Radioaktivt sönderfall DN1241,43, Fö 8

Ekvationerna som beskriver hur ett radioaktivt ämne A sönderfaller till ämnet B som i sin tur sönderfaller till C

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$

ges av

$$\begin{array}{lll} dx_1/dt & = -k_1 x_1 & x_1(0) = 100 \\ dx_2/dt & = k_1 x_1 - k_2 x_2 & x_2(0) = 0 \\ dx_3/dt & = k_2 x_2 & x_3(0) = 0 \end{array}$$

där x_1 står för mängden av ämne A, x_2 mängden av ämne B och x_3 mängden av ämne C. k_1 och k_2 är tidskonstanter för sönderfallet. Från början finns 100 kg av A och ingenting av de två övriga ämnena.

Detta är ett linjärt system av ordinära differentialekvationer, av det slag som ni bör ha sett i matematikkurserna och som kan lösas exakt. Vi skall emellertid lösa det numeriskt, och för att göra det ännu enklare för oss börjar vi med att numeriskt lösa ekvationen

$$\frac{dx}{dt} = -kx \qquad x(0) = 100$$

d.v.s. den första av ekvationerna ovan.

Vi använder MATLAB rutinen ODE23 och behöver då en egendefinierad funktion som definierar differentialekvationen. Vi skapar filen **fradiak1.m** enligt ($k = 10$ har valts)

```
function dxdt=fradiak1(t,x)
dxdt=-10*x;
```

Funktionen måste ha två parametrar:

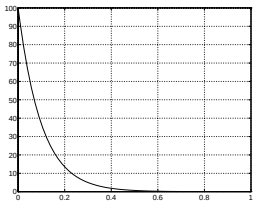
–den första är den oberoende variabeln, i vårt fall tiden t.

–den andra är den beroende variabeln, i vårt fall $x(t)$.

Vi integerar från $t = 0$ till $t = 1$ med begynnelsevärdet $x(0) = 100$

```
[tout xout]=ode23(@fradiak1,[0 1],100);
plot(tout,xout)
grid
```

och får vidstående plotbild.



För att lösa de tre kopplade ekvationerna med hjälp av ODE23 skapar vi filen **fradiak2.m** enligt ($k_1 = 10$ och $k_2 = 1$ har valts)

```
function z=fradiak2(t,x)
%z=(dx1dt dx2dt dx3dt)
k1=10; k2=1;
dx1dt=-k1*x(1);
dx2dt=k1*x(1)-k2*x(2);
dx3dt=k2*x(2);
z=[dx1dt;dx2dt;dx3dt];
```

Funktionen måste som alltid ha två parametrar:

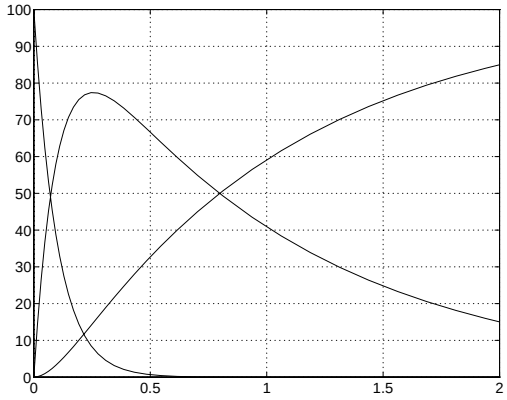
– den första är den oberoende variabeln, i vårt fall tiden t.

– den andra är den beroende variabeln, i vårt fall vektorn $x(t)$.

Vi integerar från $t = 0$ till $t = 2$ med begynnelsevektorn $x = (100,\ 0,\ 0)^T$.

```
x=[100;0;0];
[tout xout]=ode23(@fradiak2,[0 2],x);
plot(tout,xout)
grid
```

och får nedanstående plotbild. De tre komponenterna av lösningen har plottats som funktion av tiden. Vilken av kurvorna motsvarar x_2?



Glödlampan

När man släcker bilstrålkastarna ser man hur glödtråden svalnar och svartnar, men uppvärmningen tycks gå mycket fortare. Kan vi illustrera det med en matematisk modell för hur en glödtråd värms upp av strömmen och kyls när strömmen slås av? Antag att tråden är tunn så att temperaturen är densamma över hela tvärsnittet, att den värms av den Ohmska värmeutvecklingen och kyls av strålning enligt Boltzmann's T^4 lag.

