|               Brugeren vaelger et basisfilnavn til loesningerne.  |
|                                                                   |
|               Loesningen sker foerst numerisk, ved at kalde procedure|
|               calculus i class SOLVER. Derefter loeses den analytisk,|
|               ved at kalde procedure solution. Naar dette er gjort |
|               gemmes resultatet paa fil. Til sidst sammenlignes disse|
|               resultater, ved at kalde procedure minus, som ligger i |
|               matricen u. Det er den matrix, som oprettes naar lig-|
|               ningen loeses numerisk.                            |
|                                                                   |
|               Funktionen virker kun efter hensigten, hvis det er  |
|               ovenstaaende problem, som er defineret i class PROBLEM.|
|                                                                   |
| Kalder      : Procedure calculus i class SOLVER,                  |
|               procedure minus i class Matrix -den numeriske loesning.|
|               procedure solution,                                 |
|                                                                   |
| kaldes af   : -                                                   |
|                                                                   |
+-------------------------------------------------------------------;

begin
  external class SOLVER = "_solver";
  solver begin
    long real A_0, omega1;
```

```
56      integer i,j;
57      ref(Matrix) u_ana;
58      text file_name;
59      boolean omega;
60
61      !-------------------------------------------------------------------+
62      | Navn       : procedure make_alpha                                 |
63      |                                                                   |
64      | Indata     : long real array alpha, integer N                     |
65      |              long real array initialvalue                         |
66      |                                                                   |
67      | Uddata     : long real array alpha, med de beregnede alpha_n.     |
68      |                                                                   |
69      | Funktion   : Loesning af ligningen :                              |
70      |              alpha tan alpha = 0.5                                |
71      |                                                                   |
72      |              Ligningen loeses ved at bruge Newton Raphson itera-  |
73      |              tions metode. Loesningerne returneres som et antal   |
74      |              vaerdier i array alpha.                              |
75      |                                                                   |
76      | kaldes af  : Procedure solution                                   |
77      |                                                                   |
78      +-------------------------------------------------------------------;
79
80      procedure make_alpha(initialvalue, alpha, N);
81      name alpha;
82      long real array initialvalue;
83      long real array alpha;
84      integer N;
85      begin
86        integer i, iterationcounter;
87        long real array fkt(1:N);
88        long real x, fx, dfx, tolerance = 0.00000001;
89
90        for i := 1 step 1 until N do
91        begin
92          x      := initialvalue(i);
93          fx     := 2 * x * sin (x) - cos (x);
94          dfx    := 3 * sin (x) + 2 * x * cos (x);
95
96          if dfx <> 0 then
97          begin
98            iterationcounter := 0;
99
100           while abs(fx) > tolerance and iterationcounter < 100 do
101           begin
102             x                := x - fx / dfx;
103             fx               := 2 * x * sin (x) - cos (x);
104             dfx              := 3 * sin (x) + 2 * x * cos (x);
105             iterationcounter  := iterationcounter + 1;
106           end ****** while ******;
107
108           alpha(i) := x;
109           fkt(i)   := fx;
110         end ***** dfx <> 0 ***** else
111         begin
112           text fv;
113           fv :- blanks(3);
114           fv.setpos(1);
115           fv.putint(i);
116           diffu_error("dfx = 0 for i = " &
```

```
117                     fv.strip & " i procedure make_alpha",2);
118         end ***** dfx = 0 *****;
119       end **** for, hvor mange nulpunkter skal vi finde ****;
120     end *** Procedure make_alpha ***;
121
122
123     !-------------------------------------------------------------+
124     | Navn       : procedure solution                             |
125     |                                                             |
126     | Indata     : long real array u,                             |
127     |              integer spacesteps, timesteps                  |
128     |              long real delta_x, delta_t                     |
129     |                                                             |
130     | Uddata     : long real array u, med de analytiske loesninger.|
131     |                                                             |
132     | Funktion   : Loesning af den diffusionsligning, som er defineret|
133     |              i kommentaren for hele programmet.             |
134     |                                                             |
135     |              Denne loesning har formen :                    |
136     |                                                             |
137     |                                 sec alpha_n                 |
138     |              u(x,t) = sum ( ------------- exp(-4alpha_n^2 t) * |
139     |                                 (3+4alpha_n^2)              |
140     |                                                             |
141     |                                  cos(2alpha_n(x-0.5) )      |
142     |                                                             |
143     |              hvor alpha_n er de positive roedder i          |
144     |                                                             |
145     |              alpha tan(alpha) = 0.5                         |
146     |                                                             |
147     |                                                             |
148     | Kaldes af  : Hovedprogrammet i _anacalc1.                   |
149     |                                                             |
150     +-------------------------------------------------------------;
151
152     procedure solution(u,spacesteps,timesteps,delta_x,delta_t);
153     name u;
154     long real array u;
155     integer spacesteps, timesteps;
156     long real delta_x, delta_t;
157
158     begin
159
160       integer n, i, j, N_max = 20;
161       long real x, t, new_value;
162       long real array initialvalue(1:N_max), alpha(1:N_max);
163
164       initialvalue(1)  := 0.5;
165       initialvalue(2)  := 3;
166       initialvalue(3)  := 6;
167       initialvalue(4)  := 9;
168       initialvalue(5)  := 12.5;
169       initialvalue(6)  := 15.5;
170       initialvalue(7)  := 18.5;
171       initialvalue(8)  := 22;
172       initialvalue(9)  := 25;
173       initialvalue(10) := 28;
174       initialvalue(11) := 31;
175       initialvalue(12) := 34.5;
176       initialvalue(13) := 37.5;
177       initialvalue(14) := 40.5;
```

```
178       initialvalue(15) := 43.5;
179       initialvalue(16) := 47;
180       initialvalue(17) := 50;
181       initialvalue(18) := 53;
182       initialvalue(19) := 56.5;
183       initialvalue(20) := 59.5;
184
185       make_alpha(initialvalue, alpha, N_max);
186
187       x := t := 0;
188
189       for i := 0 step 1 until spacesteps do u(i,0) := 1;
190
191       for j := 1 step 1 until timesteps do
192       begin
193         t  := j * delta_t;
194
195         if mod(j, cursor_interval) = 0 then
196         begin
197           outchar('.');
198           breakoutimage;
199         end **** if mod ****;
200
201         for i := 0 step 1 until spacesteps do
202         begin
203           x         := i * delta_x;
204           new_value := 0;
205
206           for n := 1 step 1 until N_max do
207           begin
208             new_value := new_value + 1/(3+4*alpha(n)**2)/cos(alpha(n)) *
209                                      exp(-4*alpha(n)**2*t)*
210                                      cos(2*alpha(n)*(x - 0.5));
211           end ***** for *****;
212
213           u(i,j) := 4 * new_value;
214         end **** for i ****;
215       end *** for j ***;
216     end *** procedure solution ***;
217
218     external text procedure get_filename = "_get_filename";
219
220     file_name :- get_filename("diffusion");
221
222     outtext("Beregningen varer ");
223     outint(timesteps,0);
224     outtext(" tidsskridt -  ");
225     outtext(". = ");
226     outint(cursor_interval,0);
227     outtext(" tidsskridt.");
228     outimage;
229     outimage;
230     outtext("Regner numerisk  ");
231     breakoutimage;
232
233     calculus;
234
235     outtext(" Faerdig !!!");
236     outimage;
237     outimage;
238
```

```
239
240       outimage;
241       outtext("Regner analytisk ");
242       breakoutimage;
243
244       u_ana :- new Matrix;
245       solution(u_ana.elm,spacesteps, timesteps, delta_x, delta_t);
246
247       outtext(" Faerdig !!!");
248       outimage;
249       outimage;
250
251
252       outtext("Resultater gemmes - vent et oejeblik ");
253       outimage;
254       outimage;
255
256       if u.save(file_name & ".num") then
257       begin
258         outtext("Numerisk udregnede resultater er gemt som ");
259         outtext(file_name & ".num");
260         outimage;
261       end
262       else error("Kan ikke gemme resultater som "&filename&".num");
263       if u_ana.save(file_name & ".ana") then
264       begin
265         outtext("Analytisk udregnede resultater er gemt som ");
266         outtext(file_name & ".ana");
267         outimage;
268       end
269       else error("Kan ikke gemme resultater som "&filename&".ana");
270       if u.minus(u_ana).save(file_name & ".dif") then
271       begin
272         outtext("Forskellen mellem de to er gemt som ");
273         outtext(file_name & ".dif");
274         outimage;
275       end
276       else error("Kan ikke gemme forskelle som "&filename&".dif");
277     end ** SOLVER **;
278   end * _anacalc1 *;
279
```

2.5 Anacalc3

Programmet anacalc3 findes i filen `_anacal3.sim`

```
!-----------------------------------------------------------------------+
| Navn       : _anacalc                                                 |
|                                                                       |
| Indata     : class SOLVER                                             |
|                                                                       |
| Uddata     : Tre filer med data for de beregnede loesninger, en for   |
|              den numeriske - <filnavn>.num, en for den analytiske -   |
|              <filnavn>.ana, og en for differencen mellem disse loes-  |
|              loesninger - <filnavn>.dif.                              |
|                                                                       |
| Funktion   : Loesning af ligningen :                                  |
|                                                                       |
|              du              d(du)                                    |
|              --(x,t) -  alpha-----(x,t) + u(x,t) = 0                  |
|              dt              d(dx)                                    |
|                                                                       |
|              Med randbetingelserne :                                  |
|                                                                       |
|              du                                                       |
|              --(x=0,1, t) =  sin(5*t)                                 |
|              dx                                                       |
|                                                                       |
|              Og begyndelsesbetingelsen :                              |
|                                                                       |
|              u(x,0) = 0                                               |
|                                                                       |
|              Dette foregaar foerst numerisk, ved at kalde procedure   |
|              calculus i class SOLVER. Derefter loeses den analytisk,  |
|              ved at kalde procedure Neumann. Naar dette er gjort      |
|              gemmes resultatet paa fil. Til sidst sammenlignes disse  |
|              resultater, ved at kalde procedure minus, som ligger i   |
|              matricen u. Det er den matrice, som oprettes naar lig-   |
|              ningen loeses numerisk.                                  |
|                                                                       |
|              Endelig gemmes alle resultater paa filerne med suffix    |
|              *.num, *.ana og *.dif                                    |
|              Kan de ikke aabnes kaldes procedure diffu_error og pro-  |
|              grammet terminerer                                       |
|                                                                       |
|              Funktionen virker kun efter hensigten, hvis det er       |
|              ovenstaaende problem, som er defineret i class PROBLEM.  |
|                                                                       |
|              Indlaesningen af konstanterne A_0 og omega foregaar      |
|              interaktivt med brugeren, og de skal for at faa et rig-  |
|              tigt resultat, stemme overens med konstanterne i class   |
|              PROBLEM.                                                 |
|                                                                       |
|              Alle parametre udskrives paa skaermen.                   |
|                                                                       |
| Kalder     : Procedure calculus i class SOLVER,                       |
|              procedure minus i class Matrix -den numeriske loesning.  |
|              procedure Neumann,                                       |
|              procedure input_const,                                   |
|              procedure diffu_error                                    |
|                                                                       |
```

```
56  | kaldes af  : Styrefilen anacalc                                    |
57  |                                                                    |
58  +--------------------------------------------------------------------;
59
60
61
62  begin
63    external class SOLVER = "_solver";
64
65    solver begin
66      long real A_0, omega;
67      integer i,j;
68      ref(Matrix) u_ana;
69      text file_name;
70
71      external text procedure get_filename = "_get_filename.atr";
72      external procedure diffu_error = "_error.atr";
73
74
75      !-----------------------------------------------------------------+
76      | Navn       : procedure input_const                              |
77      |                                                                 |
78      | Uddata     : De globale konstanter A_0 og omega.                |
79      |                                                                 |
80      | Funktion   : Indlaeser konsterne A_0 og omega interaktivt.      |
81      |                                                                 |
82      | kaldes af  : Hovedprogrammet i _anacalc.                        |
83      |                                                                 |
84      +-----------------------------------------------------------------;
85
86      procedure input_const;
87      begin
88        outimage;
89        outtext("Input A0: ");
90        breakoutimage;
91        A_0   := inreal;
92        outtext("Input Omega: ");
93        breakoutimage;
94        omega := inreal;
95      end *** input_const ***;
96
97
98      !-----------------------------------------------------------------+
99      | Navn       : procedure Neumann                                  |
100     |                                                                 |
101     | Omgivelser : Eksisensen af objektet u_ana.                      |
102     |                                                                 |
103     | Uddata     : Den analytiske loesning returneres i matricen      |
104     |              u_ana.                                             |
105     |                                                                 |
106     | Funktion   : Beregner den analytiske loesning, til ligningen,   |
107     |              som er defineret i den oeverste kommentar.         |
108     |                                                                 |
109     |              Loesningen har formen :                            |
110     |                                                                 |
111     |                                                                 |
112     |                              4A_1 cos mx                        |
113     |              u(x,t) =  sum( ---- ------- ( a sin(omega t)       |
114     |                              pi    m^2                          |
115     |                                                                 |
116     |                     + b cos(omega t) - b exp(-(alpha2 m^2+1)*t))|
```

```
117     |                                                                      |
118     |                                  pi                                  |
119     |                     + A_1 (x - ---) sin(omega t)                     |
120     |                                  2                                   |
121     |                                                                      |
122     |  hvor                                                                |
123     |       m      = 2n-1                                                  |
124     |       A1     = A0/pi                                                 |
125     |       alpha2 = alpha * pi^2                                          |
126     |       omega2 = omega^2                                               |
127     |                                                                      |
128     |                 alpha2 m^2 + omega2 + 1                              |
129     |       a      = --------------------------                            |
130     |                 (alpha2 m^2 + 1)^2 + omega2                          |
131     |                                                                      |
132     |                 alpha2 m^2 omega                                     |
133     |       b      = --------------------------                            |
134     |                 (alpha2 m^2 + 1)^2 + omega2                          |
135     |                                                                      |
136     |  Se ogsaa kapitel om analytisk loesning af diffusionsligningen.      |
137     |                                                                      |
138     |                                                                      |
139     |  Kaldes af   : Hovedprogrammet i _anacalc                            |
140     |                                                                      |
141     +----------------------------------------------------------------------;
142
143     procedure Neumann_calc;
144     begin
145       long real x, t, a, b, err, sum, denominator;
146       long real new_value, alpha2, omega2, omegat;
147       long real tolerance = 0.0000005;
148       integer i, j, n, n2;
149
150       omega2 := omega**2;
151       alpha2 := alpha * (pi**2);
152       A_0    := A_0 / pi;
153
154       for j := 0 step 1 until timesteps do
155       begin
156         t      := j * delta_t;
157         omegat := omega * t;
158
159         if j > 0 and mod(j, cursor_interval) = 0 then
160         begin
161           outchar('.');
162           breakoutimage;
163         end ***** if j ******;
164
165         for i := 0 step 1 until spacesteps do
166         begin
167           x   := i * delta_x * pi;
168           err := 100;
169           sum :=   0;
170           n   :=  -1;
171
172           while err > tolerance do
173           begin
174             n  := n + 2;
175             n2 := n**2;
176             denominator := (alpha2*n2+1)**2 + omega2;
177             a  := alpha2*n2/denominator +
```

