```
PROGRAM TIL INVERSION AF LAPLACE TRANSFORM 120365 edda sveinsdottir
dog først
Procedurer til styring af CALCOMP-plotter.
ln gamma af komplex tal tilføjet fra GSL af MK.
begin
   integer xxx,yyy,l111,pl111,plabc;
   real deltax,deltay;
   procedure clock(print);
   value print;
   boolean print;
   begin
      real c1,c2;
      comment
         1d 2h 3m 45.67s = 45.67 + 3×60 + 2×60×60 + 1×60×60×24 = 93825.67s;
      procedure printit(c, text);
      value c;
      real c;
      string text;
      begin
         real d,h,m,s;
         integer id,ih,im;
         procedure outinteger(n);
         value n;
         integer n;
         begin
            integer mod,digit;
            boolean started;
            started := false;
            mod:=100000;
            for mod:=mod while mod>0 do
            begin
               digit := n:mod;
               if digit>0 ∨ started then
               begin
                  outchar(if digit=0 then 16 else digit);
                  started := true
               end;
               n := n - digit×mod;
               mod := mod : 10
            end mod
         end outinteger;

         outcr;
         outtext(text);
         output(⊰ndddddddddd.dd⊱,c);
         outtext(⊰<s = ⊱);
         h:=c/3600;
         id:=entier(h/24);
         h:=h-id×24;
         ih:=entier(h);
         h:=h-ih;
         h:=h×60;
         im:=entier(h);
         h:=h-im;
         s:=h×60;
         if id>0 then
         begin
            outinteger(id);
            outtext(⊰<d ⊱)
         end;


comment
```

```
;
         if ih>0 then
         begin
            outinteger(ih);
            outtext(⊰<h ⊱)
         end;
         if im>0 then
         begin
            outinteger(im);
            outtext(⊰<m ⊱)
         end;
         output(if s≥10 then ⊰nd.dd⊱ else ⊰n.dd⊱,s);
         outtext(⊰<s⊱)
      end printit;
      boolean code1,code2;
      pack(code1, 0, 41, 0);
      pack(code1,
            0,  9,  0,
           10, 19,  1,
           20, 25, 62,
           30, 35, 17,
           39, 39,  1,
           40, 40,  1);
      c1:=gier(code1);
      pack(code2, 0, 41, 0);
      pack(code2,
            0,  9,  1,
           10, 19,  1,
           20, 25, 62,
           30, 35, 17,
           39, 39,  1,
           40, 40,  1);
      c2:=gier(code2);
      if print then
      begin
         printit(c1, ⊰<GIER clock time: ⊱);
         printit(c2, ⊰<Wall-clock time: ⊱)
      end print
   end clock;

   procedure plotgraph(x0,x1,x,y,dx);
   value x0,x1;
   real x0,x1,x,y,dx;
   begin
      integer xx,yy,tr;
      boolean hop;
      boolean array B[1:l111];
      tr:=drumplace;
      drumplace:=pl111;
      from drum(B);
      drumplace:=tr;
      x:=x0;
      xx:=x0/deltax-xxx;
      yy:=y/deltay-yyy;
      gierproc(B[2],hop,xx,yy);
      xxx:=xxx+xx;
      yyy:=yyy+yy;
      for x:=x0 step dx until x1 do
      begin
         xx:=x/deltax-xxx;
         yy:=y/deltay-yyy;
         gier(hop);
comment
```

```
;
         xxx:=xxx+xx;
         yyy:=yyy+yy
      end
   end plotgraph;

   procedure plotline(x0,y0,x1,y1);
   value x0,y0,x1,y1;
   real x0,y0,x1,y1;
   begin