Då gäller med lämpligt val av skalor för tiden t och temperaturen y, att

$$\frac{dy}{dt} = q - (y^4 - 1)$$

Strömmen slås på vid $t = 0$, då $y = 1$, och slås av vid $t = 0.003$ så

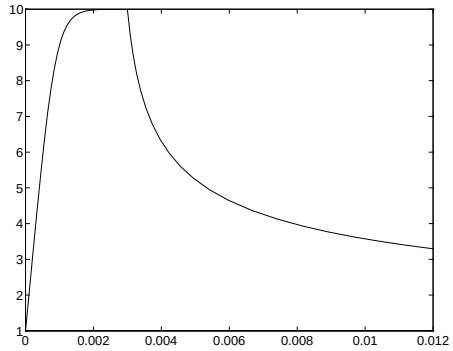
$$q = \begin{cases} 10000 & t < 0.003 \\ 0 & t > 0.003 \end{cases}$$

Lös begynnelsevärdesproblemet med MATLAB-rutinen **ode23** och plotta $y(t)$ så man ser hur tråden svalnar. Vi gör en fil **rad.m** som definierar högerledet i differentialekvationen. Syntaxen hämtar vi från **help ode23** och **odedemo.m**.

```
function [dydt]= rad(t,y);
  % dy/dt = q - (y^4 - 1)
  if t < 0.003
     dydt = 10000 +1 - y^4;
  else
     dydt = 1 - y^4;
end;
```

Integrera från $t = 0$ till $t = 0.012$ (det krävs några försök att se hur långt man ska integrera) med begynnelsevärdet $y(0) = 1$:

```
[tout,yout]= ode23(@rad,[0 0.012],1);
plot(tout,yout)
```

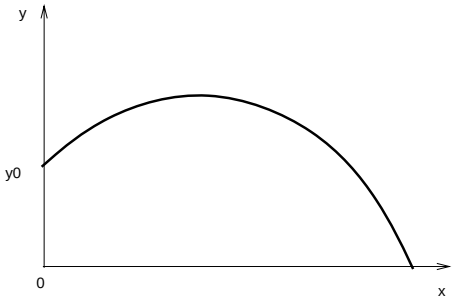


Vi ser här att temperaturen går upp till 90% av sitt slutvärde 10 på ungefär 0.001. När strömmen slås av sjunker temperaturen först raskt, men sedan allt långsammare. Även 0.009 efter släckning är tråden fortfarande varmare än 3. Den observerade effekten syns direkt.

Lösningen ser helt korrekt ut och är det också. Men problemet är inte helt enkelt: högerledet är diskontinuerligt som funktion av t vid $t = 0.003$.

Varpakast

Ekvationerna som beskriver en kaströrelse, t.ex. ett varpakast, ges enligt



$$
\begin{array}{lll}
dx/dt & = u & x(0) = 0 \\
dy/dt & = v & y(0) = y0 \\
mdu/dt & = -Cuw & u(0) = u0 \\
mdv/dt & = -mg - Cvw & v(0) = v0
\end{array}
$$

Här är $w = \sqrt{u^2 + v^2}$ och $C, y0, u0, v0$ givna konstanter. Vidare är u hastigheten i x-led, v hastigheten i y-led, w hastighetens belopp, m massan, C luftmotståndskoefficienten och $g = 9.81$.

För att simulera varpakast definierar vi vektorn $\mathbf{y} = (\mathbf{x}, \mathbf{y}, \mathbf{u}, \mathbf{w})^{\mathbf{T}}$ och skriver differentialekvationssystemet på normalform

$$\mathbf{y}' = \mathbf{f}(\mathbf{t}, \mathbf{y}); \qquad \mathbf{y}(\mathbf{0}) = \mathbf{c}$$

enligt (de två sista ekvationerna har dividerats med m)

$$\begin{array}{lll} dx/dt & = u & x(0) = 0 \\ dy/dt & = v & y(0) = y0 \\ du/dt & = -Cuw/m & u(0) = u0 \\ dv/dt & = -g - Cvw/m & v(0) = v0 \end{array}$$

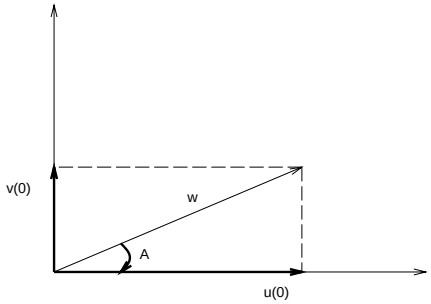
Följande MATLAB funktion beräknar derivatorna och lagrar dem i vektorn $\mathbf{z}$ där

$$\mathbf{z} = (\frac{\mathbf{dx}}{\mathbf{dt}}, \frac{\mathbf{dy}}{\mathbf{dt}}, \frac{\mathbf{du}}{\mathbf{dt}}, \frac{\mathbf{dv}}{\mathbf{dt}})^{\mathbf{T}}$$

```
function z=fvarpa(t,yvek)
  g=9.81; C=0.05; m=1;
  x=yvek(1); y=yvek(2); u=yvek(3); v=yvek(4);
  w=sqrt(u^2+v^2);
  z=[u
     v
     -C*u*w/m
     -g-C*v*w/m];
```

Funktionen har två parametrar `t,yvek` därför att vi planerar att använda ODE23 till simuleringen. Observera omlagringen från vektorn `yvek` till de beteckningar som används i den matematiska formuleringen. Detta görs enbart för att göra funktionen mer lättläst och lättkontrollerad.