```
178                 omega2   /denominator +
179                 1        /denominator;
180           b  := omega*alpha2*n2 / denominator;
181           new_value := sum +
182                        cos(n*x)/n2*
183                        (a * sin(omegat)+
184                         b * cos(omegat)-
185                         b * exp(-alpha2*n2*t - t) );
186           err := abs(sum - new_value);
187           sum := new_value;
188         end ****** while err > tolerance ******;
189
190         u_ana.elm(i,j) := 4*A_0/pi * sum - pi/2* A_0 * sin(omegat)
191                           + A_0*x * sin(omegat);
192
193       end ***** for i *****;
194     end **** for j ****;
195   end *** Neumann ***;
196
197   outtext("!12!");         ! Clear screen;
198   outimage;
199
200   file_name :- get_filename("diffusion");
201
202   input_const;
203
204   outimage;
205   outtext("Alpha er ");
206   outfix(alpha,4,8);
207   outtext("    Delta x er ");
208   outfix(delta_x,4,8);
209   outimage;
210   outtext("Beta er  ");
211   outfix(beta,4,8);
212   outtext("    Delta t er ");
213   outfix(delta_t,4,8);
214   outimage;
215
216
217   outimage;
218   outtext("Antal stedskridt: ");
219   outint(spacesteps,4);
220   outimage;
221   outtext("Antal tidsskridt: ");
222   outint(timesteps,4);
223   outimage;
224   outtext("Sluttid:          ");
225   outfix(final_time,4,9);
226   outimage;
227   outimage;
228
229   outtext("Der saettes et punktum for hvert ");
230   outint(cursor_interval,0);
231   outtext(" tidsskridt.");
232   outimage;
233   outimage;
234   outtext("Regner numerisk");
235   breakoutimage;
236
237   calculus;
238
```

```
239       outtext(" Faerdig !!!");
240       outimage;
241       outimage;
242
243       outtext("Regner analytisk ");
244       breakoutimage;
245
246       u_ana :- new Matrix;
247       Neumann_calc;
248
249       outtext(" Faerdig !!!");
250       outimage;
251       outimage;
252
253
254       outtext("Resultater gemmes - vent et oejeblik ");
255       outimage;
256       outimage;
257
258       if u.save(file_name & ".num") then
259       begin
260         outtext("Numerisk udregnede resultater er gemt som ");
261         outtext(file_name & ".num");
262         outimage;
263       end
264       else diffu_error("Kan ikke gemme resultater som "&filename&".num",2);
265       if u_ana.save(file_name & ".ana") then
266       begin
267         outtext("Analytisk udregnede resultater er gemt som ");
268         outtext(file_name & ".ana");
269         outimage;
270       end
271       else diffu_error("Kan ikke gemme resultater som "&filename&".ana",2);
272       if u.minus(u_ana).save(file_name & ".dif") then
273       begin
274         outtext("Forskellen mellem de to er gemt som ");
275         outtext(file_name & ".dif");
276         outimage;
277       end
278       else diffu_error("Kan ikke gemme forskelle som "&filename&".dif",2);
279       outimage;
280     end ** SOLVER **;
281   end * _anacalc *;
282
```

2.6 Fysik

Programmet fysik findes i filen `_fysik.sim`

```
external class PROBLEM  = "problem";


!-------------------------------------------------------------------+
| Navn       : _fysik.sim                                           |
|                                                                   |
| Omgivelser : Da programmet er praefixet med problem er alle parame-|
|              som optraeder heri kendt.                            |
|                                                                   |
| Uddata     : Programmet tilfoejer                                 |
|                                                                   |
|                - en vinkelfrekvens                                |
|                - impedansen  til den givne frekvens               |
|                - admittansen til den givne frekvens               |
|                                                                   |
|              De tre tal laegges paa filen efter hinanden adskilt med|
|              et mellemrum                                         |
|                                                                   |
| Funktion   : Beregner henholdsvis realdelen af impedansen og imagi-|
|              delen af admittansen til en givet vinkelfrekvens.    |
|                                                                   |
|              Realdelen af impedansen er givet ved :               |
|                                                                   |
|                  const * cos(phase) / Amplitude                   |
|                                                                   |
|              Imaginaerdelen af admittansen er givet ved :         |
|                                                                   |
|                  Amplitude * sin(phase) / const                   |
|                                                                   |
|              Idet :                                               |
|                                                                   |
|              konstanten const er -                                |
|                                                                   |
|                const := Area * E_0 * epsilon * anglefreq          |
|              faseforskellen phase er -                            |
|                                                                   |
|                forskellen mellem spaendingen V og en ren sinus.   |
|                                                                   |
|              Amplituden Amplitude er -                            |
|                                                                   |
|                Amplituden i spaendingen V.                        |
|                                                                   |
|              Spaendingen V er givet ved                           |
|                                                                   |
|                    L    x                                         |
|                int  int  rho(x',t) dx                             |
|                   x    0                                         |
|                                                                   |
|              Hvor rho er givet ud fra loesningen til diffusionslig-|
|              ningen :                                             |
|                                                                   |
|                d rho              d(d rho)                        |
|                -----(x,t) - alpha -------- (x,t) + beta rho(x,t) = 0|
|                 d t                d(d x)                         |
|                                                                   |
```

```
56  |              Programmet indlaeser det dimensionsloese resultat og   |
57  |              skalerer dette, saa det faar de rigtige dimensioner.   |
58  |              Derefter beregnes spaendingen V, og til sidst henholds-|
59  |              vis impedans og admittans.                              |
60  |                                                                      |
61  | Kaldes af   : Styrefilen EIS                                         |
62  |                                                                      |
63  +----------------------------------------------------------------------;
64  PROBLEM begin
65
66
67
68    !--------------------------------------------------------------------+
69    | Navn       : class Filematrix                                      |
70    |                                                                    |
71    | Inddata    : text filename, integer spacesteps, timesteps          |
72    |                                                                    |
73    | Omgivelser: Filen ved navn filename skal eksistere fysisk.         |
74    |                                                                    |
75    | Uddata     : class Filematrix returnerer todimensionale matricer   |
76    |              med vaerdier, fra en indlaest fil. Derudover returne- |
77    |              res der en boolean exist, som fortaeller om indlaes-  |
78    |              ningen af filerne gik godt.                           |
79    |                                                                    |
80    | Funktion   : Klassen erklaerer et todimensionelt array, med dimen- |
81    |              sionerne spacesteps og timesteps. I arrayet indsaettes|
82    |              vaerdierne fra den fysiske fil, med navnet filename.  |
83    |                                                                    |
84    |              Da filen, som gerne skulle vaere en loesning til en   |
85    |              givet diffusionsligning, er gemt paa foelgende format |
86    |                                                                    |
87    |              - integer spacesteps                                  |
88    |              - integer timesteps                                   |
89    |              - long real final_time                                |
90    |              - vaerdierne i loesningsmatricen, gemt under hinanden |
91    |                raekkevis.                                          |
92    |                                                                    |
93    |              Det undersoeges dernaest om parametrene spacesteps og |
94    |              timesteps stemmer overens, med de vaerdier,som er gemt|
95    |              paa den fysiske fil. Dernaest hentes konstanten       |
96    |              final_time inden den erklaerede "matrix" fyldes, med  |
97    |              vaerdierne fra loesningen til en diffusionsligning.   |
98    |                                                                    |
99    |              Til sidst saettes den logiske variable exist = TRUE / |
100   |              FALSE for at fortaelle omverdenen om indlaesningen gik|
101   |              godt.                                                 |
102   |                                                                    |
103   | Kaldes af  : Hovedprogrammet i _graphmake.                         |
104   |                                                                    |
105   +--------------------------------------------------------------------;
106
107   class filematrix(filename, spacesteps, timesteps);
108   text filename;
109   integer spacesteps, timesteps;
110   begin
111     ref(infile) file;
112     boolean exists;
113     long real final_time;
114     long real array elm(0:spacesteps, 0:timesteps);
115
116     file :- new infile(filename);
```

```
117     inspect file do
118     begin
119       if open(blanks(100)) and then
120          spacesteps = inint and then
121          timesteps  = inint then
122       begin
123         integer i, j;
124         final_time := inreal;
125
126         for j := 0 step 1 until timesteps do
127           for i := 0 step 1 until spacesteps do
128             elm(i, j) := inreal;
129
130         exists := true;
131       end **** spacestep og timestep ok ****;
132       close;
133     end *** inspect ***;
134   end ** filematrix **;
135
136
137
138   !-------------------------------------------------------------------+
139   | Navn       : procedure findphase                                  |
140   |                                                                   |
141   | Inddata    : long real array V og long real Amplitude             |
142   |                                                                   |
143   | Omgivelser : De globale konstanter delta_t og timesteps samt fre- |
144   |              kvensen anglefreq.                                   |
145   |                                                                   |
146   | Uddata     : Faseforskellen returneres i findphase, da det er     |
147   |              en funktionsprocedure.                               |
148   |                                                                   |
149   | Funktion   : proceduren finder faseforskellen - phi,  mellem      |
150   |              spaendingen                                          |
151   |                                                                   |
152   |                      sin(omega t + phi)                           |
153   |                                                                   |
154   |              som er repraesenteret i arrayet V, og en ren si-     |
155   |              nus funktion                                         |
156   |                                                                   |
157   |                       sin(omega t)                                |
158   |                                                                   |
159   |              Da spaendingen har en indsvingningstid, findes       |
160   |              faseforskellen, efter halvdelen af det antal sving-  |
161   |              ninger, som er repraesenteret i V.                   |
162   |                                                                   |
163   |              Faseforskellen findes herefter ved beregningen       |
164   |                                                                   |
165   |                  Amplitude sin(omega t + phi) = f                 |
166   |                                                                   |
167   |              Vi finder da et punkt hvori sin(omega t) = 0, hvil-  |
168   |              ket betyder at sin(omega t + phi) = sin(phi), og     |
169   |              derfor vil                                           |
170   |                                                                   |
171   |                    Amplitude sin(phi) = f          <=>            |
172   |                    phi = arcsin(f / Amplitude)                    |
173   |                                                                   |
174   | kaldes af : Saetningerne i _fysik.sim                             |
175   |                                                                   |
176   +-------------------------------------------------------------------;
177
```

```
178   long real procedure findphase(V, Amplitude);
179   long real array V;
180   long real Amplitude;
181   begin
182     integer noof_periods, j;
183     long real period;
184     boolean posgrad;
185
186     Noof_periods := entier(delta_t * timesteps * anglefreq / 2 / pi);
187     j            := (Noof_periods - 2) * 2*pi / anglefreq / delta_t;
188     if j < 0 or j > timesteps then
189     begin
190       diffu_error("Indexfejl i findphase",2)
191     end else
192     begin
193       posgrad := V(j+1) - V(j) >= 0;
194       findphase := if posgrad
195         then arcsin( V(j) / Amplitude)
196         else pi - arcsin( V(j) / Amplitude);
197     end;
198   end ** findphase **;
199
200   !-------------------------------------------------------------------+
201   | Navn       : procedure graf                                       |
202   |                                                                   |
203   | Inddata    : long real array V                                    |
204   |                                                                   |
205   | Omgivelser : Den globale konstant timesteps                       |
206   |                                                                   |
207   | Uddata     : Graffil med den interne spaending.                   |
208   |                                                                   |
209   | Funktion   : proceduren udskriver en fil med vaerdier for V       |
210   |              i et format, der kan laeses af xgraph.               |
211   |                                                                   |
212   | kaldes af : Saetningerne i _fysik.sim                             |
213   |                                                                   |
214   +-------------------------------------------------------------------;
215
216   procedure graf(V); long real array V;
217   begin
218     ref(outfile) file;
219     integer j,f;
220     text suffix;
221
222     f := anglefreq/2/pi;
223
224     suffix :- blanks(Entier(ln(f)/ln(10)+1));
225     suffix.putint(f);
226     lowten('E');
227     file :- new outfile(filename & ".V" & suffix);
228     inspect file do
229     begin
230       open(blanks(80));
231       outtext("TitleText: V (" & suffix & " Hz)");
232       outimage;
233       outimage;
234       outtext("""V");
235       outimage;
236       for j := 0 step 1 until timesteps do
237       begin
238         outreal(j*delta_t,16,0);
```

```
239         outchar(' ');
240         outreal(V(j),16,0);
241         outimage;
242       end;
243       close;
244     end;
245   end;
246
247
248   external text procedure get_filename  = "_get_filename";
249   external procedure diffu_error        = "_error";
250
251
252   ref(filematrix) rho;
253   ref(outfile) frequency;
254
255   text filename;
256   long real delta_x, delta_t, Amplitude, phase, const, ReZ, ImY, ReY;
257   integer  i, j;
258   long real array V(0:timesteps);
259
260
261   filename :- copy("diffusion");
262
263   outtext("!12!");
264   outimage;
265
266   filename :- get_filename(filename);
267
268   delta_x := Length / spacesteps;
269   delta_t := epsilon / sigma * final_time / timesteps;
270
271   rho :- new filematrix(filename & ".num", spacesteps, timesteps);
272
273   if not rho.exists then
274     diffu_error("Filen "&filename&".num kan ikke aabnes", 2)
275   else
276   begin
277     inspect rho do
278     begin
279       for i := 0 step 1 until spacesteps do
280         for j := 0 step 1 until timesteps do
281           elm(i,j) := rho_0 / epsilon * elm(i,j);
282
283       for j := 0 step 1 until timesteps do
284       begin
285         V(j) := 0;
286         for i := 0 step 1 until spacesteps do
287           V(j) := V(j) + spacesteps*elm(i,j) - i*elm(i,j) * elm(i,j);
288         V(j) := -delta_x**2 * V(j);
289       end **** for j ****;
290
291       Amplitude := abs( V(timesteps));
292
293       for j := timesteps step -1 until timesteps / 2 do
294         if abs(V(j)) > Amplitude then Amplitude := abs(V(j));
295
296       phase := findphase(V, Amplitude);
297       const := Area * E_0 * epsilon * anglefreq;
298       ReZ   := Amplitude * sin(phase) / const;
299       ImY   := const * cos(phase) / Amplitude / anglefreq;
```

```
300         ReY   := const * sin(phase) / Amplitude;
301
302         frequency :- new outfile(filename & ".data");
303
304         inspect frequency do
305         begin
306           setaccess("APPEND");
307
308           if not open(blanks(200)) then
309             diffu_error("Filen i regne tilbage kan ikke aabnes", 2)
310           else
311           begin
312             lowten('E');
313             outreal(anglefreq, 16, 0);
314             outtext(" ");
315             outreal(ReZ, 16, 0);
316             outtext(" ");
317             outreal(ImY, 16, 0);
318             outtext(" ");
319             outreal(ReY, 16, 0);
320             outtext(" ");
321             outreal(phase, 16, 0);
322             outimage;
323             close;
324           end ***** if *****;
325       end **** inspect frequency ****;
326     end *** inspect rho ***;
327     graf(V);
328   end ** if  **;
329 end * _fysik *;
```

2.7 View

Programmet view findes i filen `_view.sim`

