Ordinära differentialekvationer

Vi skall se på *begynnelsevärdesproblem* för första ordningens differentialekvation

$$\begin{cases} u' = f(t,u),\ a \le t \le b \\ u(a) = u_a \end{cases}$$

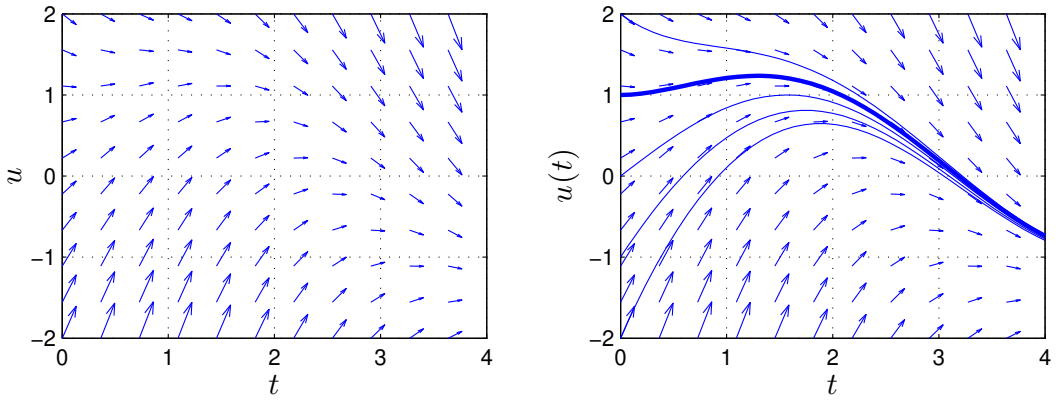
där f en given funktion och u_a en given konstant.

Som exempel tar vi problemet

$$\begin{cases} u' = -u(t) + \sin(t) + \cos(t),\ 0 \le t \le 4 \\ u(0) = u_0 \end{cases}$$

med analytisk (exakt) lösning $u(t) = \sin(t) + u_0 \mathrm{e}^{-t}$.

I den vänstra figuren nedan har vi ritat *riktningsfältet* och i den högra lösningskurvorna för några olika värden på u_0. Vi ser hur lösningskurvorna följer riktningsfältet.



Metoder för att beräkna numeriska (approximativa) lösningar till differentialekvationer bygger på idén att försöka följa riktningsfältet så noggrannt och effektivt som möjligt.

Vi skall approximera lösningen $u(t)$ till differentialekvationen på ett nät $t_n = a + nh$ för $n = 0, 1, \cdots, N$, med steglängden $h = \frac{b-a}{N}$.

Låter vi u_n beteckna approximationen av $u(t_n)$ och ersätter $u'(t_n)$ med en differenskvot så gäller

$$\frac{u(t_{n+1}) - u(t_n)}{h} \approx u'(t_n) = f(t_n, u(t_n)) \Rightarrow$$

$$u(t_{n+1}) \approx u(t_n) + hf(t_n, u(t_n))$$

Detta ger **Eulers metod**

$$u_{n+1} = u_n + hf(t_n, u_n)$$

Utgående från begynnelsevärdet försöker metoden följa riktningsfältet med korta steg.

Eget program i MATLAB

Vi skall nu beskriva hur man kan beräkna en numerisk lösning till begynnelsevärdesproblemet

$$\begin{cases} u' = f(t,u),\ a \le t \le b \\ u(a) = u_a \end{cases}$$

Vi bildar först ett nät med nodpunkterna $t_n = a + nh,\ n = 0, 1, \cdots, N$, och steglängden $h = \frac{b-a}{N}$. Detta ger en uppdelning av intervallet $a \le t \le b$ i N stycken lika långa delintervall

$$a = t_0 < t_1 < t_2 < \cdots < t_n < t_{n+1} < \cdots < t_{N-1} < t_N = b$$

Vi beräknar sedan en approximativ lösning enligt

$$\begin{aligned} U(t_0) &= u_a \\ U(t_{n+1}) &= U(t_n) + hf(t_n, U(t_n)). \end{aligned}$$

Genom att förbinda punkterna $(t_n, U(t_n))$ med räta linjer får vi en graf och funktionen $U(t)$ blir definierad också mellan beräkningsnoderna t_n.