      integer xx,yy,tr;
      boolean hop;
      boolean array B[1:l111];
      tr:=drumplace;
      drumplace:=pl111;
      from drum(B);
      drumplace:=tr;
      xx:=x0/deltax-xxx;
      yy:=y0/deltay-yyy;
      gierproc(B[2],hop,xx,yy);
      xxx:=xxx+xx;
      yyy:=yyy+yy;
      xx:=x1/deltax-xxx;
      yy:=y1/deltay-yyy;
      gier(hop);
      xxx:=xxx+xx;
      yyy:=yyy+yy
   end plotline;


   real procedure plotchar(t,x,y,h);
   value t,x,y,h;
   integer t;
   real x,y,h;
   begin
      integer xx,yy,n1,n2,hh,i,x1,y1;
      boolean hop,AA,flyt;
      boolean array A[0:1],B[1:l111];
      if t≥58 then goto exit;
      hh:=h×20;
      xx:=drumplace;
      drumplace:=pl111;
      from drum(B);
      drumplace:=plabc-(57-t)×2;
      from drum(A);
      drumplace:=xx;
      AA:=A[0];
      x1:=split(AA,0,4,n1,5,9,y1):6;
      y1:=(y1-x1×6)×hh;
      x1:=x1×hh;
      xx:=x1+x/deltax-xxx;
      yy:=y1+y/deltay-yyy;
      xxx:=x/deltax;
      yyy:=y/deltay;
      if kbon then write(⊰-nddd⊱,writecr,xx,yy);
      gierproc(B[2],hop,xx,yy);
      n2:=(n1-8)×5;
      n1:=if n1≤7 then n1×5 else 35;
      flyt:=false;
      for i:=10 step 5 until n1,0 step 5 until n2 do
      begin
         if i=0 then AA:=A[1];
comment
```

```
;
         xx:=split(AA,i,i+4,yy):6;
         if yy=31 then
         begin
            flyt:=true;
         end
         else
         begin
            yy:=(yy-xx×6)×hh-y1;
            xx:=xx×hh-x1;
            if kbon then write(⊰-nddd⊱,writecr,xx,yy);
            if flyt then
            begin
               flyt:=false;
               gierproc(B[2],hop,xx,yy);
            end
            else
            gier(hop);
            x1:=x1+xx;
            y1:=y1+yy
         end
      end i;
      xxx:=xxx+x1;
      yyy:=yyy+y1;
exit: plotchar:=if t<58 then x+100×h×deltax else x
   end plotchar;
   xxx:=yyy:=0;
   deltax:=deltay:=0.01;
   pl111:=drumplace;
   gierdrum(111,l111);
   plabc:=drumplace;
   gierdrum(113,0);
```

comment

```
;
   comment

Program af 120365 til inversion af Laplace transform = f(t)
arbejder med 3 slags data:
      1)  syntetiske data af form f(t)=Sum[N[i]×exp(-lambda[i]×t)]
som det selv genererer ved hjælp af real procedure f(x). I dette
tilfælde skrives o på skrivemaskinen inden kompileringen, for at
inkludere procedurekrop svarende til denne type data. pause-
signalet ignoreres.
      2)  syntetiske data, som ikke er en sum af eksponential
funktioner, men hvor f(t)=f(exp(x)) kan udtrykkes på explicit form.
I dette tilfælde skrives der ikke o på skrivemaskinen, men når
pause-signalet kommer, tastes procedurekroppen til real procedure
f(x) ind fra skrivemaskinen. Eks:
          f:= exp(-0.001×exp(x))/(exp(x)+0.009)↑2.
      3)  experimentelle data i form af et array af funktionsværdier
aflæst for ækvidistante værdier af lnt=x. I dette tilfælde tastes o
før compileringen. Ved at gøre n negativ (se nedenfor) vil maskinen
være parat til at indlæse en strimmel indeholdende data-array F[n,-n].
pause-signalet ignoreres.
DATASTRIMMEL I: maskinprocedure plot 111, 112, 113.
DATASTRIMMEL II:bestaar af tabel over den komplekse funktion,
Gamma(1+iu) hvor (u=0(0.1)10).