```
1
2  !-------------------------------------------------------------------+
3  | Navn       : _view                                                |
4  |                                                                   |
5  | Omgivelser: Filer med navnene *.num *.ana og *.dif i det ak-      |
6  |             tuelle katalog.                                       |
7  |                                                                   |
8  | Uddata     : Udskrift af oenskede filer.                          |
9  |                                                                   |
10 | Funktion   : Udskriver matricer paa skaermen.                     |
11 |                                                                   |
12 |              Foerst skal brugeren vaelge et filnavn. Programmet ind-|
13 |              laeser derpaa filerne <filnavn>.num, <filnavn>.ana og |
14 |              <filnavn>.dif hvis de findes.                        |
15 |              Hvis der findes mere end *.num kan brugeren vaelge,  |
16 |                                                                   |
17 |              hvilke filer han oensker at udskrive.                |
18 |                                                                   |
19 |                                                                   |
20 |                                                                   |
21 | Kalder     : Kalder foelgende klasse:                             |
22 |                                                                   |
23 |              - class Filematrix, sammenknytter en fil med et array.|
24 |                                                                   |
25 |              Kalder foelgende procedurer:                         |
26 |                                                                   |
27 |              - boolean procedure get_dimension, Laeser loesningens|
28 |                       dimensioner, til brug for Filematrix objekter.|
29 |              - text procedure get_filename, faar filnavn fra brugeren|
30 |              - long real procedure getreal, indlaeser reelle tal. |
31 |              - write, udskriver en matrix.                        |
32 |              - booelan procedure contain, checker om et tegn er i en|
33 |                       tekst.                                      |
34 |                                                                   |
35 | Kaldes af : Styrefilen view.                                      |
36 |                                                                   |
37 +-------------------------------------------------------------------;
38
39 begin
40
41
42   !-----------------------------------------------------------------+
43   | Navn       : class Filematrix                                   |
44   |                                                                 |
45   | Inddata    : text filename, integer spacesteps, timesteps       |
46   |                                                                 |
47   | Omgivelser: Filen ved navn filename skal eksistere fysisk.      |
48   |                                                                 |
49   | Uddata     : class Filematrix returnerer todimensionale matricer|
50   |              med vaerdier, fra en indlaest fil. Derudover returne-|
51   |              res der en boolean exist, som fortaeller om indlaes-|
52   |              ningen af filerne gik godt.                        |
53   |                                                                 |
54   | Funktion   : Klassen erklaerer et todimensionelt array, med dimen-|
55   |              sionerne spacesteps og timesteps. I arrayet indsaettes|
```

```
56   |            vaerdierne fra den fysiske fil, med navnet filename.  |
57   |                                                                  |
58   |            Da filen, som gerne skulle vaere en loesning til en   |
59   |            givet diffusionsligning, er gemt paa foelgende format |
60   |                                                                  |
61   |            - integer spacesteps                                  |
62   |            - integer timesteps                                   |
63   |            - long real final_time                                |
64   |            - vaerdierne i loesningsmatricen, gemt under hinanden |
65   |              raekkevis.                                          |
66   |                                                                  |
67   |            Det undersoeges dernaest om parametrerne spacesteps og|
68   |            timesteps stemmer ovrens, med de vaerdier, som er gemt|
69   |            paa den fysiske fil. Dernaest hentes konstanten       |
70   |            final_time inden den erklaerede matrix fyldes, med    |
71   |            vaerdierne fra loesningen til en diffusionsligning.   |
72   |                                                                  |
73   |            Til sidst saettes den logiske variable exist = TRUE / |
74   |            FALSE for at fortaelle omverdenen om indlaesningen gik|
75   |            godt.                                                 |
76   |                                                                  |
77   | Kaldes af : Hovedprogrammet i _view.                             |
78   |                                                                  |
79   +------------------------------------------------------------------;
80
81   class filematrix(filename, spacesteps, timesteps);
82   text filename;
83   integer spacesteps, timesteps;
84   begin
85     ref(infile) file;
86     boolean exists;
87     long real final_time;
88
89     long real array matrix(0:spacesteps, 0:timesteps);
90
91     file :- new infile(filename);
92     inspect file do
93     begin
94       if open(blanks(100)) and then
95          spacesteps = inint and then
96          timesteps  = inint then
97       begin
98         integer i, j;
99         final_time := inreal;
100
101        for j := 0 step 1 until timesteps do
102          for i := 0 step 1 until spacesteps do
103            matrix(i, j) := inreal;
104
105        exists := true;
106      end **** spacestep og timestep ok ****;
107      close;
108    end *** inspect ***;
109  end ** filematrix **;
110
111
112  !------------------------------------------------------------------+
113  | Navn       : boolean procedure get_dimensions                    |
114  |                                                                  |
115  | Inddata    : text filename, integer spacesteps, timesteps,       |
116  |              long real final_time                                |
```

```
117   |                                                                  |
118   | Omgivelser: Filen ved navn filename skal eksistere fysisk.       |
119   |                                                                  |
120   | Uddata    : I variable spacesteps, timesteps og final_time retur-|
121   |             neres vaerdierne, som er hentet fra filen filename.  |
122   |             Hvis dette gaar godt, saettes den logiske vaerdi af  |
123   |             get_dimensions til TRUE, ellers saettes den til FALSE.|
124   |                                                                  |
125   | Funktion  : Fra filen filename indlaeses dimensionerne spacesteps |
126   |             og timesteps samt sluttiden fianl_time. Disse vaerdier|
127   |             ligger som de tre foerste tal paa filen, hvorfor de er|
128   |             let tilgaengelige.                                   |
129   |                                                                  |
130   |             Derudover saettes der en logisk variabel, som angiver |
131   |             om indlaesningen af de enkelte parametre er lykkedes. |
132   |                                                                  |
133   | Kaldes af : Hovedprogrammet i _view.                             |
134   |                                                                  |
135   +------------------------------------------------------------------;
136
137   boolean procedure get_dimensions(filename, spacesteps,
                                       timesteps, final_time);
138   name spacesteps, timesteps, final_time;
139   text filename;
140   integer spacesteps, timesteps;
141   long real final_time;
142   begin
143     ref(infile) file;
144
145     file :- new infile(filename);
146
147     inspect file do
148     begin
149       if open(blanks(80)) then
150       begin
151         spacesteps := inint;
152         timesteps  := inint;
153         final_time := inreal;
154         close;
155         if spacesteps > 0 and timesteps > 0 then get_dimensions := TRUE;
156       end **** if open ****;
157     end *** inspect ***;
158   end ** get_dimensions **;
159
160   procedure write(mx, title);
161   ref(filematrix) mx;
162   text title;
163   begin
164     integer i,j;
165     outimage;
166     outtext(title);
167     outtext(" er:");
168     outimage;
169     outimage;
170
171     inspect mx do
172     begin
173       for j:= 0 step 1 until timesteps do
174       begin
175         for i := 0 step 1 until spacesteps do
176         begin
```

```
177             outfix(matrix(i,j),4,8);
178           end;
179           outimage;
180         end;
181       end;
182     end;
183
184
185     external text procedure get_filename = "_get_filename";
186
187
188     !-------------------------------------------------------------------+
189     | Navn      : boolean procedure contain                             |
190     |                                                                   |
191     | Inddata   : En text t og et tegn c.                               |
192     |                                                                   |
193     | Uddata    : Den logiske vaerdi TRUE, hvis c findes i t ellers re- |
194     |             turneres FALSE                                        |
195     |                                                                   |
196     | Funktion  : Gennemsoger t indtil c er fundet eller t er slut.     |
197     |             Derefter returneres vaerdien af den lokale logiske    |
198     |             variabel found.                                       |
199     |                                                                   |
200     | Kaldes af : Hovedprogrammet i _view.                              |
201     |                                                                   |
202     +-------------------------------------------------------------------;
203
204     boolean procedure contain(t, c);
205     text t;
206     character c;
207     begin
208       boolean found;
209
210       t.setpos(1);
211       while t.more and not found do
212       begin
213         found := t.getchar = c;
214       end *** while t.more ***;
215       contain := found;
216     end ** contain **;
217
218     ref(filematrix) num, ana, dif;
219
220     text filename, answer;
221     character direction;
222     long real delta_x, delta_t;
223     integer spacesteps, timesteps, final_time;
224     boolean okfilename;
225
226     while not okfilename do
227     begin
228
229       outtext("!12!"); ! Clear screen ;
230       outimage;
231
232       outtext("Programmet view udskriver loesninger ");
233       outtext("fra loesningsfiler");
234       outimage;
235       outimage;
236
237       filename :- get_filename("diffusion");
```

```
238     if get_dimensions(filename & ".num", spacesteps, timesteps,
                          final_time) then
239     begin
240
241       delta_x := 1 / spacesteps;
242       delta_t := final_time / timesteps;
243
244       num :- new filematrix(filename & ".num", spacesteps, timesteps);
245       ana :- new filematrix(filename & ".ana", spacesteps, timesteps);
246       dif :- new filematrix(filename & ".dif", spacesteps, timesteps);
247
248       if ana.exists then
249       begin
250         outimage;
251         outtext("Hvilke loesninger skal udskrives ?");
252         outimage;
253         outimage;
254         outtext("   - tast n for den numeriske loesning");
255         outimage;
256         outtext("   - tast a for den analytiske loesning");
257         outimage;
258         outtext("   - tast d for fejlen mellem loesningerne");
259         outimage;
260         outimage;
261         outtext("> ");
262         breakoutimage;
263
264         inimage;
265         answer :- intext(10).strip;
266
267         if not contain(answer, 'n') then num.exists := FALSE;
268         if not contain(answer, 'a') then ana.exists := FALSE;
269         if not contain(answer, 'd') then dif.exists := FALSE;
270       end  **** if ana.exists *** else
271       dif.exists := FALSE;
272       ! --- For en sikkerhedsskyld ---;
273       okfilename := TRUE;
274     end *** if getdimensions ***;
275   end ** while okfilename **;
276
277   if num.exists then
        write(num, "Den numeriske loesning til problemet " & filename);
278   if ana.exists then
        write(ana, "Den analytiske loesning til problemet " & filename);
279   if dif.exists then
        write(dif, "Fejlen mellem loesningerne til problemet " & filename);
280 end * _view *;
281
282
```

2.8 Diffu_error

Proceduren diffu_error findes i filen `_error.sim`

```
 1
 2  !-------------------------------------------------------------------+
 3  | Navn        : diffu_error                                         |
 4  |                                                                   |
 5  | Inddata     : text title og integer errorlevel.                   |
 6  |                                                                   |
 7  | Uddata      :                                                     |
 8  |                                                                   |
 9  | Funktion    : Udskriver et fejllogo samt fejludskriften           |
10  |               angivet i title. Derefter afbrydes programmet       |
11  |               afhaengigt af errorlevel.                           |
12  |               Heltalsvariablen errorlevel specificerer hvor       |
13  |               alvorlig fejlen er.                                 |
14  |                      |                                            |
15  |               Hvis errorlevel er                                  |
16  |               1,     skal brugeren spoerges om programmet        |
17  |                      oenskes stoppet. Hvis ja saa stop.           |
18  |               2,     er fejlen saa alvorlig, at programmet       |
19  |                      skal stoppes under alle omstaendigheder.     |
20  |               Andet, fortsaetter den programmet efter at          |
21  |       title er udskrevet som en advarsel.          |             |
22  |                                                                   |
23  | Kaldes af   : Solver, _numcalc, _graphmake, _graphdraw            |
24  |                                                                   |
25  +-------------------------------------------------------------------;
26
27  procedure diffu_error(title, errorlevel);
28  text title;
29  integer errorlevel;
30  begin
31
32    !-----------------------------------------------------------------+
33    | Navn      : prestrip                                            |
34    |                                                                 |
35    | Inddata   : En text t                                           |
36    |                                                                 |
37    | Uddata    : En text                                             |
38    |                                                                 |
39    | Funktion  : Fjerner alle eventuelle mellemrum i starten af      |
40    |             teksten og returnerer resultatet.                   |
41    |                                                                 |
42    | Kaldes af : procedure diffu_error.                              |
43    +-----------------------------------------------------------------;
44
45    text procedure prestrip(t); text t;
46    begin
47      t.setpos(1);
48      while t.more and then t.getchar = ' ' do
49        t :- t.sub(2,t.length - 1);
50      prestrip :- t;
51    end ** prestrip **;
52
53    text answer;
54    outtext("**********************************");
55    outimage;
```

```
56    outtext("**            FEJL                **");
57    outimage;
58    outtext("**********************************");
59    outimage;
60    outimage;
61    outtext("**->> ");
62    outtext(prestrip(title));
63    outtext(" <<-**");
64    outimage;
65    outimage;
66    if errorlevel = 2 then
67    begin
68      outtext("Programmet er stoppet.");
69      outimage;
70      outimage;
71      terminate_program;
72    end else
73    if errorlevel = 1 then
74    begin
75      outtext("Vil du stoppe her ? (j/n) [j] ");
76      breakoutimage;
77      inimage;
78      answer :- intext(1).strip;
79      if not lowcase(answer) = "n" then
80      begin
81        outtext("OK - Programmet er stoppet.");
82        outimage;
83        outimage;
84        terminate_program;
85      end *** programmet skal afbrydes ***;
86    end ** errorlevel = 1 **;
87  end * diffu_error *;
```

2.9 Get_filename

Proceduren get_filename findes i filen **_get_filename.sim**

```
1
2
3
4  !-------------------------------------------------------------------+
5  | Navn        : text procedure get_filename                         |
6  |                                                                   |
7  | Indata      : text default_filename                               |
8  |                                                                   |
9  | Uddata      : I getfilename returneres et af brugeren valgt filnavn. |
10 |                                                                   |
11 | Funktion    : Saalaenge filnavnet ikke er ok, skal brugeren vaelge et|
12 |               nyt filnavn. Filnavnet er ok, hvis det enten saettes  |
13 |               til default_filename eller, hvis filnavnets laengde er |
14 |               kortere end MAXFILENAMELENGTH.                      |
15 |               Her kan indsaettes andre betingelser.               |
16 |                                                                   |
17 | kaldes af   : Programmerne _graphmake, _numcalc, _anacalc og _view  |
18 |                                                                   |
19 +-------------------------------------------------------------------;
20
21 text procedure get_filename(default_filename);
22 text default_filename;
23 begin
24   boolean OK;
25   text newfilename;
26   integer MAXFILENAMELENGTH = 30;
27
28   while not OK do
29   begin
30     outimage;
31     outtext("Indtast filnavn (" & default_filename & ") : ");
32     breakoutimage;
33
34     inimage;
35     newfilename :- intext(80).strip;
36
37     if newfilename = NOTEXT then
38     begin
39       get_filename :- copy(default_filename);
40       OK := TRUE;
41     end else
42
43     if newfilename.length > MAXFILENAMELENGTH then
44     begin
45       outtext("Filnavn maa ikke overstige ");
46       outint(MAXFILENAMELENGTH,0);
47       outtext("tegn - om igen.");
48       outimage;
49     end else
50
51     begin
52       get_filename :- copy(newfilename);
53       OK := TRUE;
54     end *** laengden er i orden ***;
55   end ** while not OK **
56 end * get_filename *;
```

2.10 Graphmake

Programmet graphmake findes i filen **_graphmake.sim**