I MATLAB måste $U(t_n)$ representeras av en vektor `U` med N komponenter. Med andra ord, `U(n)` skall innehålla approximationen av $u(t_n)$ för beräkningsnoden (tidpunkten) t_n.

Vi ser på vårt inledande exempel och tar $u(0) = 1$. Så här enkel blir MATLAB-koden

```
>> f=@(t,u)-u+sin(t)+cos(t);
>> a=0; b=4; ua=1;
>> N=10; h=(b-a)/N;
>> t=a+(0:N)*h; U=zeros(size(t));
>> U(1)=ua;
>> for n=1:N
       U(n+1)=U(n)+h*f(t(n),U(n));
   end
>> plot(t,U)
```

Uppgift 1. Lös följande differentialekvation med begynnelsevillkor

$$\begin{cases} u' = \cos(3t) - \sin(5t)u,\ 0 \le t \le 15 \\ u(0) = 2 \end{cases}$$

med Eulers metod. Tag $h = 0.1, h = 0.01$ och $h = 0.001$ som steglängder. Rita grafer av de olika lösningarna (med olika färger).

Uppgift 2. Skriv en `function` med namnet `min_ode` och anropet `[t,U]=min_ode(f,I,ua,h)` som löser begynnelsevärdesproblem. Du skall använda programskalet `min_ode.m` på kurshemsidan.

Uppgift 3. Testa din funktion på följande begynnelsevärdesproblem. Lös först problemen analytiskt (dvs. som en formel med penna och papper). Plotta både den analytiska lösningen u och den approximativa lösningen U i samma figur. Tag $h = 0.1$ och $h = 0.001$ som steglängder.

(a). $\begin{cases} u'(t) = t^2, \ t \in [1, 3], \\ u(1) = 1. \end{cases}$

(b). $\begin{cases} u'(t) = u(t), \ t \in [0, 2], \\ u(0) = 1. \end{cases}$

(c). $\begin{cases} u'(t) = -t\, u(t), \ t \in [0, 3], \\ u(0) = 1. \end{cases}$

(d). $\begin{cases} u'(t) = -5u(t), \ t \in [0, 1], \\ u(0) = 2. \end{cases}$

Färdiga program i MATLAB

Det finns många kommandon i MATLAB för att lösa differentialekvationer. Ett sådat är `ode45`. Med `ode45` kan vi beräkna en lösning till vårt inledande exempel för t.ex. $u(0) = 1$ enligt

```
>> [t,u]=ode45(@(t,u)(-u+sin(t)+cos(t)),[0 4],1);
>> plot(t,u)
```

Uppgift 4. Lös begynnelsevärdesproblemet i uppgift 1 med `ode45`. Rita upp lösningskurvan och rita även upp, i samma bild, lösningskurvorna beräknade med Eulers metod.

System av differentialekvationer

Som exempel på ett system av ekvationer tar vi Kermack-McKendricks differentialekvationer som beskriver utvecklingen av en viss epidemi. För $t \geq 0$ definerar man följande ekvationer:

$$\begin{cases} u_1'(t) = -au_1(t)u_2(t) \\ u_2'(t) = au_1(t)u_2(t) - bu_2(t) \\ u_3'(t) = bu_2(t) \end{cases}$$

Koefficienterna a, b är positiva. I ekvationerna står $u_1(t)$ för antalet som är mottagliga för sjukdomen, $u_2(t)$ antalet smittade och $u_3(t)$ antalet bortförda (ej längre smittsamma eller mottagliga). Ekvationerna i systemet beskriver hur antalet mottagliga, smittade och bortförda utvecklas med tiden.

Vår differentialekvation har formen

$$\begin{cases} \boldsymbol{u}' = \boldsymbol{f}(t, \boldsymbol{u}) \\ \boldsymbol{u}(0) = \boldsymbol{u}_0 \end{cases}$$

där

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}, \quad \boldsymbol{f}(t, \boldsymbol{u}) = \begin{bmatrix} -a\, u_1\, u_2 \\ a\, u_1\, u_2 - b\, u_2 \\ b\, u_2 \end{bmatrix}, \quad \boldsymbol{u}_0 = \begin{bmatrix} u_1(0) \\ u_2(0) \\ u_3(0) \end{bmatrix}$$

Detta är precis samma typ vi började med, fast nu har vi vektorer. Metoderna vi såg på då fungerar lika bra nu.