INPUT FRA SKRIVEMASKINE:
dx     skridtlængde ved x-integration  (f.eks. 0.25)
du     skridtlængde ved u-integration, skal være mult af 0.1
xmax   øvre grænse ved x-integrationen, tastes evt først ind , efter
       at plotteren har tegnet grafen af integranden ved x-integration
       skal være lige mult af dx
umax   øvre grænse ved u-integration, skal være lige mult af du
lamax  hhv øvre og nedre grænse for det interval for lamda, hvori
lamin  man ønsker at undersøge g(lamda)/lamda. lamax og lamin skal
       hele mult af 10-i
noofy  antal plottepunkter i det angivne lamda-interval
n, n>0 antal exponentialled i den syntetiske kurve f(t) af typen 1)
       med parametrene
N[i], lambda[i]   (i=1,...,n)
n, n<0 i dette tilfælde indlæses strimmel med rigtige data
F[n,-n] som er den experimentelle kurve aflæst med ækvidistancen dx
       i x-intervallet [n×dx,-n×dx].
;

   begin
      integer n,m,noofy,Kplads;
      boolean nykurve;
      real dx,xmax,du,umax,lamax,lamin;
      real Kc,Ks;
      procedure Kcs(index);
      value index;
      integer index;
      begin
      real PI,PI2,LNPI,LN2,LogRootTwoPi;
      integer procedure lngamma complex(zr,zi,lnr,arg);
      value zr,zi;
      real zr,zi,lnr,arg;
      begin
         real x,y,a,b,lnsin r,lnsin i;
         integer stat l,stat s;


comment
```

```
;
         if zr ≤ 0.5 then
         begin
            x:=1-zr;
            y:=-zi;
            stat l:=lngamma lanczos complex(x,y,a,b);
            stat s:=complex logsin(zr×PI,zi×PI,lnsin r,lnsin i);
            if stat s=0 then
            begin
               integer stat r;
               lnr:=LNPI-lnsin r-a;
               arg:=-lnsin i-b;
               stat r:=angle restrict symm(arg);
               lngamma complex:=stat r;
               if stat l≠0 then lngamma complex:=stat l;
            end
         end
         else
         lngamma complex:=lngamma lanczos complex(zr,zi,lnr,arg)
      end lngamma complex;
      integer procedure lngamma lanczos complex(zr,zi,yr,yi);
      value zr,zi;
      real zr,zi,yr,yi;
      begin
         integer k;
         real log1 r,log1 i,logAg r,logAg i,Ag r,Ag i,yi tmp,R,I,a,b;
         zr:=zr-1;
         Ag r:=1;
         Ag i:=0;
         k:=1;
         for b:= 676.520368, -1259.13922, 771.323429, -176.615029,
                  12.5073433, -0.138571095, 9.98436958₁₀-6, 1.50563274₁₀-7 do
         begin
            R:=zr+k;
            I:=zi;
            a:=b/(R×R+I×I);
            Ag r:=Ag r+a×R;
            Ag i:=Ag i-a×I;
            k:=k+1
         end b;
         complex log(zr+7.5,zi,log1 r,log1 i);
         complex log(Ag r,Ag i,logAg r,logAg i);
         yr:=(zr+0.5)×log1 r-zi×log1 i-(zr+7.5)+LogRootTwoPi+logAg r;
         yi:=zi×log1 r+(zr+0.5)×log1 i-zi+logAg i;
         angle restrict symm(yi);
         lngamma lanczos complex:=0
      end lngamma lanczos complex;
      integer procedure angle restrict symm(theta);
      real theta;
      begin
         theta:=theta-PI2×entier(theta/PI2);
         if theta>PI then theta:=theta-PI2;
         if theta<-PI then theta:=theta+PI2;
         angle restrict symm:=0
      end angle restrict symm;
      integer procedure complex logsin(zr,zi,lszr,lszi);
      value zr,zi;
      real zr,zi,lszr,lszi;
      begin


comment
```

```
;
         if zi>60 then
         begin