```
 1
 2  !-------------------------------------------------------------------+
 3  | Navn       : _graphmake                                           |
 4  |                                                                   |
 5  | Omgivelser: Filer med navnene *.num *.ana *.dif og *.graph i det ak-|
 6  |             tuelle katalog.                                       |
 7  |                                                                   |
 8  | Uddata     : Data til graphdraw laegges i filen *.graph           |
 9  |                                                                   |
10  | Funktion   : Genererer filer med data til graphdraw.              |
11  |                                                                   |
12  |              Foerst skal brugeren vaelge et filnavn. Programmet ind-|
13  |              laeser derpaa filerne <filnavn>.num, <filnavn>.ana og |
14  |              <filnavn>.dif hvis de findes.                        |
15  |              Hvis der findes mere end *.num kan brugeren vaelge,  |
16  |                                                                   |
17  |              hvilke filer hun oensker at behandle.                |
18  |                                                                   |
19  |              Derefter vaelger brugeren om graferne skal have formen |
20  |              u(x=k,t) (t grafer) eller u(x,t=k) (x grafer).       |
21  |                                                                   |
22  |              Brugeren kan nu angive, for hvilke k, der skal genereres|
23  |              grafer. Disse k vaerdier vedroerer alle de valgte data-|
24  |              filer.                                               |
25  |                                                                   |
26  |              De oenskede grafer ekstraheres og gemmes paa filen   |
27  |              <filnavn>.graph.                                     |
28  |                                                                   |
29  |                                                                   |
30  | Kalder     : Kalder foelgende klasse:                             |
31  |                                                                   |
32  |              - class Filematrix, sammenknytter en fil med et array. |
33  |                                                                   |
34  |              Kalder foelgende procedurer:                         |
35  |                                                                   |
36  |              - boolean procedure get_dimension, Laeser loesningens |
37  |                       dimensioner, til brug for Filematrix objekter.|
38  |              - procedure graph_header, skriver hovedet paa graffilen.|
39  |              - procedure tgraph, fremstiller grafer af typen u(x=k,t)|
40  |              - procedure xgraph, fremstiller grafer af typen u(x,t=k)|
41  |              - text procedure get_filename, faar filnavn fra brugeren|
42  |              - long real procedure getreal, Indleaser reele tal.  |
43  |              - booelan procedure contain, checker om et tegn er i en |
44  |                       tekst.                                      |
45  |              - procedure diffu_error, udskriver eventuelle        |
46  |                       fejlmeddeleser.                             |
47  |                                                                   |
48  | Kaldes af  : Scriptet graphmake.                                  |
49  |                                                                   |
50  +-------------------------------------------------------------------;
51
52  begin
53
54
55    !-----------------------------------------------------------------+
```

```
56    | Navn       : class Filematrix                                    |
57    |                                                                  |
58    | Inddata    : text filename, integer spacesteps, timesteps        |
59    |                                                                  |
60    | Omgivelser: Filen ved navn filename skal eksistere fysisk.       |
61    |                                                                  |
62    | Uddata     : class Filematrix returnerer todimensionale matricer |
63    |              med vaerdier, fra en indlaest fil. Derudover returne-|
64    |              res der en boolean exist, som fortaeller om indlaes-|
65    |              ningen af filerne gik godt.                         |
66    |                                                                  |
67    | Funktion   : Klassen erklaerer et todimensionelt array, med dimen-|
68    |              sionerne spacesteps og timesteps. I arrayet indsaettes|
69    |              vaerdierne fra den fysiske fil, med navnet filename.|
70    |                                                                  |
71    |              Da filen, som gerne skulle vaere en loesning til en |
72    |              givet diffusionsligning, er gemt paa foelgende format|
73    |                                                                  |
74    |              - integer spacesteps                                |
75    |              - integer timesteps                                 |
76    |              - long real final_time                              |
77    |              - vaerdierne i loesningsmatricen, gemt under hinanden|
78    |                raekkevis.                                        |
79    |                                                                  |
80    |              Det undersoeges dernaest om parametrerne spacesteps og|
81    |              timesteps stemmer ovrens, med de vaerdier, som er gemt|
82    |              paa den fysiske fil. Dernaest hentes konstanten     |
83    |              final_time inden den erklaerede "matrix" fyldes, med|
84    |              vaerdierne fra loesningen til en diffusionsligning. |
85    |                                                                  |
86    |              Til sidst saettes den logiske variable exist = TRUE / |
87    |              FALSE for at fortaelle omverdenen om indlaesningen gik|
88    |              godt.                                               |
89    |                                                                  |
90    | Kaldes af  : Hovedprogrammet i _graphmake.                       |
91    |                                                                  |
92    +------------------------------------------------------------------;
93
94    class filematrix(filename, spacesteps, timesteps);
95    text filename;
96    integer spacesteps, timesteps;
97    begin
98      ref(infile) file;
99      boolean exists;
100     long real final_time;
101
102     long real array matrix(0:spacesteps, 0:timesteps);
103
104     file :- new infile(filename);
105     inspect file do
106     begin
107       if open(blanks(100)) and then
108          spacesteps = inint and then
109          timesteps  = inint then
110       begin
111         integer i, j;
112         final_time := inreal;
113
114
115         for j := 0 step 1 until timesteps do
116           for i := 0 step 1 until spacesteps do
```

```
117             matrix(i, j) := inreal;
118
119           exists := true;
120         end **** spacestep og timestep ok ****;
121         close;
122       end *** inspect ***;
123     end ** filematrix **;
124
125     external procedure diffu_error = "_error";
126
127
128     !-------------------------------------------------------------------+
129     | Navn       : boolean procedure get_dimensions                     |
130     |                                                                   |
131     | Inddata    : text filename, integer spacesteps, timesteps,        |
132     |              long real final_time                                 |
133     |                                                                   |
134     | Omgivelser: Filen ved navn filename skal eksistere fysisk.        |
135     |                                                                   |
136     | Uddata     : I variable spacesteps, timesteps og final_time retur- |
137     |              neres vaerdierne, som er hentet fra filen filename.  |
138     |              Hvis dette gaar godt, saettes den logiske vaerdi af  |
139     |              get_dimensions til TRUE, ellers saettes den til FALSE.|
140     |                                                                   |
141     | Funktion   : Fra filen filename indlaeses dimensionerne spacesteps |
142     |              og timesteps samt sluttiden fianl_time. Disse vaerdier|
143     |              ligger som de tre foerste tal paa filen, hvorfor de er|
144     |              let tilgaengelige.                                   |
145     |                                                                   |
146     |              Derudover saettes der en logisk variabel, som angiver |
147     |              om indlaesningen af de enkelte parametre er lykkedes. |
148     |                                                                   |
149     | Kaldes af  : Hovedprogrammet i _graphmake.                        |
150     |                                                                   |
151     +-------------------------------------------------------------------;
152
153     boolean procedure get_dimensions(filename, spacesteps,
                                         timesteps, final_time);
154     name spacesteps, timesteps, final_time;
155     text filename;
156     integer spacesteps, timesteps;
157     long real final_time;
158     begin
159       ref(infile) file;
160
161       file :- new infile(filename);
162
163       inspect file do
164       begin
165         if open(blanks(80)) then
166         begin
167           spacesteps := inint;
168           timesteps  := inint;
169           final_time := inreal;
170           close;
171           if spacesteps > 0 and timesteps > 0 then get_dimensions := TRUE;
172         end **** if open ****;
173       end *** inspect ***;
174     end ** get_dimensions **;
175
176
```

```
177   !-------------------------------------------------------------------+
178   | Navn      : procedure graph_header                                |
179   |                                                                   |
180   | Inddata   : text titletext, ref(outfile) file.                    |
181   |                                                                   |
182   | Uddata    : Paa filen file, skrives det hovede som kraeves for, at|
183   |             graphdraw kan tegne og printe de rigtige grafer.      |
184   |                                                                   |
185   | Funktion  : Skriver Informationer til xgraph, i file. Det er in-  |
186   |             formation om postscript og de paagaeldende grafers    |
187   |             tittel.                                               |
188   |                                                                   |
189   | Kaldes af : Hovedprogrammet i _graphmake.                         |
190   |                                                                   |
191   +-------------------------------------------------------------------;
192
193   procedure graph_header(titletext, file); text titletext;
194   ref(outfile) file;
195   inspect file do
196   begin
197     outtext("Device: Postscript");
198     outimage;
199     outtext("TitleText: " & titletext);
200     outimage;
201   end ** graph_header **;
202
203
204   !-------------------------------------------------------------------+
205   | Navn      : procedure tgraphs                                     |
206   |                                                                   |
207   | Omgivelser: ref(outfile) graphfile og delta_x skal kendes. De er  |
208   |             erklaeret globalt i _graphmake.                       |
209   |                                                                   |
210   | Uddata    : I graphfile er dannet en fil, saaledes at de oenskede |
211   |             grafer kan tegnes ved brug af graphdraw.              |
212   |                                                                   |
213   | Funktion  : Foerst skal brugeren vaelge for hvilke tidsvaerdier   |
214   |             der skal tegnes grafer. Disse tal indlaeses i en tekst|
215   |             t, hvorefter de taelles ved at undersoege hvor mange  |
216   |             mellemrum der er - ved t mellemrum, vil der vaere t+1 |
217   |             tal. Resultatet gemmes i variablen counter.           |
218   |                                                                   |
219   |             Naar dette er gjort oprettes der et array vector med  |
220   |             counter pladser. I denne vektor indlaeses de naermeste|
221   |             tidsskridt svarende til de oenskede tider.            |
222   |                                                                   |
223   |             Naar dette er klaret genereres hovedet paaa filen ved |
224   |             at kalde proceduren header.                           |
225   |                                                                   |
226   |             Efter brugerens oenske genereres der nu grafer for de |
227   |             i _graphmake's hovedprogram valgte objekter. Disse    |
228   |             valg checkes ved at undersoege vaerdien af de mulige  |
229   |             objekters exist. Graferne for de enkelte objekter     |
230   |             genereres ved at kalde proceduren generate_tgraphs.   |
231   |                                                                   |
232   |                                                                   |
233   | Kalder    : Procedurerne graph_header og generate_tgraphs.        |
234   |                                                                   |
235   | Kaldes af : Hovedprogrammet i _graphmake.                         |
236   |                                                                   |
237   +-------------------------------------------------------------------;
```

```
238
239   procedure tgraphs;
240   begin
241
242
243     !-------------------------------------------------------------+
244     | Navn      : procedure generate_tgraphs                      |
245     |                                                             |
246     | Inddata   : ref(filematrix) m, integer array counter,       |
247     |             integer counter, ref(outfile) graphfile.        |
248     |                                                             |
249     | Omgivelser: Fra hovedprogrammet i graphmake kendes delta_x og |
250     |             delta_t.                                        |
251     |                                                             |
252     | Uddata    : I graphfile skrives for et objekt m, vaerdier til |
253     |             tgraph til de valgte steder x.                  |
254     |                                                             |
255     | Funktion  : Proceduren, som er lokal i procedure tgraphs genere-|
256     |             rer grafer til de valgte stedskridt, til et objekt |
257     |             m.                                              |
258     |                                                             |
259     |             Dette goeres ved, for alle de valgte stedskridt at |
260     |             skrive den rigtige tid ned paa graphfile, hvorefter |
261     |             det rigtige tid og dens funktionsvaerdi skrives,  |
262     |             alle tidskridt paa graphfile under hinanden som   |
263     |             foelger :                                       |
264     |                                                             |
265     |             1. tid              1. funktionsvaerdi          |
266     |             2. tid              2. funktionsvaerdi          |
267     |                  ...                  ...                   |
268     |             sidste sted      sidste funktionsvaerdi         |
269     |                                                             |
270     |                                                             |
271     | Kaldes af : Hovedprogrammet i _graphmake.                   |
272     |                                                             |
273     +-------------------------------------------------------------+;
274
275     procedure generate_tgraph(m, vector, counter, graphfile);
276     ref(filematrix) m;
277     integer array vector;
278     integer counter;
279     ref(outfile) graphfile;
280
281     inspect graphfile do
282     begin
283       integer i, j;
284
285       lowten('E');
286       for j := 1 step 1 until counter do
287       begin
288         outimage;
289         outtext("""x=");
290         outreal(vector(j)*delta_x,2,0);
291         outimage;
292
293         for i := 0 step 1 until timesteps do
294         begin
295           outreal(i*delta_t, 16, 0);
296           outchar(' ');
297           outreal(m.matrix(vector(j), i), 16, 0);
298           outimage;
```

```
299         end ***** for i *****;
300       end **** for j ****;
301     end *** inspect ***;
302
303     text t;
304     character c;
305     integer counter;
306
307     outimage;
308     outtext("Vaelg de x-vaerdier, hvortil der skal genereres grafer.");
309     outimage;
310     outimage;
311     outtext("   - x tilhoerer intervallet [0;1]");
312     outimage;
313     outimage;
314     outtext("> ");
315     breakoutimage;
316
317     inimage;
318     t :- intext(80).strip;
319
320     counter := 1;
321     while t.more do
322     begin
323       c := t.getchar;
324       if c = ' ' then counter := counter + 1;
325       while t.more and then c = ' ' do c := t.getchar;
326     end *** while t.more ***;
327
328     begin
329       long real y;
330       integer i;
331       integer array vector(1:counter);
332
333       t.setpos(1);
334       for i := 1 step 1 until counter do
335       begin
336         y := getreal(t);
337         vector(i) := y  / delta_x;
338       end **** for i ****;
339
340       graphfile :- new outfile(filename & ".graph");
341
342       inspect graphfile do
343       begin
344         setaccess("ANYCREATE");
345         if not open(blanks(80)) then
346            diffu_error(filename &".graph kan ikke aabnes. ", 2)
347         else
348         begin
349           graph_header("u(x=k, t)", graphfile);
350
351           if num.exists then generate_tgraph(num,vector,counter,graphfile);
352           if ana.exists then generate_tgraph(ana,vector,counter,graphfile);
353           if dif.exists then generate_tgraph(dif,vector,counter,graphfile);
354           close;
355         end ***** if open *****;
356       end **** inspect ****;
357     end *** counter defineret ***;
358   end ** tgraphs **;
359
```

```
360
361   !-------------------------------------------------------------------+
362   | Navn       : procedure xgraphs                                    |
363   |                                                                   |
364   | Omgivelser: ref(outfile) graphfile og delta_t skal kendes. De er  |
365   |             erklaeret globalt i _graphmake.                       |
366   |                                                                   |
367   | Uddata     : I graphfile er dannet en fil, saaledes at de oenskede|
368   |             grafer kan tegnes ved brug af graphdraw.              |
369   |                                                                   |
370   | Funktion   : Foerst skal brugeren vaelge for hvilke tidsvaerdier  |
371   |              der skal tegnes grafer. Disse tal indlaeses i en tekst|
372   |              t, hvorefter de taelles ved at undersoege hvor mange |
373   |              mellemrum der er - ved x mellemrum, vil der vaere x+1|
374   |              tal. Resultatet gemmes i variablen counter.          |
375   |                                                                   |
376   |              Naar dette er gjort oprettes der et array vector med |
377   |              counter pladser. I denne vektor indlaeses de naermeste|
378   |              tidsskridt svarende til de oenskede tider.           |
379   |                                                                   |
380   |              Naar dette er klaret genereres hovedet paaa filen ved|
381   |              at kalde proceduren header.                          |
382   |                                                                   |
383   |              Efter brugerens oenske genereres der nu grafer for de|
384   |              i _graphmake's hovedprogram valgte objekter. Disse   |
385   |              valg checkes ved at undersoege vaerdien af de mulige |
386   |              objekters exist. Graferne for de enkelte objekter    |
387   |              genereres ved at kalde proceduren generate_xgraphs.  |
388   |                                                                   |
389   |                                                                   |
390   | Kalder     : Procedurerne graph_header og generate_xgraphs.       |
391   |                                                                   |
392   | Kaldes af  : Hovedprogrammet i _graphmake.                        |
393   |                                                                   |
394   +-------------------------------------------------------------------;
395
396   procedure xgraphs;
397   begin
398
399     !-----------------------------------------------------------------+
400     | Navn       : procedure generate_xgraphs                         |
401     |                                                                 |
402     | Inddata    : ref(filematrix) m, integer array counter,          |
403     |              integer counter, ref(outfile) graphfile.           |
404     |                                                                 |
405     | Omgivelser: Fra hovedprogrammet i graphmake kendes delta_x og   |
406     |              delta_t.                                           |
407     |                                                                 |
408     | Uddata     : I graphfile skrives for et objekt m, vaerdier til  |
409     |              xgraph til de valgte tider t.                      |
410     |                                                                 |
411     | Funktion   : Proceduren, som er lokal i procedure xgraphs genere-|
412     |              rer grafer til de valgte tidsskridt, til et objekt |
413     |              m.                                                 |
414     |                                                                 |
415     |              Dette goeres ved, for alle de valgte tidsskridt at |
416     |              skrive den rigtige tid ned paa graphfile, hvorefter|
417     |              det rigtige sted og dets funktionsvaerdi skrives,  |
418     |              alle stedskridt paa graphfile under hinanden som   |
419     |              foelger :                                          |
420     |                                                                 |
```