Nu bestämmer vi en lösning med MATLAB. Vi beskriver högerledet i differentialekvationen med en funktion

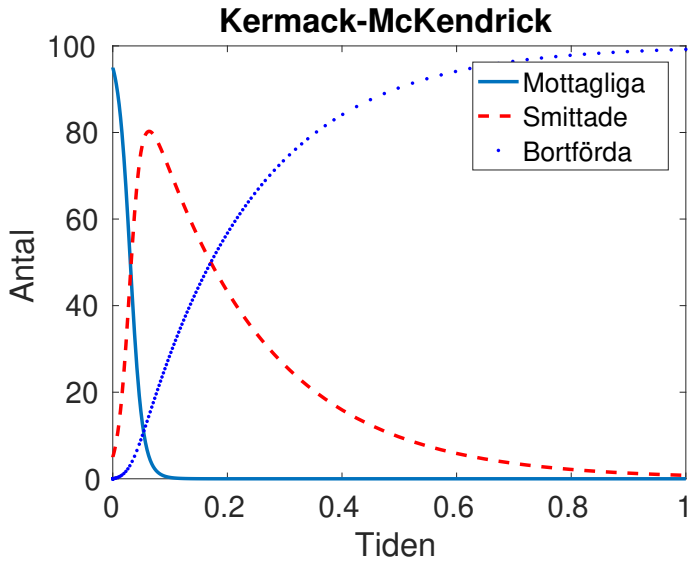
```
function f=kmk(t,u)
  a=1; b=5;
  f=[ -a*u(1)*u(2);
       a*u(1)*u(2)-b*u(2);
       b*u(2) ];
```

Vi startar med $u_1(0) = 95, u_2(0) = 5$ och $u_3(0) = 0$ och löser sedan med kommandot `ode45`.

```
>> u0=[95;5;0];
>> [t,U]=ode45(@kmk,[0 1],u0);
```

Vi ritar upp enligt:

```
>> figure(1), clf
>> plot(t,U(:,1),'-',t,U(:,2),'r- -',t,U(:,3),'b.')
>> legend('Mottagliga','Smittade','Bortförda')
>> xlabel('Tiden'), ylabel('Antal')
>> title('Kermack-McKendrick')
```



Uppgift 5. Lös Kermack-McKendricks differentialekvationer med `ode45` enligt ovan.

(a) Vad händer om man låter $u_2(0) = 0$?

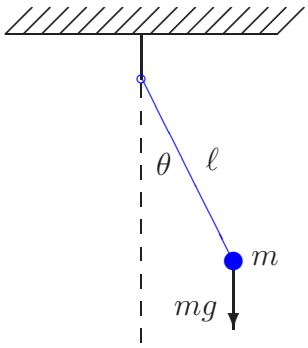
(b) Ändra faktorerna framför termerna (a och b) i ekvationerna så att sjukdomen blir mindre smittsam.

(c) Skriv en sekvens i Matlab som utifrån de värden som beräknas i exemplet bestämmer vid vilken tid antalet bortförda överstiger antalet smittade. (Ledning, andvänd t.ex. `find`)

Högre ordningens differentialekvationer

Högre ordningens differentialekvationer kan skrivas om som system av första ordningen. Dessa system kan sedan lösas numeriskt.

Som exempel tar vi den matematiska pendeln. En masspunkt med massan m hänger i en viktlös smal stav av längden ℓ.



Med beteckningarna i figuren och Newtons andra lag får vi rörelseekvationen

$$m\ell\,\ddot{\theta}(t) = -mg\sin(\theta(t))$$

Vi vill bestämma lösningen för olika begynnelseutslag θ_0, dvs. $\theta(0) = \theta_0$, då vi släpper pendeln från vila, dvs. $\dot{\theta}(0) = 0$.