            lszr:=-LN2+zi;
            lszi:=0.5×PI-zr
         end
         else if zi<-60 then
         begin
            lszr:=-LN2-zi;
            lszi:=-0.5×PI+zr
         end
         else
         begin
            real sin r,sin i;
            complex sin(zr,zi,sin r,sin i);
            complex logsin:=complex log(sin r,sin i,lszr,lszi);
         end;
         angle restrict symm(lszi)
      end complex logsin;
      real procedure sinh series(x);
      value x;
      real x;
      begin
         real y;
         y:=x×x;
         sinh series:=x×(1+y×((1/6)+y×((1/120)+y×((1/5040)+y×((1/362880)+
            y×((1/39916800)+y×((1/6227020800)+y×((1/130767437_{10}4)+
            y×(1/355687428_{10}6))))))))
      end sinh series;
      real procedure cosh m1 series(x);
      value x;
      real x;
      begin
         real y;
         y:=x×x;
         cosh m1 series:=y×((1/2)+y×((1/24)+y×((1/720)+y×((1/40320)+y×((1/3628800)+
                         y×((1/479001600)+y×((1/8717829120)+y×((1/209227899_{10}5)+
                         y×(1/640237371_{10}7))))))));
      end cosh m1 series;
      integer procedure complex sin(zr,zi,szr,szi);
      value zr,zi;
      real zr,zi,szr,szi;
      begin
         complex sin:=0;
         if abs(zi)<1 then
         begin
            real ch m1,sh;
            sh:=sinh series(zi);
            ch m1:=cosh m1 series(zi);
            szr:=sin(zr)×(ch m1+1);
            szi:=cos(zr)×sh;
         end
         else
         if abs(zi)<354.1982 then
         begin
            real ex,ch,sh;
            ex:=exp(zi);
            ch:=0.5×(ex+1/ex);
            sh:=0.5×(ex-1/ex);


comment
```

```
;
            szr:=sin(zr)×ch;
            szi:=cos(zr)×sh
         end
         else
         complex sin:=1
      end complex sin;
      integer procedure complex exp(zr,zi,exp r,exp i);
      value zr,zi;
      real zr,zi,exp r,exp i;
      begin
         real rho,theta;
         rho:=exp(zr);
         exp r:=rho×cos(zi);
         exp i:=rho×sin(zi);
         complex exp:=0
      end complex exp;
      real procedure complex logabs(ar,ai);
      value ar,ai;
      real ar,ai;
      begin
         real xabs,yabs,max,u;
         xabs := abs(ar);
         yabs := abs(ai);
         if xabs≥yabs then
         begin
            max:=xabs;
            u:=yabs/xabs
         end
         else
         begin
            max:=yabs;
            u:=xabs/yabs
         end;
         complex logabs:=ln(max)+0.5×ln(1+u↑2)
      end complex logabs;
      real procedure atan2(y,x);
      value y,x;
      real y,x;
      atan2:=if x>0 then arctan(y/x) else
             if x<0 ∧ y≥0 then arctan(y/x)+PI else
             if x<0 ∧ y<0 then arctan(y/x)-PI else
             if x=0 ∧ y>0 then PI/2 else
             if x=0 ∧ y<0 then -PI/2 else
             0;
      real procedure complex arg(zr,zi);
      value zr,zi;
      real zr,zi;
      begin
         if zr=0 ∧ zi=0 then
         complex arg:=0
         else
         complex arg:=atan2(zi,zr)
      end;
      integer procedure complex log(zr,zi,log r,log i);
      value zr,zi;
      real zr,zi,log r,log i;
      begin
         if zr≠0 ∨ zi≠0 then
         begin
            real ax,ay,min,max;

comment
```

```
;
            ax := abs(zr);
            ay := abs(zi);
            min := ax;
            max := ax;