```
421     |            1. sted              1. funktionsvaerdi          |
422     |            2. sted              2. funktionsvaerdi          |
423     |                ...                      ...                |
424     |            sidste sted      sidste funktionsvaerdi          |
425     |                                                             |
426     |                                                             |
427     | Kaldes af : Hovedprogrammet i _graphmake.                   |
428     |                                                             |
429     +-------------------------------------------------------------;
430
431     procedure generate_xgraph(m, vector, counter, graphfile);
432     ref(filematrix) m;
433     integer array vector;
434     integer counter;
435     ref(outfile) graphfile;
436
437     inspect graphfile do
438     begin
439       integer i, j;
440
441       lowten('E');
442
443       for j := 1 step 1 until counter do
444       begin
445         outimage;
446         outtext("""t=");
447         outreal(vector(j)*delta_t, 2, 0);
448         outimage;
449
450         for i := 0 step 1 until spacesteps do
451         begin
452           outreal(i*delta_x, 16, 0);
453           outchar(' ');
454           outreal(m.matrix(i, vector(j)), 16, 0);
455           outimage;
456         end ***** for i ****;
457       end **** for j ****;
458     end *** generate_xgraph ***;
459
460     text t;
461     character c;
462     integer counter;
463
464     outimage;
465     outtext("Vaelg de t-vaerdier, hvortil der skal genereres grafer.");
466     outimage;
467     outimage;
468     outtext("   - t tilhoerer intervallet [0;");
469     outreal(final_time,4,0);
470     outtext("] ");
471     outimage;
472     outimage;
473     outtext("> ");
474     breakoutimage;
475
476     inimage;
477     t :- intext(80).strip;
478
479     counter := 1;
480     while t.more do
481     begin
```

```
482       c := t.getchar;
483       if c = ' ' then counter := counter + 1;
484       while t.more and then c = ' ' do c := t.getchar;
485     end *** while t..more ***;
486
487     begin
488       long real y;
489       integer i;
490       integer array vector(1:counter);
491
492       t.setpos(1);
493       for i := 1 step 1 until counter do
494       begin
495         y := getreal(t);
496         if y > final_time or y < 0 then
497         begin
498           text errormsg;
499           errormsg :- copy(blanks(20));
500           errormsg.setpos(1);
501           errormsg.putfix(y,2);
502           diffu_error(errormsg.strip & " er uden for intervallet",2);
503         end;
504         vector(i) := y  / delta_t;
505       end **** for i ****;
506
507       graphfile :- new outfile(filename & ".graph");
508       inspect graphfile do
509       begin
510         setaccess("ANYCREATE");
511         if not open(blanks(80)) then
512            diffu_error(filename &".graph kan ikke aabnes. ", 2) else
513         begin
514           graph_header("u(x, t=k)", graphfile);
515
516           if num.exists then
517             generate_xgraph(num, vector, counter, graphfile);
518           if ana.exists then
519             generate_xgraph(ana, vector, counter, graphfile);
520           if dif.exists then
521            generate_xgraph(dif, vector, counter, graphfile);
522           close;
523         end ***** hvis graphfile kan aabnes *****;
524       end **** inspect ****;
525     end *** counter er defineret ***;
526   end ** xgraph **;
527
528
529   external text procedure get_filename = "_get_filename";
530
531   !-------------------------------------------------------------------+
532   | Navn      : long real procedure getreal                           |
533   |                                                                   |
534   | Inddata   : text t                                                |
535   |                                                                   |
536   | Funktion  : Vi har medtaget denne procedure, da simulas indbyggede|
537   |             funktion getreal hele tiden returnerer det foerste re-|
538   |             elle tal, hver gang den kaldes. Vi skal kunne laese   |
539   |             dem ind i raekkefoelge.                               |
540   |             Problemet lappes, ved at afskaere tallene efterhaanden|
541   |             som getreal tager dem.                                |
542   |                                                                   |
```

```
543   | Kaldes af : xgraph og tgraph.                                          |
544   |                                                                        |
545   +------------------------------------------------------------------------;
546
547   long real procedure getreal(t);
548   name t;
549   text t;
550   begin
551     long real x;
552     x := t.getreal;
553     t :- t.sub(t.pos, t.length - t.pos + 1);
554     getreal := x;
555   end ** getreal **;
556
557   !------------------------------------------------------------------------+
558   | Navn      : boolean procedure contain                                  |
559   |                                                                        |
560   | Inddata   : En text t og et tegn c.                                    |
561   |                                                                        |
562   | Uddata    : Den logiske vaerdi TRUE, hvis c findes i t ellers re-      |
563   |             turneres FALSE                                             |
564   |                                                                        |
565   | Funktion  : Gennemsoger t indtil c er fundet eller t er slut.          |
566   |             Derefter returneres vaerdien af den lokale logiske         |
567   |             variabel found.                                            |
568   |                                                                        |
569   | Kaldes af : Hovedprogrammet i _graphmake.                              |
570   |                                                                        |
571   +------------------------------------------------------------------------;
572
573   boolean procedure contain(t, c);
574   text t;
575   character c;
576   begin
577     boolean found;
578
579     t.setpos(1);
580     while t.more and not found do
581     begin
582       found := t.getchar = c;
583     end *** while t.more ***;
584     contain := found;
585   end ** contain **;
586
587   ref(outfile) graphfile;
588   ref(filematrix) num, ana, dif;
589
590   text filename, answer;
591   character direction;
592   long real delta_x, delta_t, final_time;
593   integer spacesteps, timesteps;
594   boolean okfilename;
595
596   while not okfilename do
597   begin
598     outtext("!12!");
599     outimage;
600     outtext("Programmet graf_make genererer udvalgte grafer ");
601     outtext("fra loesningsfiler");
602     outimage;
603     outimage;
```

```
604
605     filename :- get_filename("diffusion");
606     if get_dimensions(filename & ".num", spacesteps,
                          timesteps, final_time) then
607     begin
608
609       delta_x := 1 / spacesteps;
610       delta_t := final_time / timesteps;
611
612       num :- new filematrix(filename & ".num", spacesteps, timesteps);
613       ana :- new filematrix(filename & ".ana", spacesteps, timesteps);
614       dif :- new filematrix(filename & ".dif", spacesteps, timesteps);
615
616       if ana.exists then
617       begin
618         outimage;
619         outtext("Hvilke grafer skal genereres ?");
620         outimage;
621         outimage;
622         outtext("   - tast n for den numeriske loesning");
623         outimage;
624         outtext("   - tast a for den analytiske loesning");
625         outimage;
626         outtext("   - tast d for fejlen mellem loesningerne");
627         outimage;
628         outimage;
629         outtext("> ");
630         breakoutimage;
631
632         inimage;
633         answer :- intext(10).strip;
634
635         if not contain(answer, 'n') then num.exists := FALSE;
636
637         if not contain(answer, 'a') then ana.exists := FALSE;
638         if not contain(answer, 'd') then dif.exists := FALSE;
639       end  **** if ana.exists *** else
640       dif.exists := FALSE;
641       ! --- For en sikkerhedsskyld ---;
642       okfilename := TRUE;
643     end *** if getdimensions ***;
644   end ** while okfilename **;
645
646   while direction <> 'x' and  direction <> 't' do
647   begin
648     outimage;
649     outtext("Skal de genererede grafer vaere u(x=k, t)
                                          eller u(x, t=k) ? ");
650     outimage;
651     outimage;
652     outtext("   - tast x for u(x, t=k)");
653     outimage;
654     outtext("   - tast t for u(x=k, t)");
655     outimage;
656     outimage;
657     outtext("> ");
658     breakoutimage;
659     inimage;
660     direction := inchar;
661
662     if direction = 'x' then xgraphs else
```

```
663       if direction = 't' then tgraphs;
664
665       outimage;
666     end ** while direction .. **;
667   end * _graphmake *;
```

Kapitel 3

Styrefiler

3.1 Numcalc

```
NUMCALC
:
# Udfoerer programmet _numcalc ved hjaelp af makefilen _runnumcalc

make -f _runnumcalc
```

3.2 EIS

```
EIS
:
# Koerer numcalc og fysik for en raekke forskellige frekvenser.
# Inddata : Filnavnet paa den fil, hvor de oenskede frekvenser
# ligger, samt filnavnet under hvilket man oensker resultaterne
# gemt.

if test "$1" = ""
then
  echo -n "Indtast frekvensfilnavn (frekvens) "
  read frekvensfilnavn
  if test "$frekvensfilnavn" = ""
  then
    $0 "frekvens"
  else
    $0 $frekvensfilnavn
  fi
else
  if test "$2" = ""
  then
    echo -n "Indtast resultatfilnavn (diffusion) "
    read resultatfilnavn
    if test "$resultatfilnavn" = ""
    then
      $0 $1 "diffusion"
```

```
    else
      $0 $1 $resultatfilnavn
    fi
  else
    if test -f ${2}.num.data
    then
      echo  "Filen Eksisterer, hvad skal ske ?"
      echo  "S - filen slettes"
      echo  "T - tilfoej nye data"
      echo  "N - indtast nyt filnavn"
      echo -n "?- "
      read svar
      case $svar in
        "s" | "S") rm -f ${2}.num.data ;;
        "t" | "T") ;;
        "n" | "N") $0 $1 ;
                   exit 1 ;;
        *)         echo "Program afsluttet" ;
                   echo
                   exit 1 ;;
      esac
    fi
    echo "Regne regne... "
    for omega in `cat $1`
    do
    echo $omega > .fil1
    echo "$2" >> .fil1
    numcalc < .fil1 > /dev/null
    fysik -m=16 < .fil1 > /dev/null
    done
    echo
  fi
fi
```

3.3 Impadm

```
IMPADM
:
# Udfoerer programmet _imp_adm_graphmake ved hjaelp af makefilen _runimpadm

make -f _runimpadm
```

3.4 Fysik

```
FYSIK
:
# Udfoerer programmet _fysik ved hjaelp af makefilen _runfysik

make -f _runfysik
```

3.5 Graphmake

```
GRAPHMAKE
:
# Udfoerer programmet _graphmake ved hjaelp af makefilen _rungraphmake

make -f _rungraphmake
```

3.6 Graphdraw

```
GRAPHDRAW
:
# Graphdraw soerger for at skaffe et passende filnavn og kalde xgraph.
# Hvis man ikke angiver noget suffix, saettes .graph paa og
# graphdraw kaldes med <filnavn>.graph.
# Hvis suffixet .graph angives kaldes xgraph med filnavnet.
# Saaledes kaldes xgraph altid med <filnavn>.graph.

#set -x
clear
if test "$1" = ""
then                    # Intet filnavn paa kommandolinie
  echo -n "Indtast filnavn (diffusion) : "
  read filename
  if test "$filename" = ""
  then
    $0 "diffusion"      # Kalder graphdraw med defaultfilnavn
  else
    $0 $filename        # kalder graphdraw med indtastet filnavn
  fi
else
  if test $1 = `basename $1 .graph`.graph
  then                  # Filnavn har suffixet .graph
    $0 `basename $1 .graph`        # Graphdraw kaldes uden ekstra suffix
  else
    /usr/local/xgraph < $1.graph   # Suffixet .graph saettes paa
  fi
fi
```

3.7 View

```
VIEW
:
# Udfoerer programmet _view ved hjaelp af makefilen _runview

make -f _runview
```

3.8 List

```
LIST
:
# Giver en oversigt over systemets datafiler

echo
echo
ls *.num 2> /dev/null
ls *.ana 2> /dev/null
ls *.dif 2> /dev/null
ls *.graph 2> /dev/null
ls *.ps 2> /dev/null
echo
echo
```

3.9 Print

```
PRINT
:
# Sender graffil til postscript printer.
lpr $1
```

3.10 Clearproblem

```
CLEARPROBLEM
:
# Dette script kopierer en frisk version af PROBLEM.SIM fra .safebox.

cd
cp .safebox/problem.sim  .
```

3.11 Reset

```
RESET
:
# Kopierer hele systemet fra .safebox.

cd
cp .safebox/*.sim  .
cp .safebox/* .
```

3.12 Clean

```
CLEAN
:
$ Denne styrefil sletter alle de overfloedig filer, som SIMULA
$ oversaetteren genererer.
$ Derefter slettes de brugergenererede filer, dog saaledes, at brugeren
$ promptes for hver fil

$ Disse filer fjernes:
rm -f *.atr *.o *.lis *.err *%

echo Foelgende Filer Fjernes - tast 'y' ud for de filer, som skal slettes !

$ Disse filer fjernes, hvis der svares 'y'.
rm -i *.num *.dif *.ana *.graph *.ps
```

3.13 Help

```
HELP
:

$Udskriver en oversigt over systemets styrefiler

clear

cat << EOD
 Foelgende komandoer benyttes til at koere diffusionsligninsprogrammet.

 numcalc              - Loeser det problem, som er sat i problem.sim
                        numerisk.
 graphmake            - Genererer grafer, fra loesningsfiler.
 graphdraw            - Tegner genererede grafer.
 view                 - Viser loesninger paa skaermen i form af tal.


 Foelgende komandoer benyttes til at arbejde med systemet.

 list                 - Viser de filer, som er genereret samt
                        problem.sim.
 print                - Printer en graf ud.
 clearproblem         - Genetablerer filen problem.sim, hvis den
                        er blevet slettet.
 reset                - Genetablerer systemet, hvis det er blevet
                        slettet.
 erase                - Sletter en fil.
 clear                - Sletter filerne *.num, *.ana, *.dif, *.graph
                        og *.ps.
 help                 - Viser dette dokument.


 Foelgende kommandoer benyttes til at behandle impedans og admittans.

 eis                  - Løser diffusionslignigen for de forskellige
                        frekvenser og gemmer henholdsvis impedansen
                        og admitansen på fil.
```

```
impadm                  - Generer grafer for henholdsvis impedansen og
                          admittansen. Disse kan tegnes med graphdraw.
EOD

echo
```

3.14 _runnumcalc (makefile)

Vi har valgt kun at liste en makefil, da de alle har samme struktur.