Om vi låter $\varphi = \dot{\theta}$, dvs. inför vinkelhastigheten, kan ekvationen skrivas

$$\begin{cases} \dot{\theta} = \varphi, & \theta(0) = \theta_0 \\ \dot{\varphi} = -\frac{g}{\ell}\sin(\theta), & \varphi(0) = 0 \end{cases}$$

För att komma till standardform låter vi $u_1 = \theta$ och $u_2 = \varphi$ och får

$$\begin{cases} u_1' = u_2, & u_1(0) = \theta_0 \\ u_2' = -\frac{g}{\ell}\sin(u_1), & u_2(0) = 0 \end{cases}$$

Nu har vi standardformen

$$\begin{cases} \boldsymbol{u}' = \boldsymbol{f}(t,\boldsymbol{u}) \\ \boldsymbol{u}(0) = \boldsymbol{u}_0 \end{cases}, \quad \boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad \boldsymbol{f}(t,\boldsymbol{u}) = \begin{bmatrix} u_2 \\ -\frac{g}{\ell}\sin(u_1) \end{bmatrix}, \quad \boldsymbol{u}_0 = \begin{bmatrix} \theta_0 \\ 0 \end{bmatrix}$$

Vi beskriver differentialekvationen i MATLAB med funktionen

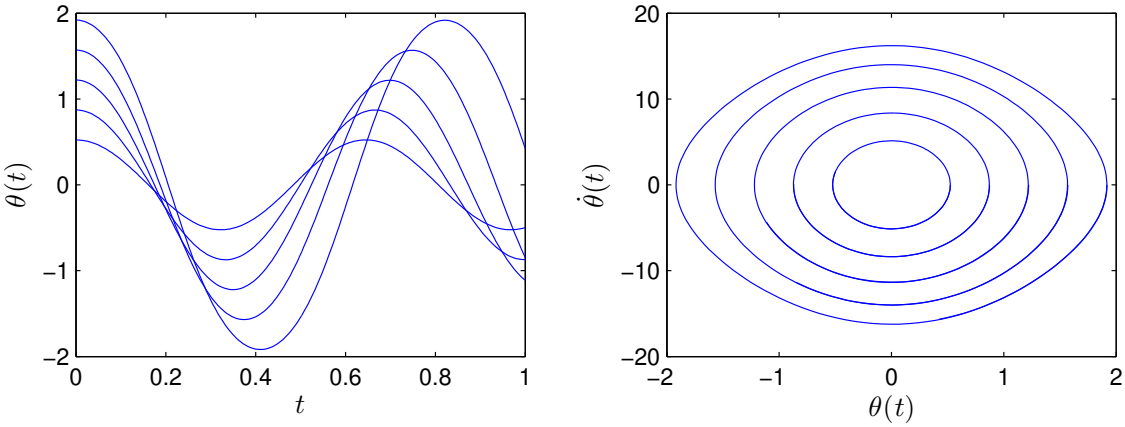
```
function f=pendel(t,u,g,l)
f=[u(2)
   -g/l*sin(u(1))];
```

Följer lösningskurvorna med `ode45` för några olika begynnelseutslag och ritar en bild som visar lösningarna $t \mapsto (t, \theta(t))$ och fasporträtten $t \mapsto (\theta(t), \dot{\theta}(t))$ för de olika begynnelseutslagen.

```
g=9.81; l=0.1; theta0=[30:20:110]*pi/180;
tspan=linspace(0,1,200);

for k=1:length(theta0)
    u0=[theta0(k);0];
    [t,U]=ode45(@(t,u)pendel(t,u,g,l),tspan,u0);
    subplot(1,2,1), plot(t,U(:,1)), hold on
    subplot(1,2,2), plot(U(:,1),U(:,2)), hold on
end
```

```
subplot(1,2,1), hold off
xlabel('$t$','interpreter','latex','fontsize',12)
ylabel('$\theta(t)$','interpreter','latex','fontsize',12),
subplot(1,2,2), hold off
xlabel('$\theta(t)$','interpreter','latex','fontsize',12)
ylabel('$\dot{\theta}(t)$','interpreter','latex','fontsize',12)
```



Från figuren ser vi att periodlängden ökar med ökande begynnelseutslag.

Uppgift 6. En dämpad matematisk pendel beskrivs av

$$\begin{cases} m\ell\,\ddot{\theta}(t) = -mg\sin(\theta(t)) - c\ell\,\dot{\theta}(t), \quad t \geq 0 \\ \theta(0) = \theta_0, \quad \dot{\theta}(0) = 0 \end{cases}$$

där c är dämpningskonstanten. Lös problemet för $\ell = 0.1$, $m = 0.1$ och $c = 0.2$ och några olika begynnelseutslagsvinklar. Använd `ode45`.

När vi gjorde figuren ovan i MATLAB skrev vi formlerna med s.k. LaTeX-kod. Så brukar matematiker skriva formler för att de skall bli snygga. Men vi får vi se det som överkurs.