            if ay<min then min:=ay;
            if ay>max then max:=ay;
            log r:=ln(max)+0.5×ln(1+(min/max)↑2);
            log i:=atan2(zi,zr);
            complex log:=0
         end
         else complex log:=1
      end complex log;
      real zr,zi,lnr,arg,Zr,Zi,table r,table i;
      integer i;
      PI:=3.141592658;
      PI2:=PI×2;
      LNPI:=ln(PI);
      LN2:=ln(2);
      LogRootTwoPi:=ln(sqrt(PI2));
      zr:=1;
      zi:=index/10;
      lngamma complex(zr,zi,lnr,arg);
      complex exp(lnr,arg,Kc,Ks);
   end Kcs;

   procedure plotcond(x,y,t,e1,e2,b1,b2);
   real x,y,t,e1,e2;
   boolean b1,b2;
   begin
      integer tr,xx,yy;
      boolean hop,s;
      boolean array B[1:l111];
      tr:= drumplace;
      drumplace:= pl111;
      from drum(B);
      drumplace:= tr;
      s:= true;
      for t:= e1,e2 while b1 do
      if -, b2 then
      s:= true
      else
      begin
         xx:= x/deltax - xxx;
         yy:= y/deltay - yyy;
         if s then
         begin
            gierproc(B[2],hop,xx,yy);
            xxx:= xxx+xx;
            yyy:= yyy+yy;
            s:= false
         end
         else
         begin
            gier(hop);
            xxx:= xxx+xx;
            yyy:= yyy+yy
         end
      end
   end plotcond;


comment
```

```
;
   n:= 0;
DATA:
   writecr;
   writetext(⊰<dx,xmax,du,umax,lamax,lamin,noofy:
⊱);
   outcr;
   outtext(⊰<program 120365 - inversion af Laplace transform⊱);
   dx:= typein;
   xmax:= typein;
   du:= typein;
   umax:= typein;
   lamax:= typein;
   lamin:= typein;
   noofy:= typein;
   m:= 1;
   writecr;
   writetext(⊰<m:⊱);
   m:= typein;
   if m=0 then goto NÆSTYDERSTE BLOK;
   if m<0 then
   begin
      n:= -m;
      nykurve:= true;
      goto NÆSTYDERSTE BLOK
   end;
   n:= m;
   writecr;
   writetext(⊰<N(i), lambda(i): ⊱);
   nykurve:= true;

NÆSTYDERSTE BLOK:
   begin
      integer i,j,k,uo,us,K,S,grænse;
      real x,u,y,ymax,ymin,dy,gy,pi,Fcu,Fsu,Kcu,Ksu,nævner,lamda,lstep,gmax;
      array N,lambda[1:n],g[0:noofy],F[-n:n];
      real procedure f(x);
      value x;
      real x;
begin
         real fx;
         fx:= 0;
         for i:=1 step 1 until n do fx:= fx + N[i]×exp(-lambda[i]×exp(x));
         f:= fx;
      end f;*;
      if m<0 then
      begin
         input(F);
         outcr;
         outtext(⊰<real data⊱);
         goto XINT
      end;
      if m>0 then
      for i:=1 step 1 until n do
      begin
         N[i]:= typein;
         lambda[i]:= typein
      end;
      outcr;
      outcr;
      outtext(⊰<N(i), lambda(i): ⊱);

comment
```

```
;
      for i:=1 step 1 until n do output(⊂-ndddd.ddd⊃,outcr,N[i],lambda[i]);
      if m=0 then goto XINT;

      begin
         real DX;
         comment i denne blok bliver log y, hvor y=exp(x)×f(x), plottet mod x
(x=-10,-9,...,10) i skridt af o.25. x-enheden=0.25 tomme. y-dekaden
=1.5 tomme. Der er 5 dekader;
         deltax:= 0.04;
         deltay:= 0.02/3×2.302585;
         comment faktoren 2.302585 i deltay er lig ln 10. Dvs kan ved kald af
plotprocedurer benytte ln i stedet for log10;
         writetext(⊂<
GØR CALCOMP KLAR t×f(t) plottes mod ln(t) - hvis dette uønsket tast 0: ⊃);
         j:= typein;
         if j=0 then goto XINT;
         x:= -10;
         DX:= 0.4;
         xxx:= yyy:= 0;
YAXE:
         for j:=1,10,100,1000,10000 do