```
_RUNNUMCALC
# makefile      : Til at koere programmet _numcalc

.PRECIOUS: _runnumcalc
.SILENT:

  _runnumcalc    : _numcalc
        _numcalc -m=16

  _numcalc       : problem.o _get_filename.o _solver.o _numcalc.o _error.o
        echo
        echo "Linker program _numcalc"
        echo
        simld _numcalc problem _solver _get_filename _error

  _numcalc.o _numcalc.atr   : _solver.atr _get_filename.atr _error.atr _numcalc.sim
        echo
        echo "Oversaetter program _numcalc"
        echo
        simcomp _numcalc

  _solver.o _solver.atr     : problem.atr _error.atr _solver.sim
        echo
        echo "Oversaetter modul _solver"
        echo
        simcomp _solver`

  problem.o problem.atr       : problem.sim
        echo
        echo "Oversaetter modul problem"
        echo
        simcomp problem

  _get_filename.o _get_filename.atr : _get_filename.sim
        echo
        echo "Oversaetter modul _get_filename"
        echo
        simcomp _get_filename

  _error.o _error.atr    : _error.sim
        echo
        echo "Oversaetter modul _error"
        echo
        simcomp _error
```

1/78 "TANKER OM EN PRAKSIS" - et matematikprojekt.
Projektrapport af: Anne Jensen, Lena Lindenskov, Marianne Kesselhahn og Nicolai Lomholt.
Vejleder: Anders Madsen

2/78 "OPTIMERING" - Menneskets forøgede beherskelsesmuligheder af natur og samfund.
Projektrapport af: Tom J. Andersen, Tommy R. Andersen, Gert Krenøe og Peter H. Lassen
Vejleder: Bernhelm Boss.

3/78 "OPCAVESAMLING", breddekursus i fysik.
Af: Lasse Rasmussen, Aage Bonde Kræmmer og Jens Højgaard Jensen.

4/78 "TRE ESSAYS" - om matematikundervisning, matematiklæreruddannelsen og videnskabsfilosofien.
Af: Mogens Niss
Nr. 4 er p.t. udgået.

5/78 "BIBLIOGRAFISK VEJLEDNING til studiet af DEN MODERNE FYSIKS HISTORIE".
Af: Helge Kragh.
Nr. 5 er p.t. udgået.

6/78 "NOGLE ARTIKLER OG DEBATINDLÆG OM - læreruddannelse og undervisning i fysik, og - de naturvidenskabelige fags situation efter studenteroprøret".
Af: Karin Beyer, Jens Højgaard Jensen og Bent C. Jørgensen.

7/78 "MATEMATIKKENS FORHOLD TIL SAMFUNDSØKONOMIEN".
Af: B.V. Gnedenko.
Nr. 7 er udgået.

8/78 "DYNAMIK OG DIAGRAMMER". Introduktion til energy-bond-graph formalismen.
Af: Peder Voetmann Christiansen.

9/78 "OM PRAKSIS' INDFLYDELSE PÅ MATEMATIKKENS UDVIKLING". - Motiver til Kepler's: "Nova Stereometria Doliorum Vinariom".
Projektrapport af: Lasse Rasmussen.
Vejleder: Anders Madsen.

10/79 "TERMODYNAMIK I GYMNASIET".
Projektrapport af: Jan Christensen og Jeanne Mortensen.
Vejledere: Karin Beyer og Peder Voetmann Christiansen.

11/79 "STATISTISKE MATERIALER".
Af: Jørgen Larsen.

12/79 "LINEÆRE DIFFERENTIALLIGNINGER OG DIFFERENTIALLIGNINGSSYSTEMER".
Af: Mogens Brun Heefelt.
Nr. 12 er udgået.

13/79 "CAVENDISH'S FORSØG I GYMNASIET".
Projektrapport af: Gert Kreinøe.
Vejleder: Albert Chr. Paulsen.

14/79 "BOOKS ABOUT MATHEMATICS: History, Philosophy, Education, Models, System Theory, and Works of".
Af: Else Høyrup.
Nr. 14 er p.t. udgået.

15/79 "STRUKTUREL STABILITET OG KATASTROFER i systemer i og udenfor termodynamisk ligevægt".
Specialeopgave af: Leif S. Striegler.
Vejleder: Peder Voetmann Christiansen.

16/79 "STATISTIK I KRÆFTFORSKNINGEN".
Projektrapport af: Michael Olsen og Jørn Jensen.
Vejleder: Jørgen Larsen.

17/79 "AT SPØRGE OG AT SVARE i fysikundervisningen".
Af: Albert Christian Paulsen.

18/79 "MATHEMATICS AND THE REAL WORLD", Proceedings af an International Workshop, Roskilde University Centre, Denmark, 1978.
Preprint.
Af: Bernhelm Booss og Mogens Niss (eds.)

19/79 "GEOMETRI, SKOLE OG VIRKELIGHED".
Projektrapport af: Tom J. Andersen, Tommy R. Andersen og Per H.H. Larsen.
Vejleder: Mogens Niss.

20/79 "STATISTISKE MODELLER TIL BESTEMMELSE AF SIKRE DOSER FOR CARCINOGENE STOFFER".
Projektrapport af: Michael Olsen og Jørn Jensen.
Vejleder: Jørgen Larsen

21/79 "KONTROL I GYMNASIET-FORMÅL OG KONSEKVENSER".
Projektrapport af: Crilles Bacher, Per S.Jensen, Preben Jensen og Torben Nysteen.

22/79 "SEMIOTIK OG SYSTEMEGENSKABER (1)".
1-port lineært response og støj i fysikken.
Af: Peder Voetmann Christiansen.

23/79 "ON THE HISTORY AF EARLY WAVE MECHANICS - with special emphasis on the role af realitivity".
Af: Helge Kragh.

24/80 a+b "MATEMATIKOPFATTELSER HOS 2.C'ERE".
1. En analyse. 2. Interviewmateriale.
Projektrapport af: Jan Christensen og Knud Lindhardt Rasmussen.
Vejleder: Mogens Niss.

25/80 "EKSAMENSOPGAVER", Dybdemodulet/fysik 1974-79.

26/80 "OM MATEMATISKE MODELLER".
En projektrapport og to artikler.
Af: Jens Højgaard Jensen m.fl.

27/80 "METHODOLOGY AND PHILOSOPHY AF SCIENCE IN PAUL DIRAC's PHYSICS".
Af: Helge Kragh.

28/80 "DIELEKTRISK RELAXATION - et forslag til en ny model bygget på væskernes viscoelastiske egenskaber".
Projektrapport af: Gert Kreinøe.
Vejleder: Niels Boye Olsen.

29/80 "ODIN - undervisningsmateriale til et kursus i differentialligningsmodeller".
Projektrapport af: Tommy R. Andersen, Per H.H. Larsen og Peter H. Lassen.
Vejleder: Mogens Brun Heefelt.

30/80 "FUSIONSENERGIEN - - - ATOMSAMFUNDETS ENDESTATION".
Af: Oluf Danielsen.
Nr. 30 er udgået.

31/80 "VIDENSKABSTEORETISKE PROBLEMER VED UNDERVISNINGSSYSTEMER BASERET PÅ MENGDELÆRE".
Projektrapport af: Troels Lange og Jørgen Karrebæk.
Vejleder: Stig Andur Pedersen.
Nr. 31 er p.t. udgået.

32/80 "POLYMERE STOFFERS VISCOELASTISKE EGENSKABER - BELYST VED HJÆLP AF MEKANISKE IMPEDANSMÅLINGER OG MØSSBAUEREFFEKTMÅLINGER".
Projektrapport af: Crilles Bacher og Preben Jensen.
Vejledere: Niels Boye Olsen og Peder Voetmann Christiansen.

33/80 "KONSTITUERING AF FAG INDEN FOR TEKNISK - NATURVIDENSKABELIGE UDDANNELSER. I-II".
Af: Arne Jakobsen.

34/80 "ENVIRONMENTAL IMPACT AF WIND ENERGY UTILIZATION".
ENERGY SERIES NO. I.
Af: Bent Sørensen
Nr. 34 er udgået.

35/80 "HISTORISKE STUDIER I DEN NYERE ATOMFYSIKS UDVIKLING".
Af: Helge Kragh.

36/80 "HVAD ER MENINGEN MED MATEMATIKUNDERVISNINGEN?".
Fire artikler.
Af: Mogens Niss.

37/80 "RENEWABLE ENERCY AND ENERGY STORAGE".
ENERGY SERIES NO. 2.
Af: Bent Sørensen.

38/81 "TIL EN HISTORIETEORI OM NATURERKENDELSE, TEKNOLOGI OG SAMFUND".
Projektrapport af: Erik Gade, Hans Hedal, Henrik Lau og Finn Physant.
Vejledere: Stig Andur Pedersen, Helge Kragh og Ib Thiersen.
Nr. 38 er p.t. udgået.

39/81 "TIL KRITIKKEN AF VÆKSTØKONOMIEN".
Af: Jens Højgaard Jensen.

40/81 "TELEKOMMUNIKATION I DANMARK - oplæg til en teknologivurdering".
Projektrapport af: Arne Jørgensen, Bruno Petersen og Jan Vedde.
Vejleder: Per Nørgaard.

41/81 "PLANNING AND POLICY CONSIDERATIONS RELATED TO THE INTRODUCTION OF RENEWABLE ENERGY SOURCES INTO ENERGY SUPPLY SYSTEMS".
ENERGY SERIES NO. 3.
Af: Bent Sørensen.

42/81 "VIDENSKAB TEORI SAMFUND - En introduktion til materialistiske videnskabsopfattelser".
Af: Helge Kragh og Stig Andur Pedersen.

43/81 1."COMPARATIVE RISK ASSESSMENT OF TOTAL ENERGY SYSTEMS".
2."ADVANTAGES AND DISADVANTAGES OF DECENTRALIZATION".
ENERGY SERIES NO. 4.
Af: Bent Sørensen.

44/81 "HISTORISKE UNDERSØGELSER AF DE EKSPERIMENTELLE FORUDSÆTNINGER FOR RUTHERFORDS ATOMMODEL".
Projektrapport af: Niels Thor Nielsen.
Vejleder: Bent C. Jørgensen.

45/82 Er aldrig udkommet.

46/82 "EKSEMPLARISK UNDERVISNING OG FYSISK ERKENDESE-
1+11 ILLUSTRERET VED TO EKSEMPLER".
Projektrapport af: Torben O.Olsen, Lasse Rasmussen og Niels Dreyer Sørensen.
Vejleder: Bent C. Jørgensen.

47/82 "BARSEBÄCK OG DET VÆRST OFFICIELT-TÆNKELIGE UHELD".
ENERGY SERIES NO. 5.
Af: Bent Sørensen.

48/82 "EN UNDERSØGELSE AF MATEMATIKUNDERVISNINGEN PÅ ADGANGSKURSUS TIL KØBENHAVNS TEKNIKUM".
Projektrapport af: Lis Eilertzen, Jørgen Karrebæk, Troels Lange, Preben Nørregaard, Lissi Pedesen, Laust Rishøj, Lill Røn og Isac Showiki.
Vejleder: Mogens Niss.

49/82 "ANALYSE AF MULTISPEKTRALE SATELLITBILLEDER".
Projektrapport af: Preben Nørregaard.
Vejledere: Jørgen Larsen og Rasmus Ole Rasmussen.

50/82 "HERSLEV - MULIGHEDER FOR VEDVARENDE ENERGI I EN LANDSBY".
ENERGY SERIES NO. 6.
Rapport af: Bent Christensen, Bent Hove Jensen, Dennis B. Møller, Bjarne Laursen, Bjarne Lillethorup og Jacob Mørch Pedersen.
Vejleder: Bent Sørensen.

51/82 "HVAD KAN DER GØRES FOR AT AFHJÆLPE PIGERS BLOKERING OVERFOR MATEMATIK ?"
Projektrapport af: Lis Eilertzen, Lissi Pedersen, Lill Røn og Susanne Stender.

52/82 "DESUSPENSION OF SPLITTING ELLIPTIC SYMBOLS".
Af: Bernhelm Booss og Krzysztof Wojciechowski.

53/82 "THE CONSTITUTION OF SUBJECTS IN ENGINEERING EDUCATION".
Af: Arne Jacobsen og Stig Andur Pedersen.

54/82 "FUTURES RESEARCH" - A Philosophical Analysis of Its Subject-Matter and Methods.
Af: Stig Andur Pedersen og Johannes Witt-Hansen.

55/82 "MATEMATISKE MODELLER" - Litteratur på Roskilde Universitetsbibliotek.
En biografi.
Af: Else Høyrup.

Vedr. tekst nr. 55/82 se også tekst nr. 62/83.

56/82 "EN - TO - MANGE" -
En undersøgelse af matematisk økologi.
Projektrapport af: Troels Lange.
Vejleder: Anders Madsen.

57/83 "ASPECT EKSPERIMENTET"-
Skjulte variable i kvantemekanikken?
Projektrapport af: Tom Juul Andersen.
Vejleder: Peder Voetmann Christiansen.
Nr. 57 er udgået.

58/83 "MATEMATISKE VANDRINGER" - Modelbetragtninger over spredning af dyr mellem småbiotoper i agerlandet.
Projektrapport af: Per Hammershøj Jensen og Lene Vagn Rasmussen.
Vejleder: Jørgen Larsen.

59/83 "THE METHODOLOGY OF ENERGY PLANNING".
ENERGY SERIES NO. 7.
Af: Bent Sørensen.

60/83 "MATEMATISK MODEKSPERTISE"- et eksempel.
Projektrapport af: Erik O. Gade, Jørgen Karrebæk og Preben Nørregaard.
Vejleder: Anders Madsen.

61/83 "FYSIKS IDEOLOGISKE FUNKTION, SOM ET EKSEMPEL PÅ EN NATURVIDENSKAB - HISTORISK SET".
Projektrapport af: Annette Post Nielsen.
Vejledere: Jens Høyrup, Jens Højgaard Jensen og Jørgen Vogelius.

62/83 "MATEMATISKE MODELLER" - Litteratur på Roskilde Universitetsbibliotek.
En biografi 2. rev. udgave.
Af: Else Høyrup.

63/83 "CREATING ENERGY FUTURES:A SHORT GUIDE TO ENERGY PLANNING".
ENERGY SERIES No. 8.
Af: David Crossley og Bent Sørensen.

64/83 "VON MATEMATIK UND KRIEG".
Af: Berhelm Booss og Jens Høyrup.

65/83 "ANVENDT MATEMATIK - TEORI ELLER PRAKSIS".
Projektrapport af: Per Hedegård Andersen, Kirsten Habekost, Carsten Holst-Jensen, Annelise von Moos, Else Marie Pedersen og Erling Møller Pedersen.
Vejledere: Bernhelm Booss og Klaus Grünbaum.

66/83 "MATEMATISKE MODELLER FOR PERIODISK SELEKTION I ESCHERICHIA COLI".
Projektrapport af: Hanne Lisbet Andersen, Ole Richard Jensen og Klavs Frisdahl.
Vejledere: Jørgen Larsen og Anders Hede Madsen.

67/83 "ELEPSOIDE METODEN - EN NY METODE TIL LINEÆR PROGRAMMERING?"
Projektrapport af: Lone Biilmann og Lars Boye.
Vejleder: Mogens Brun Heefelt.

68/83 "STOKASTISKE MODELLER I POPULATIONSGENETIK"
- til kritikken af teoriladede modeller.
Projektrapport af: Lise Odgård Gade, Susanne Hansen, Michael Hviid og Frank Mølgård Olsen.
Vejleder: Jørgen Larsen.

69/83 "ELEVFORUDSÆTNINGER I FYSIK"
- en test i 1.g med kommentarer.
Af: Albert C. Paulsen.

70/83 "INDLÆRINGS - OG FORMIDLINGSPROBLEMER I MATEMATIK PÅ VOKSENUNDERVISNINGSNIVEAU".