Vi kan nu simulera varpakast och plottar $(x(t), y(t))$ med t som en osynlig parameter. För illustrationen studerar vi en varpa som väger $m = 1$ kg och har luftmotståndskoefficienten 0.05.



Utkasthastigheterna $u(0)$ och $v(0)$ specificerar vi lämpligen enligt figuren som en utkastvinkel A och hastighetens belopp w.

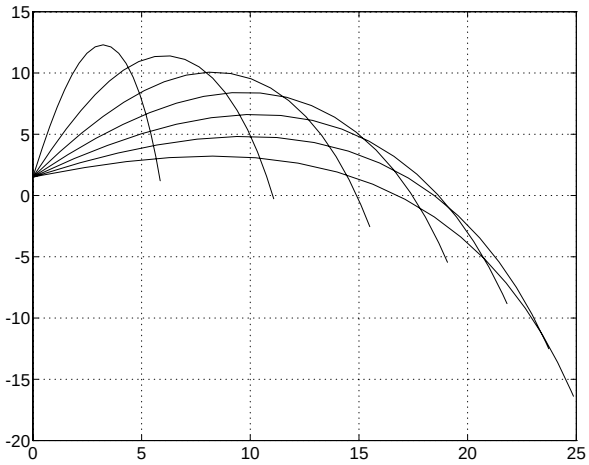
Vi simulerar sju st kast med olika utkastvinklar enligt programmet nedan. Initiala kasthöjden $y(0) = 1.5$ och utgångshastighetens belopp $w = 20$.

Vi låter varpan flyga under 3 sekunder i programmet **varpa.m**

```
clf; hold off;
for vinkel=20:10:80
```

```
  alpha=vinkel*pi/180;
  yvek=[0;1.5;20*cos(alpha);20*sin(alpha)];
  [tout,yout]=ode23(@fvarpa,[0 3],yvek);
  plot(yout(:,1),yout(:,2))
  if vinkel==20,
    hold on
  end
end
grid
```

De resulterande sju kastbanorna ser vi i figuren nedan



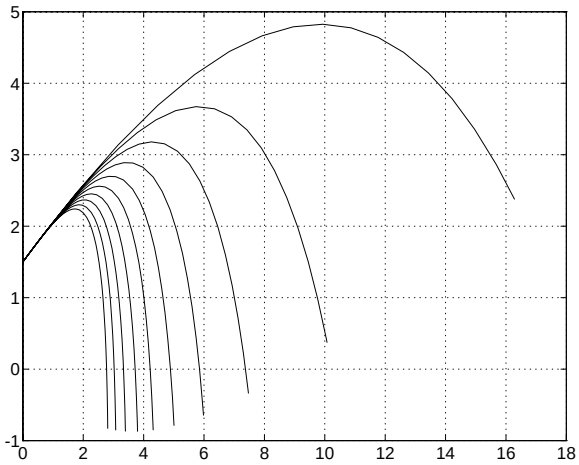
Vi är lite osäkra på hur stor luftmotståndskoefficienten C egentligen är och vill därför undersöka hur lösningen förändras om C ändras. För att göra detta på ett smidigt sätt definierar vi C som en global variabel. Vi definierar differentialekvationssystemet i funktionen **gvarpa.m** som inleds med

```
function z=gvarpa(t,yvek)
global CLUFTMOT
g=9.81; m=1; C=CLUFTMOT;
```

Resten av funktionen är identisk med **fvarpa.m**. I huvudprogrammet **globvarpa.m** väljer vi utkastvinkeln 30° och varierar C.

```
global CLUFTMOT     %globala variabler bör ha långa namn med stora bokstäver
clf; hold off;
vinkel=30; alpha=vinkel*pi/180;
for CLUFTMOT=0.05:0.1:1,
  yvek=[0;1.5;20*cos(alpha);20*sin(alpha)];
  [tout,yout]=ode23(@gvarpa,[0 1.5],yvek);
  plot(yout(:,1),yout(:,2))
  if CLUFTMOT==0.05,
    hold on
  end
end
grid
```

Resultatet blir



Vilken kastbana motsvarar det minsta värdet på C?