         for k:=2 step 1 until 10 do
         begin
            plotline(x,ln((k-1)×j),x,ln(k×j));
            plotline(x,ln(k×j),x+DX,ln(k×j));
         end k;
         if x=10 then goto XAXE;
         for k:=6 step -1 until 1 do
         begin
            x:= -10-k×0.7;
            j:= 1;
            for j:=j while x<-10.6 do
            begin
               x:= plotchar(j,x,ln(10↑(k-1)),0.175);
               j:= 16
            end;
         end k;
         x:= 10;
         DX:= -0.4;
         goto YAXE;
XAXE:
         deltay:= deltay/2.302585×1.5;
         comment y-enhed = 1 tomme;
         for j:=8 step -2 until -10 do
         begin
            plotline(j+2,0,j,0);
            plotline(j,0,j,-0.1);
            if j=0 then
            begin
               plotchar(16,-0.24,-0.4,0.15);
               x:= plotchar(23,-1.5,-0.9,0.15);
               plotline(x+0.16,-0.8,x+0.44,-0.8);
               plotline(x+0.16,-0.85,x+0.44,-0.85);
               x:= x+0.6;
               for i:=35,37,19 do x:= plotchar(i,x,-0.9,0.15);
            end
            else
            if j>0 then plotchar(j,j-0.24,-0.4,0.15)
            else
            if j≠-10 then
            begin
comment
```

```
;
               plotline(j-0.8,-0.325,j-0.55,-0.325);
               plotchar(-j,j-0.24,-0.4,0.15)
            end;
         end;
         x:= -8;
         x:= plotchar(53,x,7,0.15);
         x:= plotchar(23,x,7.1,0.1);
         for j:=54,53 do x:= plotchar(j,x,7,0.15);
         plotchar(23,x,7.1,0.1);
         deltay:= deltay×2.302585/1.5;
         comment y-dekade = 1.5 tomme;
         plotcond(x,x+ln(f(x)),x,-10,x+0.15,x≤10,x+ln(f(x))>ln(0.5));
         writecr;
         writetext(⊰<xmax (lige mult af dx): ⊱);
         xmax:= typein;
      end paa blok med plotning af exp(x)×f(x);

XINT: grænse:= entier(xmax/dx);
      j:= entier(umax×10);
      begin
         real k1,k2,k3,k4,k5;
         array Fc,Fs[0:j];
         integer u1,du1,umax1;
         begin
            real plus,minus,Fcos,Fsin;
            array fc,fs[0:grænse];
            comment i denne blok foretages x-integrationen ved hjælp af
Simpson. Dvs Fc(u) og Fs(u), hhv den reelle og imaginære del
af F(u) beregnes;
            for k:=0 step 1 until grænse do
            begin
               x:= k×dx;
               if kbon then write(⊰nddd⊱,writecr,
                                 writetext(⊰<k,grænse I: ⊱),k,grænse);
               if m<0 then
               begin
                  plus:= exp(x)×F[k];
                  minus:= exp(-x)×F[-k]
               end
               else
               begin
                  plus:= exp(x)×f(x);
                  minus:= exp(-x)×f(-x)
               end;
               fc[k]:= plus+minus;
               fs[k]:= plus-minus;
            end k;
            du1:= du×10;
            umax1:= umax×10;
            if kbon then write(⊰nddd.dd⊱,writecr,writetext(⊰<du,du1: ⊱),du,du1);
            for u1:=0 step du1 until umax1 do
            begin
               u:= u1/10;
               if kbon then writetext(writecr,⊰<kilroy 1⊱);
               Fcos:= fc[0]+4×fc[1]×cos(u×dx)+fc[grænse]×cos(u×xmax);
               Fsin:= 4×fs[1]×sin(u×dx)+fs[grænse]×sin(u×xmax);
               for k:=2 step  2 until grænse-2 do
               begin
                  if kbon then write(⊰nddd⊱,writecr,writetext(⊰<k,grænse II: ⊱),