Projektrapport af: Hanne Lisbet Andersen, Torben J. Andreasen, Svend Åge Houmann, Helle Glerup Jensen, Keld Fl. Nielsen, Lene Vagn Rasmussen.
Vejleder: Klaus Grünbaum og Anders Hede Madsen.

71/83 "PIGER OG FYSIK"
- et problem og en udfordring for skolen?
Af: Karin Beyer, Sussanne Blegaa, Birthe Olsen, Jette Reich og Mette Vedelsby.

72/83 "VERDEN IFØLGE PEIRCE" - to metafysiske essays, om og af C.S Peirce.
Af: Peder Voetmann Christiansen.

73/83 ""EN ENERGIANALYSE AF LANDBRUG"
- økologisk contra traditionelt.
ENERGY SERIES NO. 9
Specialeopgave i fysik af: Bent Hove Jensen.
Vejleder: Bent Sørensen.

74/84 "MINIATURISERING AF MIKROELEKTRONIK" - om videnskabeliggjort teknologi og nytten af at lære fysik.
Projektrapport af: Bodil Harder og Linda Szkotak Jensen.
Vejledere: Jens Højgaard Jensen og Bent C. Jørgensen.

75/84 "MATEMATIKUNDERVISNINGEN I FREMTIDENS GYMNASIUM"
- Case: Lineær programmering.
Projektrapport af: Morten Blomhøj, Klavs Frisdahl og Frank Mølgaard Olsen.
Vejledere: Mogens Brun Heefelt og Jens Bjørneboe.

76/84 "KERNEKRAFT I DANMARK?" - Et høringssvar indkaldt af miljøministeriet, med kritik af miljøstyrelsens rapporter af 15. marts 1984.
ENERGY SERIES No. 10
Af: Niels Boye Olsen og Bent Sørensen.

77/84 "POLITISKE INDEKS - FUP ELLER FAKTA?"
Opinionsundersøgelser belyst ved statistiske modeller.
Projektrapport af: Svend Åge Houmann, Keld Nielsen og Susanne Stender.
Vejledere: Jørgen Larsen og Jens Bjørneboe.

78/84 "JÆVNSTRØMSLEDNINGSEVNE OG GITTERSTRUKTUT I AMORFT GERMANIUM".
Specialrapport af: Hans Hedal, Frank C. Ludvigsen og Finn C. Physant.
Vejleder: Niels Boye Olsen.

79/84 "MATEMATIK OC ALMENDANNELSE".
Projektrapport af: Henrik Coster, Mikael Wennerberg Johansen, Povl Kattler, Birgitte Lydholm og Morten Overgaard Nielsen.
Vejleder: Bernhelm Booss.

80/84 "KURSUSMATERIALE TIL MATEMATIK B".
Af: Mogens Brun Heefelt.

81/84 "FREKVENSAFHÆNGIG LEDNINGSEVNE I AMORFT GERMANIUM".
Specialerapport af: Jørgen Wind Petersen og Jan Christensen.
Vejleder: Niels Boye Olsen.

82/84 "MATEMATIK - OG FYSIKUNDERVISNINGEN I DET AUTOMATISEREDE SAMFUND".
Rapport fra et seminar afholdt i Hvidovre 25-27 april 1983.
Red.: Jens Højgaard Jensen, Bent C. Jørgensen og Mogens Niss.

83/84 "ON THE QUANTIFICATION OF SECURITY":
PEACE RESEARCH SERIES NO. 1
Af: Bent Sørensen
nr. 83 er p.t. udgået

84/84 "NOGLE ARTIKLER OM MATEMATIK, FYSIK OG ALMENDANNELSE".
Af: Jens Højgaard Jensen, Mogens Niss m. fl.

85/84 "CENTRIFUGALREGULATORER OG MATEMATIK".
Specialerapport af: Per Hedegård Andersen, Carsten Holst-Jensen, Else Marie Pedersen og Erling Møller Pedersen.
Vejleder: Stig Andur Pedersen.

86/84 "SECURITY IMPLICATIONS OF ALTERNATIVE DEFENSE OPTIONS FOR WESTERN EUROPE".
PEACE RESEARCH SERIES NO. 2
Af: Bent Sørensen.

87/84 "A SIMPLE MODEL OF AC HOPPING CONDUCTIVITY IN DISORDERED SOLIDS".
Af: Jeppe C. Dyre.

88/84 "RISE, FALL AND RESURRECTION OF INFINITESIMALS".
Af: Detlef Laugwitz.

89/84 "FJERNVARMEOPTIMERING".
Af: Bjarne Lillethorup og Jacob Mørch Pedersen.

90/84 "ENERGI I 1.G - EN TEORI FOR TILRETTELÆGGELSE".
Af: Albert Chr. Paulsen.

91/85 "KVANTETEORI FOR GYMNASIET".
1. Lærervejledning
Projektrapport af: Biger Lundgren, Henning Sten Hansen og John Johansson.
Vejleder: Torsten Meyer.

92/85 "KVANTETEORI FOR GYMNASIET".
2. Materiale
Projektrapport af: Biger Lundgren, Henning Sten Hansen og John Johansson.
Vejleder: Torsten Meyer.

93/85 "THE SEMIOTICS OF QUANTUM - NON - LOCALITY".
Af: Peder Voetmann Christiansen.

94/85 "TREENIGHEDEN BOURBAKI - generalen, matematikeren og ånden".
Projektrapport af: Morten Blomhøj, Klavs Frisdahl og Frank M. Olsen.
Vejleder: Mogens Niss.

95/85 "AN ALTERNATIV DEFENSE PLAN FOR WESTERN EUROPE".
PEACE RESEARCH SERIES NO. 3
Af: Bent Sørensen

96/85 "ASPEKTER VED KRAFTVARMEFORSYNING".
Af: Bjarne Lilletorup.
Vejleder: Bent Sørensen.

97/85 "ON THE PHYSICS OF A.C. HOPPING CONDUCTIVITY".
Af: Jeppe C. Dyre.

98/85 "VALGMULIGHEDER I INFORMATIONSALDEREN".
Af: Bent Sørensen.

99/85 "Der er langt fra Q til R".
Projektrapport af: Niels Jørgensen og Mikael Klintorp.
Vejleder: Stig Andur Pedersen.

100/85 "TALSYSTEMETS OPBYGNING".
Af: Mogens Niss.

101/85 "EXTENDED MOMENTUM THEORY FOR WINDMILLS IN PERTURBATIVE FORM".
Af: Ganesh Sengupta.

102/85 OPSTILLING OG ANALYSE AF MATEMATISKE MODELLER, BELYST VED MODELLER OVER KØERS FODEROPTAGELSE OG - OMSÆTNING".
Projektrapport af: Lis Eilertzen, Kirsten Habekost, Lill Røn og Susanne Stender.
Vejleder: Klaus Grünbaum.

103/85 "ØDSLE KOLDKRIGERE OG VIDENSKABENS LYSE IDEER".
Projektrapport af: Niels Ole Dam og Kurt Jensen.
Vejleder: Bent Sørensen.

104/85 "ANALOGREGNEMASKINEN OG LORENZLIGNINGER".
Af: Jens Jæger.

105/85 "THE FREQUENCY DEPENDENCE OF THE SPECIFIC HEAT AF THE GLASS REANSITION".
Af: Tage Christensen.

"A SIMPLE MODEL AF AC HOPPING CONDUCTIVITY".
Af: Jeppe C. Dyre.
Contributions to the Third International Conference on the Structure of Non - Crystalline Materials held in Grenoble July 1985.

106/85 "QUANTUM THEORY OF EXTENDED PARTICLES".
Af: Bent Sørensen.

107/85 "EN MYG GØR INGEN EPIDEMI".
- flodblindhed som eksempel på matematisk modellering af et epidemiologisk problem.
Projektrapport af: Per Hedegård Andersen, Lars Boye, CarstenHolst Jensen, Else Marie Pedersen og Erling Møller Pedersen.
Vejleder: Jesper Larsen.

108/85 "APPLICATIONS AND MODELLING IN THE MATEMATICS CURRICULUM" - state and trends -
Af: Mogens Niss.

109/85 "COX I STUDIETIDEN" - Cox's regressionsmodel anvendt på studenteroplysninger fra RUC.
Projektrapport af: Mikael Wennerberg Johansen, Poul Katler og Torben J. Andreasen.
Vejleder: Jørgen Larsen.

110/85 "PLANNING FOR SECURITY".
Af: Bent Sørensen

111/85 JORDEN RUNDT PÅ FLADE KORT".
Projektrapport af: Birgit Andresen, Beatriz Quinones og Jimmy Staal.
Vejleder: Mogens Niss.

112/85 "VIDENSKABELIGGØRELSE AF DANSK TEKNOLOGISK INNOVATION FREM TIL 1950 - BELYST VED EKSEMPLER".
Projektrapport af: Erik Odgaard Gade, Hans Hedal, Frank C. Ludvigsen, Annette Post Nielsen og Finn Physant.
Vejleder: Claus Bryld og Bent C. Jørgensen.

113/85 "DESUSPENSION OF SPLITTING ELLIPTIC SYMBOLS II".
Af: Bernhelm Booss og Krzysztof Wojciechowski.

114/85 "ANVENDELSE AF GRAFISKE METODER TIL ANALYSE AF KONTIGENSTABELLER".
Projektrapport af: Lone Biilmann, Ole R. Jensen og Anne-Lise von Moos.
Vejleder: Jørgen Larsen.

115/85 "MATEMATIKKENS UDVIKLING OP TIL RENÆSSANCEN".
Af: Mogens Niss.

116/85 "A PHENOMENOLOGICAL MODEL FOR THE MEYER-NELDEL RULE".
Af: Jeppe C. Dyre.

117/85 "KRAFT & FJERNVARMEOPTIMERING"
Af: Jacob Mørch Pedersen.
Vejleder: Bent Sørensen

118/85 "TILFÆLDIGHEDEN OG NØDVENDIGHEDEN IFØLGE PEIRCE OG FYSIKKEN".
Af: Peder Voetmann Christiansen

119/86 "DET ER GANSKE VIST - - EUKLIDS FEMTE POSTULAT KUNNE NOK SKABE RØRE I ANDEDAMMEN".
Af: Iben Maj Christiansen
Vejleder: Mogens Niss.

120/86 "ET ANTAL STATISTISKE STANDARDMODELLER".
Af: Jørgen Larsen

121/86 "SIMULATION I KONTINUERT TID".
Af: Peder Voetmann Christiansen.

122/86 "ON THE MECHANISM OF GLASS IONIC CONDUCTIVITY".
Af: Jeppe C. Dyre.

123/86 "GYMNASIEFYSIKKEN OG DEN STORE VERDEN".
Fysiklærerforeningen, IMFUFA, RUC.

124/86 "OPGAVESAMLING I MATEMATIK".
Samtlige opgaver stillet i tiden 1974-jan. 1986.

125/86 "UVBY,β - systemet - en effektiv fotometrisk spektralklassifikation af B-,A- og F-stjerner".
Projektrapport af: Birger Lundgren.

126/86 "OM UDVIKLINGEN AF DEN SPECIELLE RELATIVITETSTEORI".
Projektrapport af: Lise Odgaard & Linda Szkotak Jensen
Vejledere: Karin Beyer & Stig Andur Pedersen.

127/86 "GALOIS' BIDRAG TIL UDVIKLINGEN AF DEN ABSTRAKTE ALGEBRA".
Projektrapport af: Pernille Sand, Heine Larsen & Lars Frandsen.
Vejleder: Mogens Niss.

128/86 "SMÅKRYB" - om ikke-standard analyse.
Projektrapport af: Niels Jørgensen & Mikael Klintorp.
Vejleder: Jeppe Dyre.

129/86 "PHYSICS IN SOCIETY"
Lecture Notes 1983 (1986)
Af: Bent Sørensen

130/86 "Studies in Wind Power"
Af: Bent Sørensen

131/86 "FYSIK OG SAMFUND" - Et integreret fysik/historieprojekt om naturanskuelsens historiske udvikling og dens samfundsmæssige betingethed.
Projektrapport af: Jakob Heckscher, Søren Brønd, Andy Wierød.
Vejledere: Jens Høyrup, Jørgen Vogelius, Jens Højgaard Jensen.

132/86 "FYSIK OG DANNELSE"
Projektrapport af: Søren Brønd, Andy Wierød.
Vejledere: Karin Beyer, Jørgen Vogelius.

133/86 "CHERNOBYL ACCIDENT: ASSESSING THE DATA.
ENERGY SERIES NO. 15.
AF: Bent Sørensen.

134/87 "THE D.C. AND THE A.C. ELECTRICAL TRANSPORT IN AsSeTe SYSTEM"
Authors: M.B.El-Den, N.B.Olsen, Ib Høst Pedersen, Petr Viščor

135/87 "INTUITIONISTISK MATEMATIKS METODER OG ERKENDELSESTEORETISKE FORUDSÆTNINGER"
MASTEMATIKSPECIALE: Claus Larsen
Vejledere: Anton Jensen og Stig Andur Pedersen

136/87 "Mystisk og naturlig filosofi: En skitse af kristendommens første og andet møde med græsk filosofi"
Projektrapport af Frank Colding Ludvigsen
Vejledere: Historie: Ib Thiersen
Fysik: Jens Højgaard Jensen

137/87 "HOPMODELLER FOR ELEKTRISK LEDNING I UORDNEDE FASTE STOFFER" - Resume af licentiatafhandling
Af: Jeppe Dyre
Vejledere: Niels Boye Olsen og Peder Voetmann Christiansen.

138/87 "JOSEPHSON EFFECT AND CIRCLE MAP."

Paper presented at The International Workshop on Teaching Nonlinear Phenomena at Universities and Schools, "Chaos in Education". Balaton, Hungary, 26 April-2 May 1987.

By: Peder Voetmann Christiansen

139/87 "Machbarkeit nichtbeherrschbarer Technik durch Fortschritte in der Erkennbarkeit der Natur"

Af: Bernhelm Booss-Bavnbek
Martin Bohle-Carbonell

140/87 "ON THE TOPOLOGY OF SPACES OF HOLOMORPHIC MAPS"

By: Jens Gravesen

141/87 "RADIOMETERS UDVIKLING AF BLODGASAPPARATUR - ET TEKNOLOGIHISTORISK PROJEKT"

Projektrapport af Finn C. Physant

Vejleder: Ib Thiersen

142/87 "The Calderón Projektor for Operators With Splitting Elliptic Symbols"

by: Bernhelm Booss-Bavnbek og
Krzysztof P. Wojciechowski

143/87 "Kursusmateriale til Matematik på NAT-BAS"

af: Mogens Brun Heefelt

144/87 "Context and Non-Locality - A Peircean Approach

Paper presented at the Symposium on the Foundations of Modern Physics The Copenhagen Interpretation 60 Years after the Como Lecture. Joensuu, Finland, 6 - 8 august 1987.

By: Peder Voetmann Christiansen

145/87 "AIMS AND SCOPE OF APPLICATIONS AND MODELLING IN MATHEMATICS CURRICULA"

Manuscript of a plenary lecture delivered at ICMTA 3, Kassel, FRG 8.-11.9.1987

By: Mogens Niss

146/87 "BESTEMMELSE AF BULKRESISTIVITETEN I SILICIUM"

- en ny frekvensbaseret målemetode.

Fysikspeciale af Jan Vedde

Vejledere: Niels Boye Olsen & Petr Viščor

147/87 "Rapport om BIS på NAT-BAS"

redigeret af: Mogens Brun Heefelt

148/87 "Naturvidenskabsundervisning med Samfundsperspektiv"

af: Peter Colding-Jørgensen DLH
Albert Chr. Paulsen

149/87 "In-Situ Measurements of the density of amorphous germanium prepared in ultra high vacuum"

by: Petr Viščor

150/87 "Structure and the Existence of the first sharp diffraction peak in amorphous germanium prepared in UHV and measured in-situ"

by: Petr Viščor

151/87 "DYNAMISK PROGRAMMERING"

Matematikprojekt af:
Birgit Andresen, Keld Nielsen og Jimmy Staal

Vejleder: Mogens Niss

152/87 "PSEUDO-DIFFERENTIAL PROJECTIONS AND THE TOPOLOGY OF CERTAIN SPACES OF ELLIPTIC BOUNDARY VALUE PROBLEMS"

by: Bernhelm Booss-Bavnbek
Krzysztof P. Wojciechowski

153/88 "HALVLEDERTEKNOLOGIENS UDVIKLING MELLEM MILITÆRE OG CIVILE KRÆFTER"

Et eksempel på humanistisk teknologihistorie
Historiespeciale

Af: Hans Hedal

Vejleder: Ib Thiersen

154/88 "MASTER EQUATION APPROACH TO VISCOUS LIQUIDS AND THE GLASS TRANSITION"

By: Jeppe Dyre

155/88 "A NOTE ON THE ACTION OF THE POISSON SOLUTION OPERATOR TO THE DIRICHLET PROBLEM FOR A FORMALLY SELFADJOINT DIFFERENTIAL OPERATOR"

by: Michael Pedersen

156/88 "THE RANDOM FREE ENERGY BARRIER MODEL FOR AC CONDUCTION IN DISORDERED SOLIDS"

by: Jeppe C. Dyre

157/88 " STABILIZATION OF PARTIAL DIFFERENTIAL EQUATIONS BY FINITE DIMENSIONAL BOUNDARY FEEDBACK CONTROL: A pseudo-differential approach."

by: Michael Pedersen

158/88 "UNIFIED FORMALISM FOR EXCESS CURRENT NOISE IN RANDOM WALK MODELS"

by: Jeppe Dyre

159/88 "STUDIES IN SOLAR ENERGY"

by: Bent Sørensen

160/88 "LOOP GROUPS AND INSTANTONS IN DIMENSION TWO"

by: Jens Gravesen

161/88 "PSEUDO-DIFFERENTIAL PERTURBATIONS AND STABILIZATION OF DISTRIBUTED PARAMETER SYSTEMS:

Dirichlet feedback control problems"

by: Michael Pedersen

162/88 "PIGER & FYSIK - OG MEGET MERE"

AF: Karin Beyer, Sussanne Blegaa, Birthe Olsen,
Jette Reich , Mette Vedelsby

163/88 "EN MATEMATISK MODEL TIL BESTEMMELSE AF PERMEABILITETEN FOR BLOD-NETHINDE-BARRIEREN"

Af: Finn Langberg, Michael Jarden, Lars Frellesen

Vejleder: Jesper Larsen

164/88 "Vurdering af matematisk teknologi
Technology Assessment
Technikfolgenabschätzung"

Af: Bernhelm Booss-Bavnbek, Glen Pate med
Martin Bohle-Carbonell og Jens Højgaard Jensen

165/88 "COMPLEX STRUCTURES IN THE NASH-MOSER CATEGORY"

by: Jens Gravesen

166/88 "Grundbegreber i Sandsynligheds-regningen"

Af: Jørgen Larsen

167a/88 "BASISSTATISTIK 1. Diskrete modeller"

Af: Jørgen Larsen

167b/88 "BASISSTATISTIK 2. Kontinuerte modeller"

Af: Jørgen Larsen

168/88 "OVERFLADEN AF PLANETEN MARS"
Laboratorie-simulering og MARS-analoger undersøgt ved Mössbauerspektroskopi.

Fysikspeciale af:

Birger Lundgren

Vejleder: Jens Martin Knudsen
Fys.Lab./HCØ

169/88 "CHARLES S. PEIRCE: MURSTEN OG MØRTEL TIL EN METAFYSIK."

Fem artikler fra tidsskriftet "The Monist" 1891-93.

Introduktion og oversættelse:

Peder Voetmann Christiansen

170/88 "OPGAVESAMLING I MATEMATIK"

Samtlige opgaver stillet i tiden 1974 - juni 1988

171/88 "The Dirac Equation with Light-Cone Data"

af: Johnny Tom Ottesen

172/88 "FYSIK OG VIRKELIGHED"

Kvantemekanikkens grundlagsproblem i gymnasiet.

Fysikprojekt af:

Erik Lund og Kurt Jensen

Vejledere: Albert Chr. Paulsen og Peder Voetmann Christiansen

---

173/89 "NUMERISKE ALGORITMER"

af: Mogens Brun Heefelt

174/89 " GRAFISK FREMSTILLING AF FRAKTALER OG KAOS"

af: Peder Voetmann Christiansen

175/89 " AN ELEMENTARY ANALYSIS OF THE TIME DEPENDENT SPECTRUM OF THE NON-STATONARY SOLUTION TO THE OPERATOR RICCATI EQUATION

af: Michael Pedersen

176/89 " A MAXIUM ENTROPY ANSATZ FOR NONLINEAR RESPONSE THEORY"

af : Jeppe Dyre

177/89 "HVAD SKAL ADAM STÅ MODEL TIL"

af: Morten Andersen, Ulla Engström, Thomas Gravesen, Nanna Lund, Pia Madsen, Dina Rawat, Peter Torstensen

Vejleder: Mogens Brun Heefelt

178/89 "BIOSYNTESEN AF PENICILLIN - en matematisk model"

af: Ulla Eghave Rasmussen, Hans Oxvang Mortensen, Michael Jarden

vejleder i matematik: Jesper Larsen
biologi: Erling Lauridsen

179a/89 "LÆRERVEJLEDNING M.M. til et eksperimentelt forløb om kaos"

af: Andy Wierød, Søren Brønd og Jimmy Staal

Vejledere: Peder Voetmann Christiansen
Karin Beyer

179b/89 "ELEVHÆFTE: Noter til et eksperimentelt kursus om kaos"

af: Andy Wierød, Søren Brønd og Jimmy Staal

Vejledere: Peder Voetmann Christiansen
Karin Beyer

180/89 "KAOS I FYSISKE SYSTEMER eksemplificeret ved torsions- og dobbeltpendul".

af: Andy Wierød, Søren Brønd og Jimmy Staal
Vejleder: Peder Voetmann Christiansen

181/89 "A ZERO-PARAMETER CONSTITUTIVE RELATION FOR PURE SHEAR VISCOELASTICITY"

by: Jeppe Dyre

183/89 "MATEMATICAL PROBLEM SOLVING, MODELLING. APPLICATIONS AND LINKS TO OTHER SUBJECTS - State. trends and issues in mathematics instruction

by: WERNER BLUM, Kassel (FRG) og
MOGENS NISS, Roskilde (Denmark)

184/89 "En metode til bestemmelse af den frekvensafhængige varmefylde af en underafkølet væske ved glasovergange"

af: Tage Emil Christensen

185/90 "EN MESTEN PERIODISK HISTORIE"
Et matematisk projekt
af: Steen Grode og Thomas Jessen
Vejleder: Jacob Jacobsen

186/90 "RITUAL OG RATIONALITET i videnskabers udvikling"
redigeret af Arne Jakobsen og Stig Andur Pedersen

187/90 "RSA - et kryptografisk system"
af: Annemette Sofie Olufsen, Lars Frellesen
og Ole Møller Nielsen
Vejledere: Michael Pedersen og Finn Munk

188/90 "FERMICONDENSATION - AN ALMOST IDEAL GLASS TRANSITION"
by: Jeppe Dyre

189/90 "DATAMATER I MATEMATIKUNDERVISNINGEN PÅ GYMNASIET OG HØJERE LÆREANSTALTER

af: Finn Langberg

190/90 "FIVE REQUIREMENTS FOR AN APPROXIMATE NONLINEAR RESPONSE THEORY"

by: Jeppe Dyre

191/90 "MOORE COHOMOLOGY, PRINCIPAL BUNDLES AND ACTIONS OF GROUPS ON C*-ALGEBRAS"

by: Iain Raeburn and Dana P. Williams

192/90 "Age-dependent host mortality in the dynamics of endemic infectious diseases and SIR-models of the epidemiology and natural selection of co-circulating influenza virus with partial cross-immunity"

by: Viggo Andreasen

193/90 "Causal and Diagnostic Reasoning"

by: Stig Andur Pedersen

194a/90 "DETERMINISTISK KAOS"
Projektrapport af : Frank Olsen

194b/90 "DETERMINISTISK KAOS"
Kørselsrapport

Projektrapport af: Frank Olsen

195/90 "STADIER PÅ PARADIGMETS VEJ"
Et projekt om den videnskabelige udvikling der førte til dannelse af kvantemekanikken.

Projektrapport for 1. modul på fysikuddannelsen, skrevet af:

Anja Boisen, Thomas Hougård, Anders Gorm Larsen, Nicolai Ryge.

Vejleder: Peder Voetmann Christiansen

196/90 "ER KAOS NØDVENDIGT?"

- en projektrapport om kaos' paradigmatiske status i fysikken.

af: Johannes K. Nielsen, Jimmy Staal og Peter Bøggild

Vejleder: Peder Voetmann Christiansen

197/90 "Kontrafaktiske konditionaler i HOL

af: Jesper Voetmann, Hans Oxvang Mortensen og Aleksander Høst-Madsen

Vejleder: Stig Andur Pedersen

198/90 "Metal-Isolator-Metal systemer"

Speciale

af: Frank Olsen

199/90 "SPREDT FÆGTNING" Artikelsamling

af: Jens Højgaard Jensen

200/90 "LINEÆR ALGEBRA OG ANALYSE"

Noter til den naturvidenskabelige basisuddannelse.

af: Mogens Niss

201/90 **"Undersøgelse af atomare korrelationer i amorfe stoffer ved røntgendiffraktion"**

af: Karen Birkelund og Klaus Dahl Jensen

Vejledere: Petr Viščor, Ole Bakander

202/90 **"TEGN OG KVANTER"**
Foredrag og artikler, 1971-90.

af: Peder Voetmann Christiansen

203/90 **"OPGAVESAMLING I MATEMATIK" 1974-1990**

afløser tekst 170/88

204/91 "ERKENDELSE OG KVANTEMEKANIK"

et Breddemodul Fysik Projekt

af: Thomas Jessen

Vejleder: Petr Viščor

205/91 "PEIRCE'S LOGIC OF VAGUENESS"

by: Claudine Engel-Tiercelin
Department of Philosophy
Université de Paris-1
(Panthéon-Sorbonne)

206a+b/91 "GERMANIUMBEAMANALYSE SAMT A - GE TYNDFILMS ELEKTRISKE EGENSKABER"

Eksperimentelt Fysikspeciale

af: Jeanne Linda Mortensen

og Annette Post Nielsen

Vejleder: Petr Viščor

207/91 "SOME REMARKS ON AC CONDUCTION IN DISORDERED SOLIDS"

by: Jeppe C. Dyre

208/91 "LANGEVIN MODELS FOR SHEAR STRESS FLUCTUATIONS IN FLOWS OF VISCO-ELASTIC LIQUIDS"

by: Jeppe C. Dyre

209/91 "LORENZ GUIDE" Kompendium til den danske fysiker Ludvig Lorenz, 1829-91.

af: Helge Kragh

210/91 "Global Dimension, Tower of Algebras, and Jones Index of Split Seperable Subalgebras with Unitality Condition.

by: Lars Kadison

211/91 "I SANDHEDENS TJENESTE"

- historien bag teorien for de komplekse tal.

af: Lise Arleth, Charlotte Gjerrild, Jane Hansen, Linda Kyndlev, Anne Charlotte Nilsson, Kamma Tulinius.

Vejledere: Jesper Larsen og Bernhelm Booss-Bavnbek

212/91 "Cyclic Homology of Triangular Matrix Algebras"

by: Lars Kadison

213/91 "Disease-induced natural selection in a diploid host

by: Viggo Andreasen and Freddy B. Christiansen

214|91 "Halløj i æteren" - om elektromagnetisme. Oplæg til undervisningsmateriale i gymnasiet.

Af: Nils Kruse, Peter Gastrup, Kristian Hoppe, Jeppe Guldager

Vejledere: Petr Viscor, Hans Hedal

215|91 "Physics and Technology of Metal-Insulator-Metal thin film structures used as planar electron emitters

by: A.Delong, M.Drsticka, K.Hladil, V.Kolarik, F.Olsen, P.Pavelka and Petr Viscor.

216|91 "Kvantemekanik på PC'eren"

af: Thomas Jessen

217/92 "Two papers on APPLICATIONS AND MODELLING IN THE MATHEMATICS CURRICULUM"

by: Mogens Niss

218/92 "A Three-Square Theorem"

by: Lars Kadison

219/92 "RUPNOK - stationær strømning i elastiske rør"

af: Anja Boisen, Karen Birkelund, Mette Olufsen

Vejleder: Jesper Larsen

220/92 "Automatisk diagnosticering i digitale kredsløb"

af: Bjørn Christensen, Ole Møller Nielsen

Vejleder: Stig Andur Pedersen

221/92 "A BUNDLE VALUED RADON TRANSFORM, WITH APPLICATIONS TO INVARIANT WAVE EQUATIONS"

by: Thomas P. Branson, Gestur Olafsson and Henrik Schlichtkrull

222/92 On the Representations of some Infinite Dimensional Groups and Algebras Related to Quantum Physics

by: Johnny T. Ottesen

223/92 *THE FUNCTIONAL DETERMINANT*

by: Thomas P. Branson

224/92 *UNIVERSAL AC CONDUCTIVITY OF NON-METALLIC SOLIDS AT LOW TEMPERATURES*

by: Jeppe C. Dyre

225/92 *"HATMODELLEN" Impedansspektroskopi i ultrarent en-krystallinsk silicium*

af: Anja Boisen, Anders Gorm Larsen, Jesper Varmer, Johannes K. Nielsen, Kit R. Hansen, Peter Bøggild og Thomas Hougaard

Vejleder: Petr Viscor

226/92 *"METHODS AND MODELS FOR ESTIMATING THE GLOBAL CIRCULATION OF SELECTED EMISSIONS FROM ENERGY CONVERSION"*

by: Bent Sørensen

227/92 "Computersimulering og fysik"

af: Per M.Hansen, Steffen Holm, Peter Maibom, Mads K. Dall Petersen, Pernille Postgaard, Thomas B.Schrøder, Ivar P. Zeck

Vejleder: Peder Voetmann Christiansen

228/92 "Teknologi og historie"

Fire artikler af:

Mogens Niss, Jens Høyrup, Ib Thiersen, Hans Hedal

229/92 "Masser af information uden betydning"

En diskussion af informationsteorien i Tor Nørretranders' "Mærk Verden" og en skitse til et alternativ basseret på andenordens kybernetik og semiotik.

af: Søren Brier

230/92 "Vinklens tredeling - et klassisk problem"

et matematisk projekt af

Karen Birkelund, Bjørn Christensen

Vejleder: Johnny Ottesen

231A/92 "Elektrondiffusion i silicium - en matematisk model"

af: Jesper Voetmann, Karen Birkelund, Mette Olufsen, Ole Møller Nielsen

Vejledere: Johnny Ottesen, H.B.Hansen

231B/92 "Elektrondiffusion i silicium - en matematisk model" Kildetekster

af: Jesper Voetmann, Karen Birkelund, Mette Olufsen, Ole Møller Nielsen

Vejledere: Johnny Ottesen, H.B.Hansen