                                                k,grænse);

comment
```

```
;
                  Fcos:= Fcos + 2×fc[k]×cos(u×k×dx) + 4×fc[k+1]×cos(u×(k+1)×dx);
                  Fsin:= Fsin + 2×fs[k]×sin(u×k×dx) + 4×fs[k+1]×sin(u×(k+1)×dx);
               end k;
               if kbon then write(⦃nddd⦄,writecr,writetext(⦃<u1,du1,umax1: ⦄),
                                          u1,du1,umax1);
               Fc[u1] := Fcos×dx/3;
               if kbon then writetext(writecr,⦃<kilroy 2⦄);
               Fs[u1] := Fsin×dx/3;
               if kbon then  writetext(writecr,⦃<kilroy 3⦄);
            end u;
         end x-integration;

         comment nu kendes Fc,Fs,Kc,Ks og u-integrationen kan
foretages (Simpson);
         clock(false);
         uo:= entier(umax/0.1);
         us:= du/0.1;
         if kbon then write(⦃ndddd⦄,writecr,writetext(⦃<uo,us: ⦄),uo,us);
         ymax:= -ln(lamin);
         ymin:= -ln(lamax);
         dy:= (ymax-ymin)/noofy;
         i:= 0;
         Kcs(0);
         k1:= (Fc[0]×Kc+Fs[0]×Ks)/(Kc↑2+Ks↑2);
         Kcs(us);
         k2:= 4×(Fc[us]×Kc+Fs[us]×Ks)/(Kc↑2+Ks↑2);
         k3:= 4×(Fs[us]×Kc-Fc[us]×Ks)/(Kc↑2+Ks↑2);
         Kcs(uo);
         k4:= (Fc[uo]×Kc+Fs[uo]×Ks)/(Kc↑2+Ks↑2);
         k5:= (Fs[uo]×Kc-Fc[uo]×Ks)/(Kc↑2+Ks↑2);
         for y:=ymin step dy until ymax do
         begin
            gy:= k1+k2×cos(y×du)+k3×sin(y×du)+k4×cos(y×umax)+k5×sin(y×umax);
            for k:=2×us step 2×us until uo-2 do
            begin
               K:= k;
               S:= 2;
OM:            Fcu:= Fc[K];
               Fsu:= Fs[K];
               Kcs(K);
               Kcu:= Kc;
               Ksu:= Ks;
               nævner:= Kcu↑2 + Ksu↑2;
               u:= K×0.1;
               gy:= gy+S×(Fcu×Kcu+Fsu×Ksu)/nævner×cos(y×u)
                      +S×(Fsu×Kcu-Fcu×Ksu)/nævner×sin(y×u);
               K:=K + us;
               S:= 4;
               if K= k+us then goto OM;
            end k;
            g[i]:= gy;
            i:= i+1;
         end y og u-integration;
      end integrationer;

      clock(true);
      comment nu udskrives og plottes resultaterne;
      outcr;
      outcr;
      outtext(⦃<dx,xmax,du,umax,lamax,lamin,noofy: ⦄);
      outcr;
comment
```

```
;
      output(⊰-nddd.dddd⊱, dx,xmax,du,umax,lamax,lamin,noofy);
      outcr;
      outcr;
      outtext(⊰<y,lambda,g(exp(-y)): ⊱);
      for i:=0 step 1 until noofy do
      begin
         y:= ymin + i×dy;
         lamda:= exp(-y);
         outcr;
         output(⊰-ndd.dd⊱, y);
         output(⊰-nddd.dddd⊱, lamda);
         output(⊰-ndd.d10+d⊱, outsp(4), g[i]);
      end i;
      gmax:= g[0];
      for i:= 1 step 1 until noofy do if gmax<g[i] then gmax:= g[i];
      outcr;
      outcr;
      outtext(⊰<100: ⊱);
      output(⊰ndd10d⊱, gmax);
      for j:=0 step 1 until noofy do g[j]:= g[j]/gmax×100;
      writecr;
      writetext(⊰<gør Calcomp klar til ny plotning⊱);
      j:= typein;
      deltax:= 0.01×(ymax-ymin)/5;
      deltay:= 0.01×100/5;
      comment y-akses længde = 9 tommer;
      xxx:= ymin;
      yyy:= 0;
      for i:= 10 step 10 until 140 do
      begin
         plotline(ymin,i-10,ymin,i);
         plotline(ymin,i,ymin+10×deltax,i);
         if i=50 then
         begin
            x:= ymin-36×deltax;
            x:= plotchar(5,x,50,0.175);
            plotchar(16,x,50,0.175)
         end;
         if i=100 then
         begin
            x:= ymin-54×deltax;
            x:= plotchar(1,x,100,0.175);
            x:= plotchar(16,x,100,0.175);
            plotchar(16,x,100,0.175);
         end i=100;
      end i;
      for i:= -10 step -10 until -40 do
      begin
         plotline(ymin,i+10,ymin,i);
         plotline(ymin,i,ymin+10×deltax,i)
      end;
      plotline(ymin,0,entier(ymin+1),0);
      plotline(entier(ymin+1),0,entier(ymin+1),-2);
      i:= 1;
      for i:=i+1 while entier(ymin+i)≤ymax do
      begin
         y:= entier(ymin+i);
         plotline(y-1,0,y,0);
         plotline(y,0,y,-2)
      end;

comment
```

```
;
      plotline(entier(ymin+i-1),0,ymax,0);
      plotline(ymax,-40,ymax-10×deltax,-40);
      for i:=-30 step 10 until 140 do
      begin
         plotline(ymax,i-10,ymax,i);
         plotline(ymax,i,ymax-10×deltax,i) end;
         for i:=1 step 1 until noofy do
         plotline(ymin+(i-1)×dy,g[i-1],ymin+i×dy,g[i]);
         i:=10;
         lstep:= 1;
         for i:=i×10 while entier(i×lamax)=0 do lstep:= 1/i/10;
         if kbon then output(⟨-n.dd10+d⟩,outcr,outtext(⟨<lstep: ⟩),lstep);
         i:= 0;
         y:= ymin;
         lamda:= lamax;
         for i:= i+1 while y<ymax do
         begin
            if kbon then output(⟨nddd.dd⟩,outcr,outtext(⟨<i: ⟩),i,
                                 outtext(⟨< y: ⟩),y);
            for k:= -1 step 1 until 4 do
            if abs(lamda-10↑(-k))<10-7 then
            begin
               if kbon then output(⟨-ndd⟩,outcr,outtext(⟨<k: ⟩),k);
               if k≤0 then
               begin
                  x:=plotchar(1,y-k×8×deltax,-49,0.15);
                  for i:=k step 1 until -1 do x:=plotchar(16,x,-49,0.15);
               end
               else
               begin
                  x:= y-k×8×deltax;
                  x:= plotchar(16,x,-49,0.03);
                  for i:=2 step 1 until k do x:= plotchar(16,x,-49,0.15);
                  plotchar(1,x,-49,0.15)
               end
            end;
            lamda:= lamda - lstep;
            if lamda<10-7 then
            begin
               lamda:= lamda+lstep-lstep/10;
               lstep:= lstep/10
            end;
            y:= -ln(lamda);
            if y>ymax then
            begin
               plotline(-ln(lamda+lstep),-40,ymax,-40);
               goto l1
            end;
            plotline(-ln(lamda+lstep),-40,y,-40);
            plotline(y,-40,y,-42);
l1:      end lamdaakse 1;
         i:= 10;
         lstep:= 1;
         for i:=i×10 while entier(i×lamax)=0 do lstep:=1/i/10;
         i:= 0;
         y:= ymin;
         lamda:= lamax;
         for i:=i+1 while y<ymax do
         begin
            lamda:= lamda - lstep;

comment
```

```
;
            if lamda<10-7 then
            begin
               lamda:= lamda+lstep-lstep/10;
               lstep:= lstep/10
            end;
            y:= -ln(lamda);
            if y>ymax then
            begin
               plotline(-ln(lamda+lstep),140,ymax,140);
               goto l2
            end;
            plotline(-ln(lamda+lstep),140,y,140);
            plotline(y,140,y,138);
l2:       end lamdaakse 2;
      end næstyderste blok;
      nykurve:= false;
      goto DATA;
   end yderste blok
end som skal svare til plotprograms begin;
```